LNBIP 433

Jorge Munoz-Gama
Xixi Lu (Eds.)

Process Mining Workshops

ICPM 2021 International Workshops
Eindhoven, The Netherlands, October 31 – November 4, 2021
Revised Selected Papers

Springer

OPEN ACCESS

Lecture Notes in Business Information Processing 433

More information about this series at https://link.springer.com/bookseries/7911

Jorge Munoz-Gama · Xixi Lu (Eds.)

Process Mining Workshops

ICPM 2021 International Workshops
Eindhoven, The Netherlands, October 31 – November 4, 2021
Revised Selected Papers

Springer

Editors
Jorge Munoz-Gama
Pontificia Universidad Católica de Chile
Santiago, Chile

Xixi Lu
Utrecht University
Utrecht, The Netherlands

ISSN 1865-1348 ISSN 1865-1356 (electronic)
Lecture Notes in Business Information Processing
ISBN 978-3-030-98580-6 ISBN 978-3-030-98581-3 (eBook)
https://doi.org/10.1007/978-3-030-98581-3

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

The International Conference on Process Mining (ICPM) was established two years ago as the conference where people from academia and industry could meet and discuss the latest developments in the area of process mining research and practice, including theory, algorithmic challenges, and applications. Although the ICPM conference series is very young, it has attracted innovative research of high quality from scholars and industrial researchers.

This year the conference was organized in Eindhoven, the Netherlands. Workshops were held on November 1, 2021, on the first of the conference days. While adhering to the restrictions introduced by the COVID-19 pandemic, it did not negatively affect the program that was offered by ICPM. In fact, the conference included co-located workshops presenting a wide range of outstanding research ideas in terms of the full paper presentations. In addition, the resulting workshop programs were complemented with keynotes, round-table panels, and poster sessions, providing a lively discussion forum for the whole community.

ICPM 2021 featured six workshops, each focusing on particular aspects of process mining, either a particular technical aspect or a particular application domain:

- 2nd International Workshop on Event Data and Behavioral Analytics (EDBA)
- 2nd International Workshop on Leveraging Machine Learning in Process Mining (ML4PM)
- 2nd International Workshop on Trust, Privacy, and Security in Process Analytics (TPSA)
- 4th International Workshop on Process-Oriented Data Science for Healthcare (PODS4H)
- 2nd International Workshop on Streaming Analytics for Process Mining (SA4PM)
- 6th International Workshop on Process Querying, Manipulation, and Intelligence (PQMI)

These proceedings present and summarize the work that was discussed during the workshops. In total, the ICPM 2021 workshops received 65 submissions, of which 28 papers were accepted for publication, leading to a total acceptance rate of about 43%. Supported by ICPM, each workshop also conferred a best workshop paper award. Finally, it is worth mentioning that to promote open-research, ICPM proudly offered to publish the entire proceedings as open-access.

In addition to the 28 papers accepted for the aforementioned six workshops, we are proud to announce that the organizers of the XES 2.0 workshop, which took place on November 2 during the second day of the conference, were also invited to publish their results in these proceedings. The XES 2.0 workshop focused on a survey conducted by the IEEE Task Force on Process Mining on the challenges faced during event data preparation (from source data to event log).

We would like to thank all the people from the ICPM community who helped to make the ICPM 2021 workshops a success. We particularly thank the general chair, Boudewijn van Dongen, and all the organization committee members for organizing such an outstanding conference despite the COVID-19 pandemic and the associated challenges. We also thank the workshop organizers, the numerous reviewers, and, of course, the authors for making the ICPM 2021 workshops such a success.

December 2021

Jorge Munoz-Gama
Xixi Lu

Organization

Workshop Chairs

Jorge Munoz-Gama	Pontificia Universidad Católica de Chile, Chile
Xixi Lu	Utrecht University, The Netherlands

Program Committee

Davide Aloini	University of Pisa, Italy
Robert Andrews	Queensland University of Technology, Australia
Periklis Andritsos	University of Toronto, Canada
Annalisa Appice	University of Bari Aldo Moro, Italy
Ahmed Awad	University of Tartu, Finland
Dorina Bano	HPI, University of Potsdam, Germany
Nathalie Baracaldo	IBM, USA
Sylvio Barbon Jr.	Universidade Estadual de Londrina, Brazil
Iris Beerepoot	Utrecht University, The Netherlands
Amin Beheshti	Macquarie University, Australia
Elisabetta Benevento	University of Pisa, Italy
Gaël Bernard	University of Toronto, Canada
Andrea Burattin	Technical University of Denmark, Denmark
Daniel Capurro	University of Melbourne, Australia
Michelangelo Ceci	Universita degli Studi di Bari Aldo Moro, Italy
Paolo Ceravolo	University of Milan, Italy
Ioannis Chatzigiannakis	Sapienza University of Rome, Italy
Marco Comuzzi	Ulsan National Institute of Science and Technology, South Korea
Benjamin Dalmas	Mines Saint-Etienne, France
Rene de la Fuente	Pontificia Universidad Católica de Chile, Chile
Massimiliano de Leoni	University of Padua, Italy
Jochen De Weerdt	Katholieke Universiteit Leuven, Belgium
Benoît Depaire	Hasselt University, Belgium
Shridhar Devamane	APS College of Engineering, India
Claudio Di Ciccio	Sapienza University of Rome, Italy
Chiara Di Francescomarino	Fondazione Bruno Kessler, Italy
Onur Dogan	Izmir Bakircay University, Turkey
Matthias Ehrendorfer	University of Vienna, Austria
Helm Emmanuel	Upper Austria University of Applied Sciences, Austria

Dirk Fahland	Eindhoven University of Technology, The Netherlands
Stephan Fahrenkrog-Petersen	Humbodt-Universität Berlin, Germany
Luís Paulo Faina Garcia	University of Brasília, Brazil
Bettina Fazzinga	University of Calabria, Italy
Carlos Fernandez-Llatas	Universitat Politècnica de València, Spain
Francesco Folino	ICAR-CNR, Italy
Luciano García-Bañuelos	Tecnológico de Monterrey, Mexico
Roberto Gatta	Università Cattolica del Sacro Cuore, Italy
Kanika Goel	Queensland University of Technology, Australia
Maria Teresa Gómez López	University of Seville, Spain
Monika Gupta	IBM, India
Antonella Guzzo	University of Calabria, Italy
Marwan Hassani	Eindhoven University of Technology, The Netherlands
Gert Janssenswillen	Universiteit Hasselt, Belgium
Owen Johnson	University of Leeds, UK
Anna Kalenkova	University of Melbourne, Australia
Agnes Koschmider	Kiel University, Germany
Mariangela Lazoi	CCII, Università del Salento, Italy
Francesco Leotta	Sapienza University of Rome, Italy
Xixi Lu	Utrecht University, The Netherlands
Fabrizio Maria Maggi	Free University of Bozen-Bolzano, Italy
Felix Mannhardt	Eindhoven University of Technology, The Netherlands
Ronny Mans	VitalHealth Software, The Netherlands
Mar Marcos	Universitat Jaume I, Spain
Gabriel Marques Tavares	Università degli Studi di Milano, Italy
Niels Martin	Hasselt University, Belgium
Renata Medeiros de Carvalho	Eindhoven University of Technology, The Netherlands
Jan Mendling	Humboldt-Universität zu Berlin, Germany
Judith Michael	RWTH Aachen University, Germany
Marco Montali	KRDB Research Centre, Free University of Bozen-Bolzano, Italy
Catarina Moreira Moreira	Queensland University of Technology, Australia
Jorge Munoz-Gama	Pontificia Universidad Católica de Chile, Chile
Hye-Young Paik	University of New South Wales, Australia
Emerson Paraiso	Pontificia Universidade Catolica do Parana, Brazil
Marco Pegoraro	RWTH Aachen University, Germany
Sarajane M. Peres	University of São Paulo, Brazil
Artem Polyvyanyy	University of Melbourne, Australia

Simon Poon	University of Sydney, Australia
Domenico Potena	Università Politecnica delle Marche, Italy
Maurizio Proietti	IASI-CNR, Italy
Luise Pufahl	TU Berlin, Germany
Ricardo Quintano	Philips Research, Brazil
Hajo Reijers	Utrecht University, The Netherlands
David Riaño	Universitat Rovira i Virgili, Italy
Florian Richter	Ludwig Maximilian University of Munich, Germany
Stefanie Rinderle-Ma	Technical University of Munich, Germany
Eric Rojas	Pontificia Universidad Católica de Chile, Chile
Lucia Sacchi	University of Pavia, Italy
Shazia Sadiq	University of Queensland
Sebastian Sardina	RMIT University, Australia
Thomas Seidl	Technical University of Denmark, Denmark
Arik Senderovich	Technion, Israel
Fernando Seoane	Karolinska Institutet, Sweden
Marcos Sepúlveda	Pontificia Universidad Católica de Chile, Chile
Natalia Sidorova	Eindhoven University of Technology, The Netherlands
Renuka Sindhgatta	Queensland University of Technology, Australia
Pnina Soffer	University of Haifa, Israel
Minseok Song	Pohang University of Science and Technology, South Korea
Frederic Stahl	DFKI GmbH, Germany
Alessandro Stefanini	University of Pisa, Italy
Emilio Sulis	University of Turin, Italy
Niek Tax	Meta, USA
Irene Teinemaa	Booking.com, The Netherlands
Arthur ter Hofstede	Queensland University of Technology, Australia
Pieter Toussaint	Norwegian University of Science and Technology, Norway
Vicente Traver	Universitat Politècnica de València, Spain
Florian Tschorsch	TU Berlin, Germany
Han van der Aa	University of Mannheim, Germany
Wil van der Aalst	RWTH Aachen University, Germany
Sebastiaan van Zelst	RWTH Aachen University/FIT, Germany
Seppe Vanden Broucke	Katholieke Universiteit Leuven, Belgium
Rob Vanwersch	Eindhoven University of Technology, The Netherlands
Eric Verbeek	Eindhoven University of Technology, The Netherlands

Hagen Völzer	IBM Research - Zurich, Switzerland
Matthias Weidlich	Humboldt-Universität zu Berlin, Germany
Mathias Weske	HPI, University of Potsdam, Germany
Moe Thandar Wynn	Queensland University of Technology, Australia
Nicola Zannone	Eindhoven University of Technology, The Netherlands
Bruno Zarpelao	State University of Londrina, Brazil

Additional Reviewers

Mathilde Boltenhagen
Graziella De Martino
Angelo Impedovo
Andrea Chiorrini
Antonio Pellicani
Aurora Rimini
Icaro Miranda
Vincenzo Pasquadibisceglie
Pedro Pio

Contents

XES 2.0 Workshop and Survey

EdbA 2021: 2nd International Workshop on Event Data and Behavioral Analytics

ML4PM 2021: 2nd International Workshop in Leveraging Machine Learning for Process Mining

SA4PM 2021: 2nd International Workshop on Streaming Analytics for Process Mining

PQMI 2021: 6th International Workshop on Process Querying, Manipulation, and Intelligence

PODS4H 2021: 4th International Workshop on Process-Oriented Data Science for Healthcare

TPSA 2021: 2nd International Workshop on Trust, Privacy, and Security in Process Analytics

XES 2.0 Workshop and Survey

Rethinking the Input for Process Mining: Insights from the XES Survey and Workshop

Moe Thandar Wynn[1](✉), Julian Lebherz[2], Wil M. P. van der Aalst[3], Rafael Accorsi[4], Claudio Di Ciccio[5], Lakmali Jayarathna[1], and H. M.W. Verbeek[6]

[1] Queensland University of Technology, Brisbane, Australia
{m.wynn,lakmali.herathjayarathna}@qut.edu.au

[2] A.P. Møller-Mærsk, Copenhagen, Denmark
IEEE_TFPM_SC@lebherz.me

[3] RWTH Aachen University, Aachen, Germany
wvdaalst@pads.rwth-aachen.de

[4] Accenture, Zurich, Switzerland
rafael.accorsi@accenture.com

[5] Sapienza University of Rome, Rome, Italy
claudio.diciccio@uniroma1.it

[6] Eindhoven University of Technology, Eindhoven, The Netherlands
h.m.w.verbeek@tue.nl

Abstract. Although the popularity and adoption of process mining techniques grew rapidly in recent years, a large portion of effort invested in process mining initiatives is still consumed by event data extraction and transformation rather than process analysis. The IEEE Task Force on Process Mining conducted a study focused on the challenges faced during event data preparation (from source data to event log). This paper presents findings from the online survey with 289 participants spanning the roles of practitioners, researchers, software vendors, and end-users. These findings were presented at the XES 2.0 workshop co-located with the 3rd International Conference on Process Mining. The workshop also hosted presentations from various stakeholder groups and a discussion panel on the future of XES and the input needed for process mining. This paper summarises the main findings of both the survey and the workshop. These outcomes help us to accelerate and improve the standardisation process, hopefully leading to a new standard widely adopted by both academia and industry.

Keywords: Process Mining · XES · Event Data · Data Transformation

1 Introduction

It is well known that data pre-processing is the most time-consuming task of a process mining project. The XES workshop, organised by the IEEE Task force

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 3–16, 2022.
https://doi.org/10.1007/978-3-030-98581-3_1

on Process Mining XES working group, aims to seek contributions from process mining vendors and researchers on the challenges faced in curating data input for process mining projects. The scope of the workshop covers the different aspects of the data input pipeline, starting from the raw event data to generating an event log (e.g., data curation, data cleaning, data standardisation). The intended outcome is a collection of data-related challenges and potential solutions to address these challenges. This paper summarises the main findings from this initiative.

The rest of the paper is organised as follows: Sect. 2 provides an overview of the current IEEE standard for eXtensible Event Stream (XES). Section 3 describes the key insights from the online survey, while Sect. 4 synthesises the discussion on the day of the XES workshop. Section 5 concludes the paper.

2 XES Standard: A Brief Overview

MXML (Mining eXtensible Markup Language), defined in 2003, was the first process mining standard to exchange event data [1]. Due to its limitations, the standardisation for new format called XES started in 2009 supported by the IEEE Task Force on Process Mining. Already in the first meeting of the Task Force on September 15th 2010 at the Stevens Institute of Technology in Hoboken USA there was consensus to establish XES as an official standard. The XES standard was adopted by the IEEE Standards Association (SA) as the "IEEE Standard for eXtensible Event Stream (XES) for Achieving Interoperability in Event Logs and Event Streams" [2] in 2016.

After the adoption of the XES standard by the IEEE, work was done on creating new extensions to the XES standard. The conceptual model of XES introduces components (logs, traces, events, and attributes) that may all contain attributes. Every such attribute is represented as a key-value mapping, where the value is assigned according to the attribute's type (string, timestamp, integer, real, boolean, ID, or list).

The purpose of the extensions was, and still is, to provide semantics to the attribute keys. A typical example for this is the "`concept:name`" key, which is generally considered to be the name of the corresponding activity (for an event) or the name of the corresponding case (for a trace). However, to provide this key with semantics, the `Concept` extension needs to be included in the XES log, as, by default, keys have no fixed semantics. To provide semantics to some basic attributes, the XES standard comes with a collection of standard extensions[1]. The `Concept` extension is a typical example thereof, and the standard additionally includes the `Lifecycle`, `Time`, `Organizational`, and `Cost`. In the end, this work led to the adoption of a number of additional extensions by the XES Working Group (WG), like `Micro` in 2016, `Software` in 2017, and `Artifact Lifecycle` in 2018.

However, the adoption of the XES standard by the different software tools in the process mining community remained low. Also, whenever a tool claimed

[1] www.tf-pm.org/resources/xes-standard/about-xes/standard-extensions.

to support the XES standard it was often unclear to what extent it supported the XES standard. To provide a better overview of this support of the XES standard, the XES Working Group initiated a XES certification process in 2017. As a result, at the time of writing **twelve** process mining tools[2] have been certified by the XES WG as supporting the XES standard. The XES standard helped to progress the field of process mining. It led to consensus about core concepts [1] and many publicly available event logs were made available for competitions and benchmarks. However, adoption in industry is limited, mostly due to the verbosity of the XML serialisation of XES. Moreover, the extraction and pre-processing of event data is still seen as a limiting factor for process mining.

3 Survey Design and Insights

To investigate the challenges faced during event data preparation for process mining, we conducted an online survey collecting the insights from the process mining community from various roles (i.e., academia, professional services, software vendors, and commercial end users).

Survey Design. The survey instrument was developed by the XES WG through several review iterations. The survey contained **12** questions and captured the participants' insights on the suggestions for speeding up the data pre-processing, particularly to understand what enhancement can be made to an industry-wide process mining data standard such as XES.

1. How much experience do you have with Process Mining?
2. Which area and role best describe how you have interacted with PM?
3. What share of effort is typically spent on data pre-processing?
4. Which process mining solutions have you used?
5. Which technologies have you used in data pre-processing for process mining?
6. In which format(s) is your source data available in?
7. Which source systems have you analysed with process mining?
8. How big was the largest data set you worked with in process mining?
9. To what extent did you encounter the following data-related challenges while undertaking PM projects in terms of sourcing data, processing data, analysing process data?
10. Which data-related challenges have you encountered beyond the ones listed in question 9?
11. There is general consensus amongst practitioners that data pre-processing tasks still consume most of the effort put into process mining initiatives. How could we speed up the data pre-processing to focus on analysis?
12. How could a re-imagined industry-wide process mining data standard help you excel in your role?

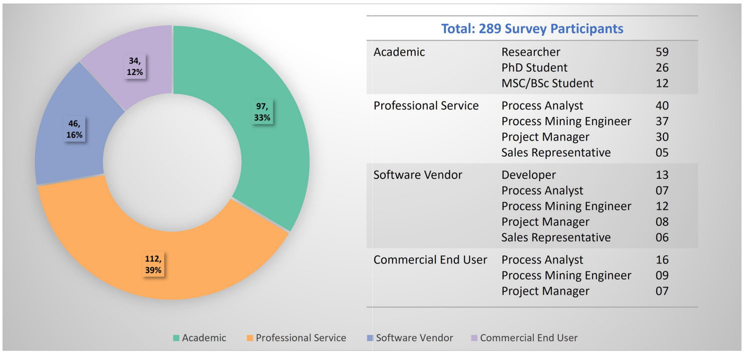
[2] www.tf-pm.org/resources/xes-standard/for-vendors/certification/tools.

The XES online survey was distributed to the international process mining community (through LinkedIn posts, email lists and website announcements) and was opened from June to July 2021. In total, 290 responses were received. A duplicate response was detected and removed, thus the total number of responses used for the analysis is **289**.

Survey Insights. The responses for Questions 1 to 9 were quantitatively analysed using the descriptive and frequency analysis. In addition, the responses are grouped based on a participant's role. Free-text responses provided in Questions 10, 11, and 12 were analysed by a research assistant to identify the emerging themes and then reviewed by two XES WG members. This led to the final grouping of common themes presented later in the section.

Out of the 289 responses, the highest response rate is from the professional service role ($n = 112$, 39%), followed by academia ($n = 97$, 33%), software vendors ($n = 46$, 16%), and commercial end users ($n = 34$, 12%), as depicted in Fig. 1. The highest range of experience reported was 2–5 years (38%), followed by 5–10 years (24%), 1–2 years (18%), 10+ years (10%), and less than one year (9%). Participants with no experience are less than 1%.
Next, we present individual key findings for Questions 3–12.

Q3: What share of effort is typically spent on data pre-processing? Figure 2 shows that 61% to 80% of the effort of share for data pre-processing is the highest reported response by participants (36%) across all roles. The maximum percentage reported was 90% for the academic role and the professional service role. These results confirm that a significant amount of effort is being spent to pre-process event data for process mining. It is also interesting to notice that most of the participants with less than one year of experience did not respond to this question. This may indicate that process mining novices are more focused on the novel techniques and tool development than on the input data.



Total: 289 Survey Participants

Academic	Researcher	59
	PhD Student	26
	MSC/BSc Student	12
Professional Service	Process Analyst	40
	Process Mining Engineer	37
	Project Manager	30
	Sales Representative	05
Software Vendor	Developer	13
	Process Analyst	07
	Process Mining Engineer	12
	Project Manager	08
	Sales Representative	06
Commercial End User	Process Analyst	16
	Process Mining Engineer	09
	Project Manager	07

Fig. 1. XES survey participants: demographics

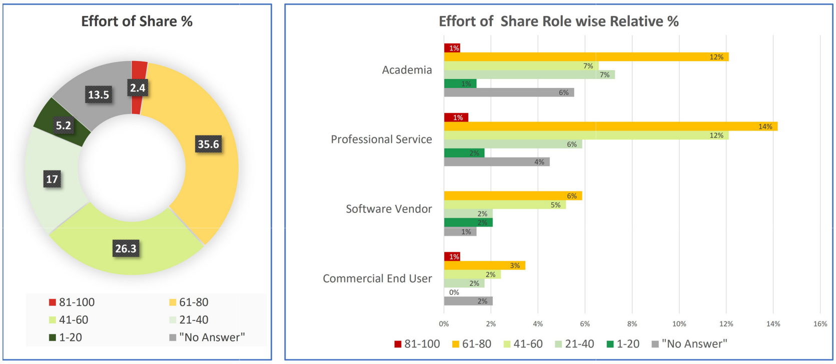


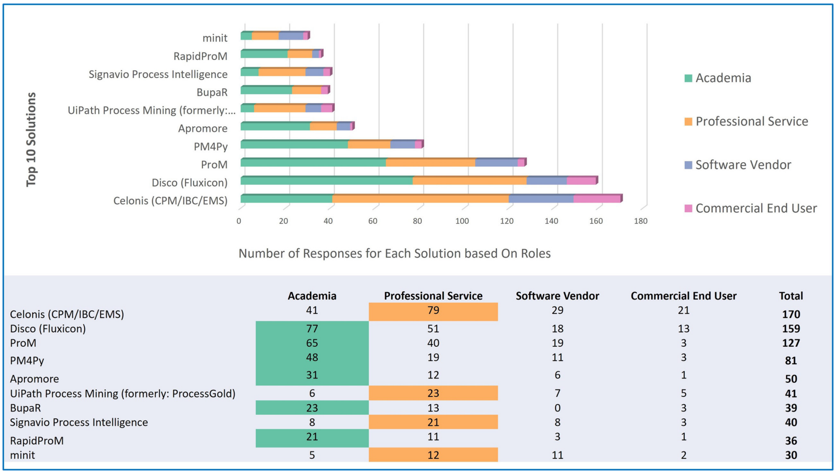
Fig. 2. Q3: Share of effort on data pre-processing

Q4: Which process mining solutions have you used? Celonis is the overall highest selection ($n = 170$), with Disco ($n = 159$) and ProM ($n = 127$) rounding off the top three process mining solutions reported by the participants (see Fig. 3). Note that it is possible for participants to select multiple solutions, and many opted for this. The role-wise comparison for the top ten process mining solutions, where variations can be observed among the four roles. For example, Disco (Fluxicon) is the most selected option for academics ($n = 77$), closely followed by ProM ($n = 65$).

Q5: Which technologies have you used in data pre-processing for process mining? Microsoft SQL server is the highest selected response for database management and data storage systems ($n = 125$). Figure 4 shows a slightly different perspective among the four roles, with academia selecting MySQL ($n = 45$) ahead of Microsoft SQL server and the software vendors preferring PostgresSQL ($n = 26$). PowerBI ($n = 122$) has been the most selected response as a data visualisation tool (see Fig. 5). Python ($n = 177$) turned out to be the most used custom data transformation language (see Fig. 6).

Q6: In which formats is your source data available in? A plain text file (e.g., `txt` or `csv`) is the most commonly available source data format ($n = 229$), with the relational format access ($n = 168$), and the XML format such as XES ($n = 112$) being selected as the second and the third most common ones (see Fig. 7). Please notice that participants could select more than one source data format. The responses also confirm that XML (e.g., XES) is not widely used in the community with only 39% ($n = 112$) selecting this option.

The frequencies and their relative order among the top five source formats are also different across the different roles (as shown in Fig. 8). For example, XML is the second, third, fourth, and third choice for academia, professional services, software vendors, and commercial end users, respectively.



	Academia	Professional Service	Software Vendor	Commercial End User	Total
Celonis (CPM/IBC/EMS)	41	79	29	21	**170**
Disco (Fluxicon)	77	51	18	13	**159**
ProM	65	40	19	3	**127**
PM4Py	48	19	11	3	**81**
Apromore	31	12	6	1	**50**
UiPath Process Mining (formerly: ProcessGold)	6	23	7	5	**41**
BupaR	23	13	0	3	**39**
Signavio Process Intelligence	8	21	8	3	**40**
RapidProM	21	11	3	1	**36**
minit	5	12	11	2	**30**

Fig. 3. Q4: 10 most used process mining solutions

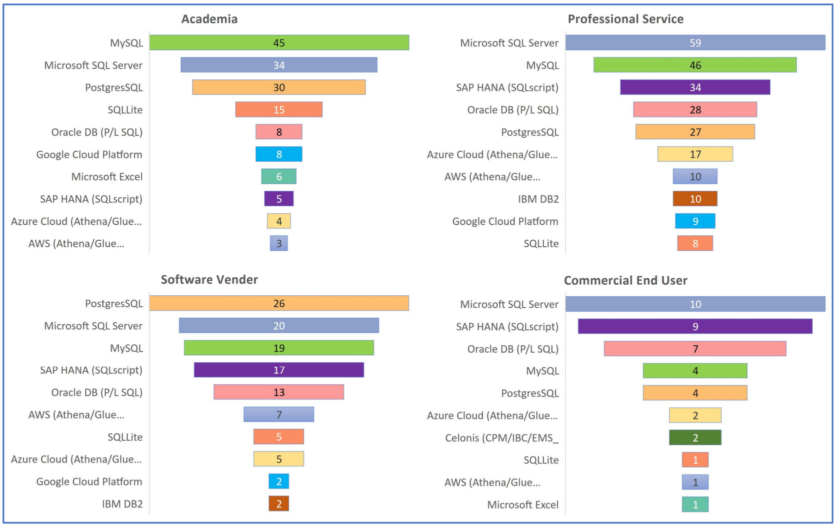


Fig. 4. Q5: 10 most used database management and data storage systems (role-wise)

Q7: Which source systems have you analysed with process mining? SAP ECC (R/3) ($n = 114$), SAP S/4 HANA ($n = 101$), and Salesforce ($n = 71$) are the top three most analysed source systems (see Fig. 9). Interestingly, 35% of academics (34 out of 97) selected "I don't know" as their response for this question. This is probably due to the fact that they primarily work with publicly available data sets such as those provided by the BPI challenges.

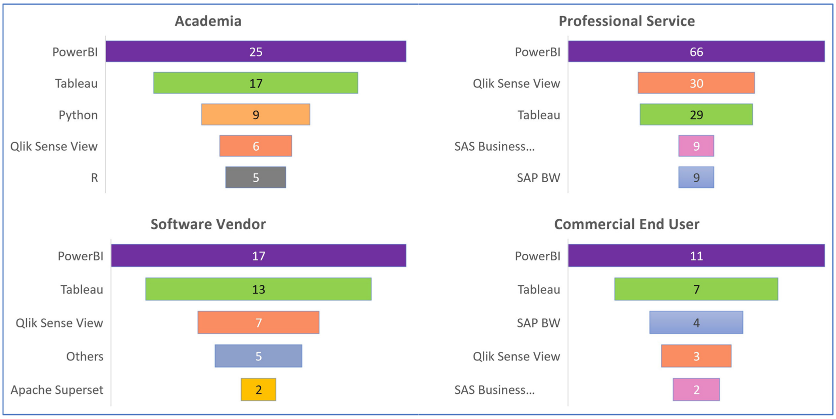


Fig. 5. Q5: 10 most used data visualisation technologies (role-wise)

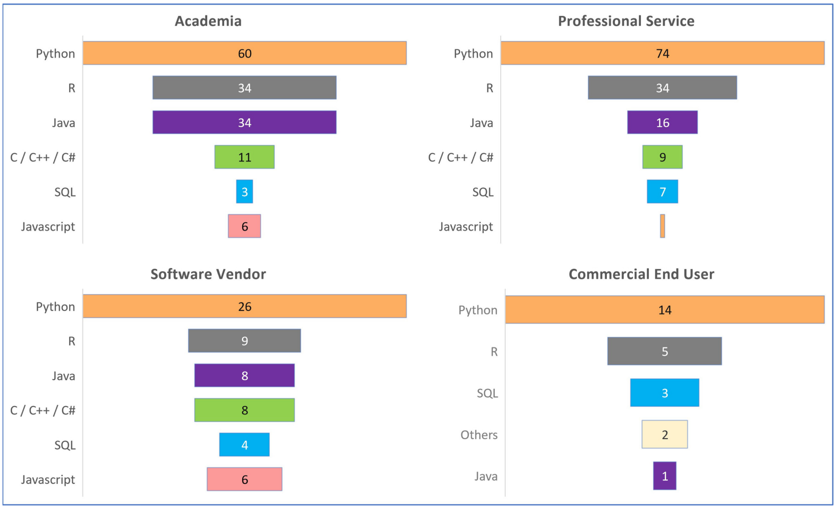
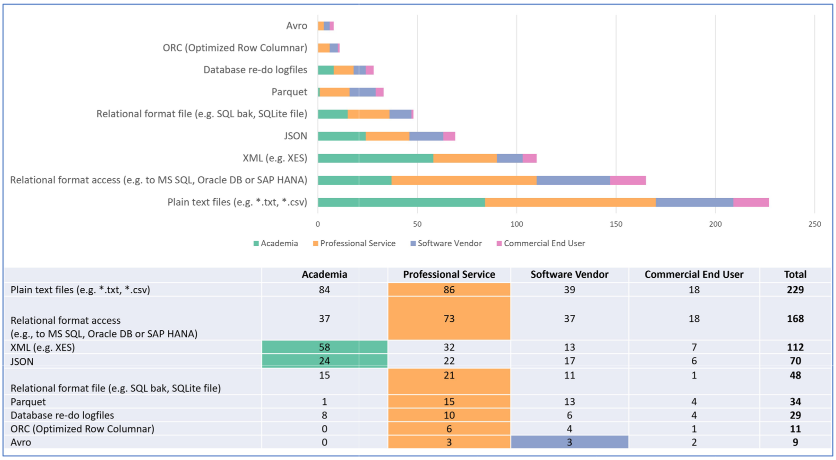


Fig. 6. Q5: 10 most adopted custom data transformation languages (role-wise)

Q8: How big was the largest data set you worked with in process mining? Around 16% of participants ($n = 45$) have mentioned that they have worked with less than 1000 events and 0.05% participants have mentioned that they have worked with more than 1,000,000,000 (1 billion) events. Moreover, around 20% of the participants ($n = 58$) have worked with less than 1000 process cases or instances and around 4% of the participants ($n = 12$) mentioned that the highest number of process cases or instances they have worked with is larger than 1 billion.



	Academia	Professional Service	Software Vendor	Commercial End User	Total
Plain text files (e.g. *.txt, *.csv)	84	86	39	18	**229**
Relational format access (e.g., to MS SQL, Oracle DB or SAP HANA)	37	73	37	18	**168**
XML (e.g. XES)	58	32	13	7	**112**
JSON	24	22	17	6	**70**
Relational format file (e.g. SQL bak, SQLite file)	15	21	11	1	**48**
Parquet	1	15	13	4	**34**
Database re-do logfiles	8	10	6	4	**29**
ORC (Optimized Row Columnar)	0	6	4	1	**11**
Avro	0	3	3	2	**9**

Fig. 7. Q6: 10 most used source data formats

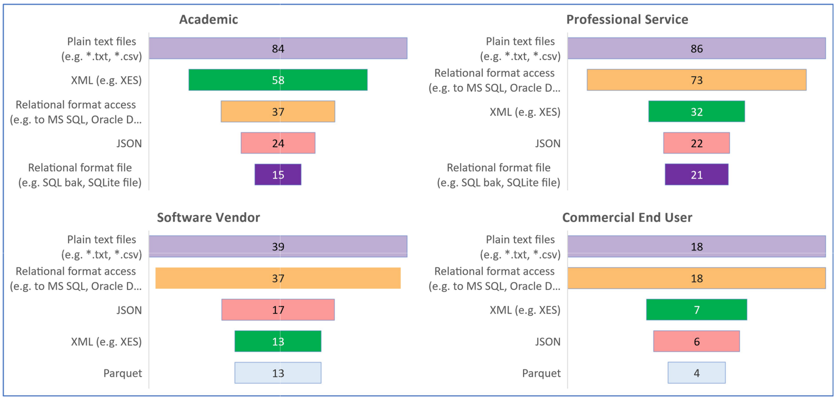
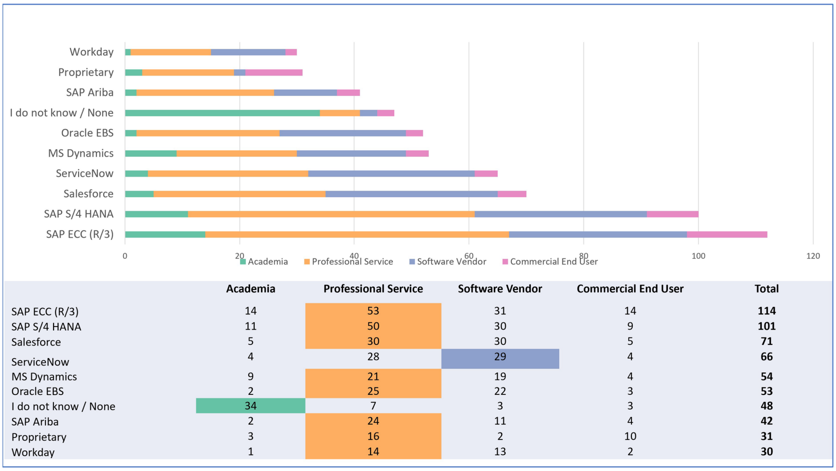


Fig. 8. Q6: 10 most used source data formats (role-wise)

Q9: To what extent did you encounter data-related challenges while undertaking PM projects in terms of sourcing data, processing data, and analysing process data? Figure 10 depicts an overview of sixteen data-related challenges identified across three categories: sourcing data, process data, and analysing data. The participants were asked to select a single option, ranging from none to very significant, for each data challenge.

Among the six challenges linked to the sourcing of process data, the challenge of complex data structures stands out as the most problematic, with 64% of the participants ($n = 185$) selecting either significant or very significant. Moreover, 54% of them selected the undocumented data structures as a key challenge (significant or very significant). On the other hand, 61% indicated that the challenge of identifying the required data in the source systems as either moderate, minor



	Academia	Professional Service	Software Vendor	Commercial End User	Total
SAP ECC (R/3)	14	53	31	14	**114**
SAP S/4 HANA	11	50	30	9	**101**
Salesforce	5	30	30	5	**71**
ServiceNow	4	28	29	4	**66**
MS Dynamics	9	21	19	4	**54**
Oracle EBS	2	25	22	3	**53**
I do not know / None	34	7	3	3	**48**
SAP Ariba	2	24	11	4	**42**
Proprietary	3	16	2	10	**31**
Workday	1	14	13	2	**30**

Fig. 9. Q7: 10 most used source systems

or none, while 49% felt the same about the challenge of exporting data from source systems.

Among the five processing data-related challenges, 45% ($n = 140$) identified inconsistent data as being a relevant challenge (significant or very significant) while 42% identified incomplete data as being a significant challenge. However, 75% of all participants ($n = 217$) indicated that the performance issues are not very significant by selecting either moderate, minor or none for that challenge.

Among the six data-related challenges linked to the analysis, the limitation related to analysing one-to-many and many-to-many relationships has been identified as a crucial challenge (48% selecting either significant or very significant) while 76% indicated that exporting data from a process mining tool is the least significant challenge by selecting either moderate, minor, or none.

Q10: Which data-related challenges have you encountered beyond the ones listed in question 9? Figure 11 shows the frequency of the new challenges proposed by the participants. Among them, lack of documentation and data quality feature as the two top themes.

Q11: How could we speed up the data pre-processing to focus on analysis? The main themes identified from the responses ($n = 199$) relate to the standardisation of data formats as well as data transformation pipelines, suggestions for better tool support, scaling up of domain and process mining expertise, and suggestions to improve data quality. Figure 12 captures the main themes with exemplar comments received from the participants.

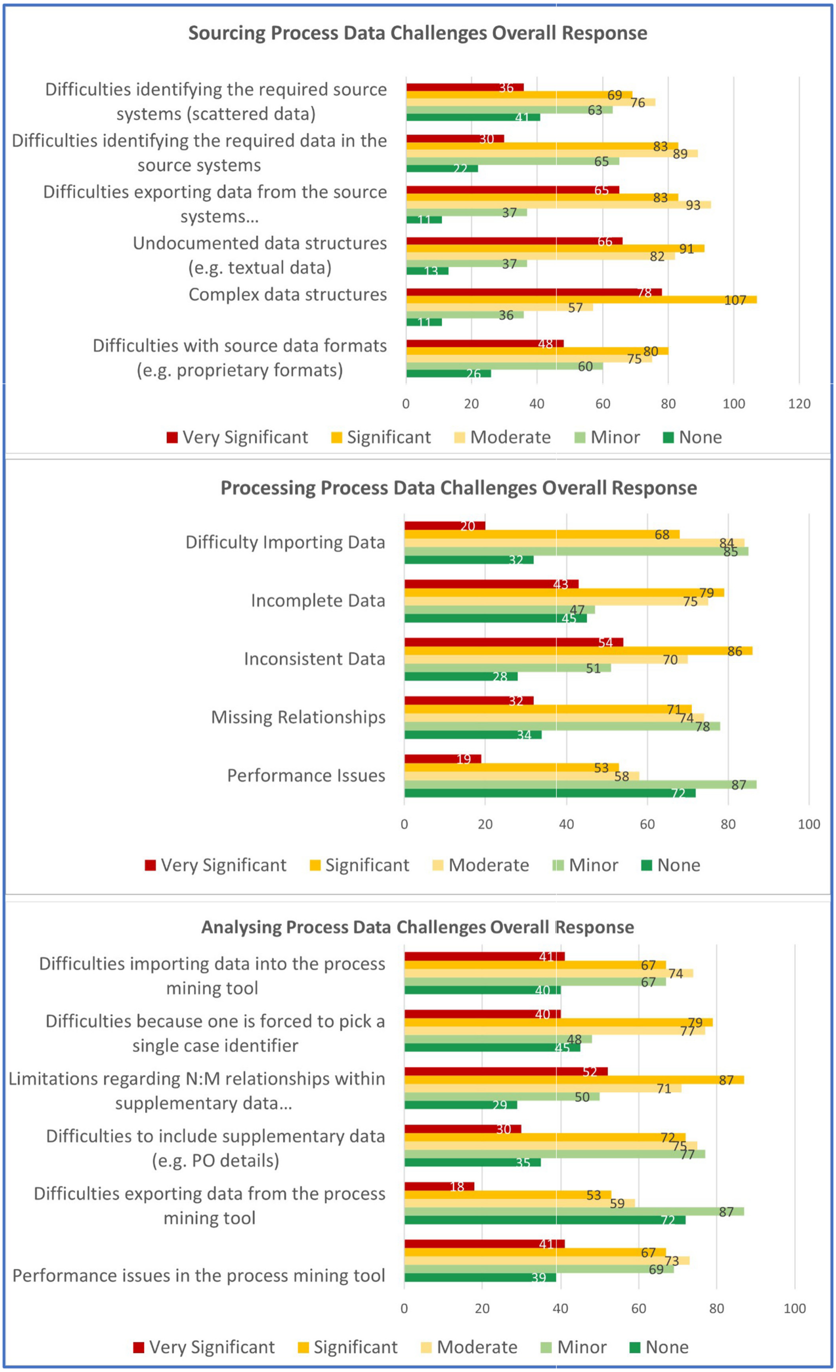
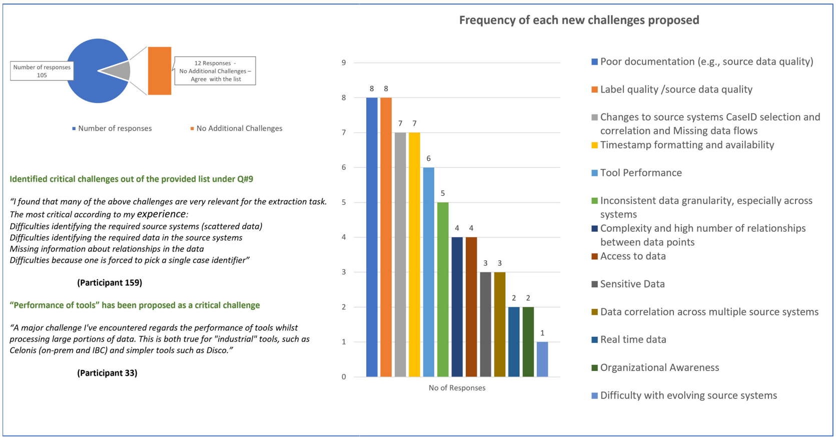


Fig. 10. Q9: Ranking the significance of data-related challenges



Identified critical challenges out of the provided list under Q#9

"I found that many of the above challenges are very relevant for the extraction task.
The most critical according to my experience:
Difficulties identifying the required source systems (scattered data)
Difficulties identifying the required data in the source systems
Missing information about relationships in the data
Difficulties because one is forced to pick a single case identifier"

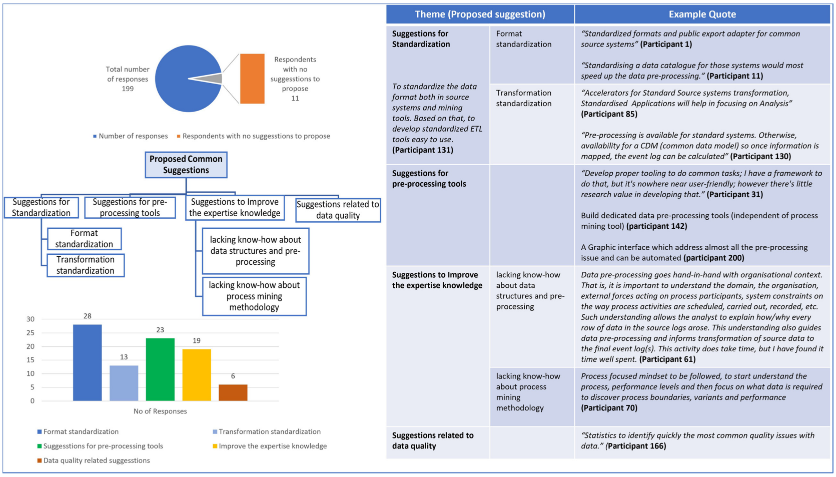
(Participant 159)

"Performance of tools" has been proposed as a critical challenge

"A major challenge I've encountered regards the performance of tools whilst processing large portions of data. This is both true for "industrial" tools, such as Celonis (on-prem and IBC) and simpler tools such as Disco."

(Participant 33)

Fig. 11. Q10: Qualitative insights of other data-related challenges

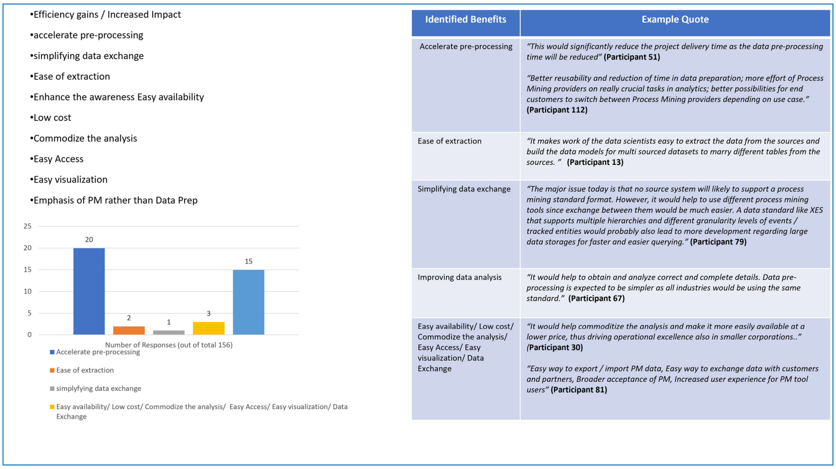


Theme (Proposed suggestion)		Example Quote
Suggestions for Standardization *To standardize the data format both in source systems and mining tools. Based on that, to develop standardized ETL tools easy to use.* **(Participant 131)**	Format standardization	*"Standardized formats and public export adapter for common source systems"* **(Participant 1)** *"Standardising a data catalogue for those systems would most speed up the data pre-processing."* **(Participant 11)**
	Transformation standardization	*"Accelerators for Standard Source systems transformation, Standardised Applications will help in focusing on Analysis"* **(Participant 85)** *"Pre-processing is available for standard systems. Otherwise, availability for a CDM (common data model) so once information is mapped, the event log can be calculated"* **(Participant 130)**
Suggestions for pre-processing tools		*"Develop proper tooling to do common tasks; I have a framework to do that, but it's nowhere near user-friendly; however there's little research value in developing that."* **(Participant 31)** Build dedicated data pre-processing tools (independent of process mining tool) **(participant 142)** A Graphic interface which address almost all the pre-processing issue and can be automated **(participant 200)**
Suggestions to Improve the expertise knowledge	lacking know-how about data structures and pre-processing	*Data pre-processing goes hand-in-hand with organisational context. That is, it is important to understand the domain, the organisation, external forces acting on process participants, system constraints on the way process activities are scheduled, carried out, recorded, etc. Such understanding allows the analyst to explain how/why every row of data in the source logs arose. This understanding also guides data pre-processing and informs transformation of source data to the final event log(s). This activity does take time, but I have found it time well spent.* **(Participant 61)**
	lacking know-how about process mining methodology	*Process focused mindset to be followed, to start understand the process, performance levels and then focus on what data is required to discover process boundaries, variants and performance* **(Participant 70)**
Suggestions related to data quality		*"Statistics to identify quickly the most common quality issues with data."* (**Participant 166)**

Fig. 12. Q11: Qualitative insights for suggestions to speed up the data pre-processing

Q12: How could a re-imagined industry-wide process mining data standard help you excel in your role? The participants foresaw a variety of potential benefits ranging from the acceleration of data pre-processing to commodisation of analysis ($n = 156$) (see Fig. 13).

Discussion. The survey results reconfirm the common belief that the data pre-processing task is highly time consuming (with the maximum amount of effort estimated to be 90%) while 36% estimated their efforts to be within the range of range60% to 80% (cf. Q3). The responses also confirm that the XML format (i.e., the one of XES) is not widely used in the community to store event logs,



- •Efficiency gains / Increased Impact
- •accelerate pre-processing
- •simplifying data exchange
- •Ease of extraction
- •Enhance the awareness Easy availability
- •Low cost
- •Commodize the analysis
- •Easy Access
- •Easy visualization
- •Emphasis of PM rather than Data Prep

Identified Benefits	Example Quote
Accelerate pre-processing	*"This would significantly reduce the project delivery time as the data pre-processing time will be reduced"* **(Participant 51)** *"Better reusability and reduction of time in data preparation; more effort of Process Mining providers on really crucial tasks in analytics; better possibilities for end customers to switch between Process Mining providers depending on use case."* **(Participant 112)**
Ease of extraction	*"It makes work of the data scientists easy to extract the data from the sources and build the data models for multi sourced datasets to marry different tables from the sources. "* **(Participant 13)**
Simplifying data exchange	*"The major issue today is that no source system will likely to support a process mining standard format. However, it would help to use different process mining tools since exchange between them would be much easier. A data standard like XES that supports multiple hierarchies and different granularity levels of events / tracked entities would probably also lead to more development regarding large data storages for faster and easier querying."* **(Participant 79)**
Improving data analysis	*"It would help to obtain and analyze correct and complete details. Data pre-processing is expected to be simpler as all industries would be using the same standard."* **(Participant 67)**
Easy availability/ Low cost/ Commodize the analysis/ Easy Access/ Easy visualization/ Data Exchange	*"It would help commoditize the analysis and make it more easily available at a lower price, thus driving operational excellence also in smaller corporations.."* (**Participant 30)** *"Easy way to export / import PM data, Easy way to exchange data with customers and partners, Broader acceptance of PM, Increased user experience for PM tool users"* **(Participant 81)**

Fig. 13. Q12: Qualitative insights for the expected benefits of an industry-wide standard

with only 39% selecting this option (cf. Q6). There seems to be consensus among the process mining community that there are significant data-related challenges associated with complex data structures, complex one-to-many and many-to-many relationships, inconsistent data, incomplete data and missing relationships (cf. Q9). These data challenges should be carefully considered and addressed when a new standard is being prepared.

The participants also indicated a need for systematic and automated data pre-processing techniques for efficient and reproducible data transformations for process mining. A dedicated methodology for data pre-processing to support a structured approach to PM methodology (*Stage 0*) seems definitely desirable, with the ability to create templates that capture best practices to ultimately speed up the data pre-processing task (cf. Q11). Approaches to assess and improve the data quality issues identified in the survey (e.g., inconsistent data or incomplete data) could be beneficial. Furthermore, a new event log standard should leave room for various mechanisms to import/export the event data, not only using XML.

4 Adding Context: Reflections from the XES 2.0 Workshop

In order to challenge and validate the survey's takeaways presented above, the XES WG hosted a workshop co-located with the Third Int. Conference on Process Mining in Eindhoven (Netherlands) on November 2, 2021. A session on survey results set the scene, followed by contributions from software vendors (represented by Celonis and Signavio), academia (represented by RWTH Aachen and the Free University of Bozen-Bolzano) and professional services (represented by

KPMG). Concluding with a panel discussion centred around select findings from the survey, the workshop not only offered well-balanced viewpoints from different players in the discipline, but most notably revealed an unexpected homogeneity concerning the most relevant levers for a successful evolution of the XES standard.

First, rethink the Core Concepts of an Event Log. Numerous participants raised questions about the fundamental scope of what information is captured in event logs. The support of extensions render the current XES standard extremely flexible – even towards future, non-anticipated requirements – but at the cost of complexity. With limited awareness and use of existing extensions, this split needs to be revisited. It also became apparent that even though XES itself does not stipulate any storage format, most participants equate XES with its XML schema definition and call out its misfit with data volume and velocity of current, practical use cases. This showcases the need to strictly focus on storage-agnostic core concepts first and to later create multiple relevant reference implementations.

In addition, recent trends in industry and academia (e.g., object-centric event logs like OCEL, multi-event logs, and knowledge graphs) point to the need for complex data structures and relationships to be captured in an event log. A consensus has been reached to revisit the core concepts in an event log and propose a conceptual data model alongside a metadata schema that can support complex data structures (including many-to-many relationships between multiple objects, cases, and events).

Event Logs as a Semantic Layer. The current standard focuses mainly on syntactic interoperability and, to a lesser extent, on semantic aspects. However, enriching event logs with semantics would open up a whole array of possibilities across academia and industry (e.g., novel algorithms, autonomous data transformation, dynamic perspective change, real-time data extraction). Additionally, domain-specificity could tailor the semantics extensions to selective industries and thereby mimic real-world domain ontologies.

Taking this concept one step further, domain ontologies linked with event data could support process analytics without case identifiers. Different event logs could be generated as views over the same event data store. This intermediate layer would also hide the ultimate sources of the event data (let them be single or multiple, homogeneous or heterogeneous, legacy or newly implemented).

Generating Momentum amongst Industry Players. Contributions, Q&A and panel discussion also evidenced an intrinsic challenge of generating momentum around an industry standard for interoperability. It is acknowledged that the current XES standard is hardly used in industry or professional services. Vendors see themselves in a balancing act with true interoperability on the one side, arguably a catalyst for the whole industry, and proprietary solutions on the other side, often attributed with preventing customer churn. In the end, the community needs to find ways to present all sides with compelling cases to not only support, but jointly design the next evolution of XES. Not only vendors of process analytics tools should be involved, but also those implementing systems for process execution. Their support could become the linchpin to propel the industry.

5 Conclusion

This paper presents a summary of findings from an online survey with 289 participants, who span across the roles of practitioners, researchers, software vendors and end-users. It also provides a synthesis of the discussion among the participants during the XES workshop at the International Conference on Process Mining 2021 and sketches the next steps for the XES WG.

Acknowledgements. The authors would like to thank all survey respondents who contributed their ideas presented in this paper. We would also like to thank the following presenters and panelists who participated during the XES workshop: Marco Montali, Philipp Hoch, Elham Ramezani, Björn Wagner, Sven Wagner-Boysen, Constantin Wehmschulte, and Sebastiaan van Zelst. The presentation slides are available at www.tf-pm.org/resources/xes-standard/xes-2-0-workshop.

References

1. van der Aalst, W.M.P.: Process Mining - Data Science in Action, 2nd edn. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
2. IEEE 1849 (XES) WG: IEEE standard for eXtensible event stream (XES) for achieving interoperability in event logs and event streams. IEEE Std. **1849**, 1–50 (2016)

EdbA 2021: 2nd International Workshop on Event Data and Behavioral Analytics

Second International Workshop on Event Data and Behavioral Analytics (EdbA’21)

Over the past decades, capturing, storing, and analyzing event data has gained attention in various domains such as process mining, clickstream analytics, IoT analytics, e-commerce, and retail analytics, online gaming analytics, security analytics, website traffic analytics, and preventive maintenance, to name a few. The interest in event data lies in its analytical potential as it captures the dynamic behavior of people, objects, and systems at a fine-grained level.

Behavior often involves multiple entities, objects, and actors to which events can be correlated in various ways. In these situations, a unique, straightforward process notion does not exist, is unclear or different processes or dynamics could be recorded in the same data set.

The Event Data & Behavioral Analytics (EdbA) workshop’s objective is to provide a forum to practitioners and researchers for studying a quintessential, minimal notion of events as the common denominator for records of discrete behavior in all its forms. The workshop aims to stimulate the development of new techniques, algorithms, and data structures for recording, storing, managing, processing, analyzing, and visualizing event data in various forms. To this end, different types of submissions are welcome such as original research papers, case study reports, position papers, idea papers, challenge papers, and work in progress papers on event data and behavioral analytics. For more information, visit http://edba.science.

The second edition of the EdbA workshop attracted 18 submissions. After careful multiple reviews by the workshop’s program committee members, seven were accepted for a full-paper presentation at the workshop, while 1 submission was accepted as a work-in-progress presentation. All full-paper papers have been included in the proceedings. This year’s papers again cover a broad spectrum of topics, which can be organized into three main themes: pattern discovery, beyond traditional event logs, and IoT.

In the final plenary discussion session, the workshops participants reflected on the changing nature of research questions, evaluation criteria, and benchmarks when leaving the classical process mining setting: more than one object, more than one possible value for an attribute, more than one model, more than one data source.

The organizers wish to thank all the people who submitted papers to the EdbA’21 workshop, the many participants creating fruitful discussion and sharing insights and the EdbA’21 Program Committee members for their valuable work in reviewing the submissions. A final word of thanks goes out to the organizers of ICPM 2021 for making this workshop possible.

Organization

Workshop Chairs

Benoît Depaire	Hasselt University, Belgium
Dirk Fahland	Eindhoven University of Technology, the Netherlands
Francesco Leotta	Sapienza University of Rome, Italy
Arik Senderovich	University of Toronto, Canada

Program Committee

Piriklis Andritsos	University of Toronto, Canada
Gaël Bernard	University of Toronto, Canada
Ioannis Chatzigiannakis	Sapienza University of Rome, Italy
Massimiliano de Leoni	University of Padua, Italy
Jochen De Weerdt	Katholieke Universiteit Leuven, Belgium
Claudio Di Ciccio	Sapienza University of Rome, Italy
Chiara Di Francescomarino	Fondazione Bruno Kessler-IRST, Italy
Bettina Fazzinga	DICES, University of Calabria, Italy
Marwan Hassani	Eindhoven University of Technology, the Netherlands
Gert Janssenswillen	Hasselt University, Belgium
Xixi Lu	Utrecht University, the Netherlands
Fabrizio Maria Maggi	Free University of Bozen-Bolzano, Italy
Felix Mannhardt	Eindhoven University of Technology, the Netherlands
Niels Martin	Hasselt University, Belgium
Jan Mendling	Humboldt-Universität zu Berlin, Germany
Marco Montali	KRDB Research Centre, Free University of Bozen-Bolzano, Italy
Jorge Munoz-Gama	Pontificia Universidad Católica de Chile, Chile
Matthias Weidlich	Humboldt-Universität zu Berlin, Germany

Probability Estimation of Uncertain Process Trace Realizations

Marco Pegoraro$^{(\boxtimes)}$, Bianka Bakullari, Merih Seran Uysal,
and Wil M. P. van der Aalst

Process and Data Science Group (PADS), Department of Computer Science,
RWTH Aachen University, Aachen, Germany
{pegoraro,bianka.bakullari,uysal,wvdaalst}@pads.rwth-aachen.de
http://www.pads.rwth-aachen.de/

Abstract. Process mining is a scientific discipline that analyzes event data, often collected in databases called event logs. Recently, *uncertain event logs* have become of interest, which contain non-deterministic and stochastic event attributes that may represent many possible real-life scenarios. In this paper, we present a method to reliably estimate the probability of each of such scenarios, allowing their analysis. Experiments show that the probabilities calculated with our method closely match the true chances of occurrence of specific outcomes, enabling more trustworthy analyses on uncertain data.

Keywords: Process Mining · Uncertain Data · Partial Order

1 Introduction

Process mining is a discipline that focuses on extracting insights about processes in a data-driven manner. For instance, on the basis of the recorded information on historical process executions, process mining allows to automatically extract a model of the behavior of process instances, or to measure the compliance of the process data with a prescribed normative model of the process. In process mining, the central focus is on the *event log*, a collection of data that tracks past process instances. Every activity performed in a process is recorded in the event log, together with information such as the corresponding process case and the timestamp of the activity, in a sequence of events called a *trace*.

Recently, research on novel forms of event data have garnered the attention of the scientific community. Among these there are *uncertain event logs*, which contain data affected by imprecision [8]. This data contains meta-information describing the nature and entity of the uncertainty. Such meta-information can be obtained from the inherent precision with which the data has been recorded (e.g., timestamps only indicating the date have a possible "true value" range of

We thank the Alexander von Humboldt (AvH) Stiftung for supporting our research interactions.

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 21–33, 2022.
https://doi.org/10.1007/978-3-030-98581-3_2

24 h), from the precision of the tools involved in supporting the process (e.g., the absolute error of sensors), or from the domain knowledge provided by a process expert. An uncertain trace corresponds to multiple possible real-life scenarios, each of which might have very diverse implications on features of cases such as compliance to a model. It is then important to be able to assess the risk of occurrence of specific outcomes of uncertain traces, which enables to estimate the impact of such traces on indicators such as cost and conformance.

In this paper, we present a method to obtain a complete probability distribution over the possible instantiations of uncertain attributes in a trace. As a possible example of application, we frame our results in the context of conformance checking, and show the impact of assessing probability estimates for uncertain traces on insights about the compliance of an uncertain trace to a process model. We validate our method with experiments based on a Monte Carlo simulation, which shows that the probability estimates are reliable and reflect the true chances of occurrence of a specific outcome.

The remainder of the paper is structured as follows. Section 2 examines relevant related work. Section 3 illustrates a motivating running example for our technique. Section 4 presents preliminary definitions of different types of uncertainty in process mining. Section 5 illustrates a method for computing probabilities of realizations for uncertain process traces. Section 6 validates our method through experimental results. Finally, Sect. 7 concludes the paper.

2 Related Work

The analysis of uncertain data in process mining is a very recent research direction. The specific formulation and definition of uncertain data utilized in this paper has been introduced in 2019 [8], in the context of an analysis approach consisting in computing bounds for the conformance score of uncertain traces through alignments [5]. Subsequently, that work has been extended with an inductive mining approach for process discovery over uncertainty [9] and a taxonomy of different types of uncertain data, with their characteristics [10].

Uncertain data, as formulated in our present and previous work, is closely related to a considerably more studied data anomaly in process mining: partially ordered event data. In fact, uncertain data as described here is a generalization of partially ordered traces. Lu et al. [7] proposed a conformance checking approach based on alignments to measure conformance of partially ordered traces. More recently, Van der Aa et al. [1] illustrated a method for inferring a linear extension, i.e., a compliant total order, of events in partially ordered traces, based on examples of correct orderings extracted from other traces in the log. Busany et al. [4] estimated probabilities for partially ordered events in IoT event streams.

An associated topic, which draws from disciplines such as pattern and sequence mining and is antithetical to the analysis of partially ordered data, is the inference of partial orders from fully sequential data as a way to model its behavior. This goes under the name of *episode mining*, which can be performed with many techniques both on batched data and with online streams of events [2,6,11].

In this paper, we present a method to estimate the likelihood of any scenario in an uncertain setting, which covers partially ordered traces as well as other types of uncertainty illustrated in the taxonomy [10]. Furthermore, we will cover both the non-deterministic case (*strong uncertainty*) and the probabilistic case (*weak uncertainty*).

3 Running Example

In this section, we will provide a running example of uncertain process instance related to a sample process. We will then apply our probability estimation method to this uncertain trace, to illustrate its operation. The example we analyze here is a simplified generalization of a remote credit card fraud investigation process. This process is visualized by the Petri net in Fig. 1.

Firstly, the credit card owner alerts the credit card company of a possibly fraudulent transaction. The customer may either notify the company by calling their hotline (*alert hotline*) or arrange an urgent meeting with personnel of the bank that issued the credit card (*alert bank*). In both scenarios, his credit is frozen (*freeze credit*) to prevent further fraud. All information provided by the customer about the transaction is summarized when filing the formal report (*file report*). As a next step, the credit card company tries to contact the merchant that charged the credit card. If this happens (*contact merchant*), the credit card company clarifies whether there has been just a mistake (e.g., merchant charging not delivering a product, or a billing mistake) on the merchant's side. In such cases, the customer gets a *refund from merchant* and the case is closed. Another outcome might be the discovery of a *friendly fraud*, which is when a cardholder makes a purchase and then disputes it as fraud even though it was not. If contacting the merchant is impossible, a *fraud investigation* is initiated. In this case, fraud investigators will usually start with the transaction data and look for timestamps, geolocation, IP addresses, and other elements that can be used to prove whether or not the cardholder was involved in the transaction. The outcome might be either friendly fraud or *true fraud*. True fraud can also happen when both the merchant and the cardholder are affected by the fraud. In this case, the cardholder receives a refund from the credit institute (activity *refund credit institute*) and the case is closed.

Note that for simplicity, we have used single letters to represent the activity labels in the Petri net transitions. Some possible traces in this process are for example: $\langle h, c, r, m, u \rangle$, $\langle b, c, r, m, f \rangle$, $\langle h, c, r, i, f \rangle$ and $\langle b, c, r, i, t, v \rangle$.

Suppose that the credit card company wants to perform conformance checking to identify deviant process instances. However, some traces in the information system of the company are affected by uncertainty, such as the one in Table 1.

Suppose that in the first half of October 2020, the company was implementing a new system for automatic event data generation. During this time, the event data regarding the credit card fraud investigation process often had to be inserted manually by the employees. Such manual recordings were subject to inaccuracies, leading to imprecise or missing data affecting the cases during

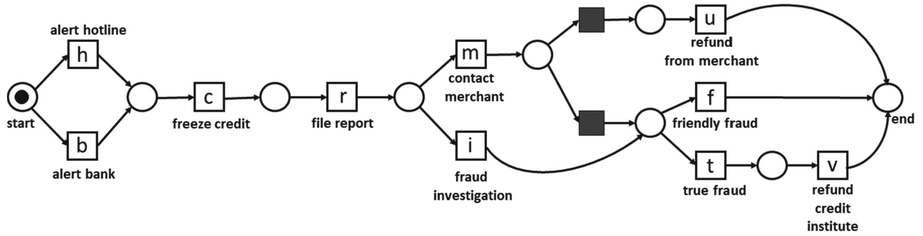


Fig. 1. A Petri net model of the credit card fraud investigation process. This net allows for 10 possible traces.

Table 1. Example of an uncertain case from the credit card fraud investigation process.

Case ID	Event ID	Activity	Timestamp	Ind.
5167	e_1	h (alert hotline)	05-10-2020 23:00	
5167	e_2	c (freeze credit)	06-10-2020	
5167	e_3	r (file report)	U(05-10-2020 20:00, 06-10-2020 10:00)	
5167	e_4	i (fraud investigation)	09-10-2020 10:00	
5167	e_5	$\{f : 0.3$ (friendly fraud), $t : 0.7$ (true fraud)$\}$	14-10-2020 09:00	
5167	e_6	v (refund credit institute)	15-10-2020 10:00	?

this period. The process instance from Table 1 is one of the affected instances. Here, events e_2, e_3, e_5, e_6 are uncertain. The timestamp of event e_2 is not precise enough, so the possible timestamp lies between 06-10-2020 00:00 and 06-10-2020 23:59. Event e_3 has happened some time between 20:00 on October 5th and 10:00 on October 6th. Event e_5 has two possible activity labels: f with probability 0.3 and t with probability 0.7. Refunding the customer (event e_6) has been recorded in the system, but the customer has not received the money yet, which is why the event is indeterminate: this is indicated with a question mark (?) in the rightmost column, and indicates an event that has been recorded, but for which is unclear if it actually occurred in reality.

The credit card company is interested in understanding if and how the data in this uncertain trace conforms with the normative process model, and the entity of the actual compliance risk; they are specifically interested in knowing whether a severely non-compliant scenario is highly likely. In the remainder of the paper, we will describe a method able to estimate the probability of all possible outcome scenarios.

4 Preliminaries

Let us now present some preliminary definitions regarding uncertain event data.

Definition 1 (Uncertain attributes). *Let* $\mathbb{U}$ *be the* universe of attribute domains*, and the set* $\mathcal{D} \in \mathbb{U}$ *be an* attribute domain*. Any* $\mathcal{D} \in \mathbb{U}$ *is a discrete set or a totally ordered set. A* strongly uncertain attribute *of domain* $\mathcal{D}$ *is a subset* $d_S \subseteq \mathcal{D}$ *if* $\mathcal{D}$ *is a discrete set, and it is a closed interval* $d_S = [d_{min}, d_{max}]$ *with* $d_{min} \in \mathcal{D}$ *and* $d_{max} \in \mathcal{D}$ *otherwise. We denote with* $S_\mathcal{D}$ *the set of all such strongly uncertain attributes of domain* $\mathcal{D}$*. A* weakly uncertain attribute $f_\mathcal{D}$ *of domain* $\mathcal{D}$ *is a function* $f_\mathcal{D} \colon \mathcal{D} \not\to [0,1]$ *such that* $0 < \sum_{x \in \mathcal{D}} f_\mathcal{D}(x) \leq 1$ *if* $\mathcal{D}$ *is finite,* $0 < \int_{-\infty}^{\infty} f_\mathcal{D}(x)\, dx \leq 1$ *otherwise. We denote with* $W_\mathcal{D}$ *the set of all such weakly uncertain attributes of domain* $\mathcal{D}$*. We collectively denote with* $\mathcal{U}_\mathcal{D} = S_\mathcal{D} \cup W_\mathcal{D}$ *the set of* uncertain attributes *of domain* $\mathcal{D}$*.*

It is easy to see how a "certain" attribute x, with a value not affected by any uncertainty, can be represented through the definitions in use here: if its domain is discrete, it can be represented with the singleton $\{x\}$; otherwise, it can be represented with the degenerate interval $[x, x]$.

Definition 2 (Uncertain events). *Let* $\mathbb{U}_I$ *be the* universe of event identifiers*. Let* $\mathbb{U}_C$ *be the* universe of case identifiers*. Let* $A \in \mathbb{U}$ *be the discrete domain of all the* activity identifiers*. Let* $T \in \mathbb{U}$ *be the totally ordered domain of all the* timestamp identifiers*. Let* $O = \{?\} \in \mathbb{U}$*, where the "?" symbol is a placeholder denoting* event indeterminacy*. The* universe of uncertain events *is denoted with* $\mathcal{E} = \mathbb{U}_I \times \mathbb{U}_C \times \mathcal{U}_A \times \mathcal{U}_T \times \mathcal{U}_O$*.*

The activity label, timestamp and indeterminacy attribute values of an uncertain event are drawn from $\mathcal{U}_A$, $\mathcal{U}_T$ and $\mathcal{U}_O$; in accordance with Definition 1, each of these attributes can be strongly uncertain (set of possible values or interval) or weakly uncertain (probability distribution). The indeterminacy domain is defined on a single element "?": thus, strongly uncertain indeterminacy may be $\{?\}$ (indeterminate event) or $\varnothing$ (no indeterminacy). In weakly uncertain indeterminacy, the "?" element is associated to a probability value.

Definition 3 (Projection functions). *For an uncertain event* $e = (i, c, a, t, o) \in \mathcal{E}$*, we define the following projection functions:* $\pi_a(e) = a$*,* $\pi_t(e) = t$*,* $\pi_o(e) = o$*. We define* $\pi_a^{set}(e) = a$ *if* a *is strongly uncertain, and* $\pi_a^{set}(e) = \{x \in \mathcal{U}_A \mid f_A(x) > 0\}$ *with* $a = f_A$ *otherwise. If the timestamp* $t = [t_{min}, t_{max}]$ *is strongly uncertain, we define* $\pi_{t_{min}}(e) = t_{min}$ *and* $\pi_{t_{max}}(e) = t_{max}$*. If the timestamp* $t = f_T$ *is weakly uncertain, we define* $\pi_{t_{min}}(e) = \operatorname{argmin}_x(f_T(x) > 0)$ *and* $\pi_{t_{max}}(e) = \operatorname{argmax}_x(f_T(x) > 0)$*.*

Definition 4 (Uncertain traces and logs). $\tau \subset \mathcal{E}$ *is an* uncertain trace *if all the event identifiers in* τ *are unique and all events in* τ *share the same case identifier* $c \in \mathbb{U}_C$*.* $\mathcal{T}$ *denotes the universe of uncertain traces.* $L \subset \mathcal{T}$ *is an* uncertain log *if all the event identifiers in* L *are unique.*

Definition 5 (Realizations of uncertain traces). *Let* $e, e' \in \mathcal{E}$ *be two uncertain events.* $\prec_\mathcal{E}$ *is a strict partial order defined on the universe of strongly uncertain events* $\mathcal{E}$ *as* $e \prec_\mathcal{E} e' \Leftrightarrow \pi_{t_{max}}(e) < \pi_{t_{min}}(e')$*. Let* $\tau \in \mathcal{T}$ *be an uncertain trace. The sequence* $\rho = \langle e_1, e_2, \ldots, e_n \rangle \in \mathcal{E}^*$*, with* $n \leq |\tau|$*, is an* order-realization *of* τ *if there exists a total function* $f \colon \{1, 2, \ldots, n\} \to \tau$ *such that:*

- *for all* $1 \leq i < j \leq n$ *we have that* $\rho[j] \not<_{\mathcal{E}} \rho[i]$*,*
- *for all* $e \in \tau$ *with* $\pi_o(e) = \varnothing$ *there exists* $1 \leq i \leq n$ *such that* $f(i) = e$*.*

We denote with $\mathcal{R}_O(\tau)$ *the set of all such order-realizations of the trace* τ*.*

Given an order-realization $\rho = \langle e_1, e_2, \ldots, e_n \rangle \in \mathcal{R}_O(\tau)$*, the sequence* $\sigma \in \mathcal{U}_A{}^*$ *is a* realization *of* ρ *if* $\sigma \in \{\langle a_1, a_2, \ldots, a_n \rangle \mid \forall_{1 \leq i \leq n}\, a_i \in \pi_a^{set}(i)\}$*. We denote with* $\mathcal{R}_A(\rho) \subseteq \mathcal{U}_A{}^*$ *the set of all such realizations of the order-realization* ρ*. We denote with* $\mathcal{R}(\tau) \subseteq \mathcal{U}_A{}^*$ *the union of the realizations obtainable from all the order-realizations of* τ*:* $\mathcal{R}(\tau) = \bigcup_{\rho \in \mathcal{R}_O(\tau)} \mathcal{R}_A(\rho)$*. We will say that an order-realization* $\rho \in \mathcal{R}_O(\tau)$ enables *a sequence* $\sigma \in \mathcal{U}_A{}^*$ *if* $\sigma \in \mathcal{R}_A(\rho)$*.*

Detailing an algorithm to generate all realizations of an uncertain trace is beyond the scope of this paper. The literature illustrates a conformance checking method over uncertain data which employs a *behavior net*, a Petri net able to replay all and only the realizations of an uncertain trace [8]. Exhaustively exploring all complete firing sequences of a behavior net, e.g., through its reachability graph, provides all realizations of the corresponding uncertain trace.

Given the above formalization, we can now define more clearly the research question that we are investigating in this paper. Given an uncertain trace $\tau \in \mathcal{T}$ and one of its realizations $\sigma \in \mathcal{R}(\tau)$, our goal is to obtain a procedure to reliably compute $P(\sigma \mid \tau)$ = "*probability of* σ *given that we observe* τ". In other words, provided that σ corresponds to a scenario (i.e., a realization) for the uncertain trace τ, we are interested in calculating the probability that σ is the actual scenario occurred in reality, which caused the recording of the uncertain trace τ in the event log. In the next section, we will illustrate how to calculate such probabilities of uncertain traces realizations.

5 Method

Before we show how we can obtain probability estimates for all realizations of an uncertain trace, it is important to state an assumption: the information on uncertainty related to a particular attribute in some event is independent of the possible values of the same attribute present in other events, and it is independent of the uncertainty information on other attributes of the same event. Note that in the examples of uncertainty sources given in Sect. 1 (data coarseness and sensor errors), this independence assumption often holds.

Additionally, we need to consider the fact that strongly uncertain attributes do not come with known probability values: their description only specifies the values that attributes might acquire, but not the likelihood of each possible value. As a consequence, estimating probability for specific realizations in a strongly uncertain environment is only possible with a-priori assumptions on how probability distributes among the attribute value. At times, it might be possible to assume the distribution in an informed way—for instance, on the basis of features of the information system hosting the data, of the sensors recording events and attributes, or other tools involved in the management of the process.

In case no indication is present, a reasonable assumption—which we will hold for the remainder of the paper—is that any possible value of a strongly uncertain attribute is equally likely. Formally, with $e = (i, c, a, t, o) \in \mathcal{E}$ let $\tau_s \colon \mathcal{E} \to \mathcal{E}$ be a function such that $\tau_s(e) = (i, c, a', t', o')$, where $a' = \{(x, \frac{1}{|\pi_a^{set}(e)|}) \mid x \in \pi_a^{set}(e)\}$ if $a \in S_A$ and $a' = a$ otherwise; $t' = U(\pi_{t_{min}}(e), \pi_{t_{max}}(e))$ if $t \in S_T$ and $t' = t$ otherwise; $o' = 0.5$ if $o = \{?\}$ and $o' = o$ otherwise.

First, observe that the probability $P(\sigma \mid \tau)$ that an activity sequence $\sigma \in {\mathcal{U}_A}^*$ is indeed a realization of the trace $\tau \in \mathcal{T}$, and thus $\sigma \in \mathcal{R}(\tau)$, increases with the number of order-realizations enabling it. Furthermore, for each such order-realizations, one can construct a probability function $P_O(\rho \mid \tau)$ reflecting the likelihood of the sequence ρ itself given the trace τ, and a probability function $P_A(\sigma \mid \rho)$ reflecting the likelihood that the realization corresponding to ρ is indeed σ. The value of $P_O(\rho \mid \tau)$ is affected by the uncertainty information in timestamps and indeterminate events, while the value of $P_A(\sigma \mid \rho)$ is aggregated from the uncertainty information in the activity labels.

Given a realization σ of an uncertain process instance and the set of its enablers, its probability is computed as following:

$$P(\sigma \mid \tau) = \sum_{\rho \in \mathcal{E}*} P_O(\rho \mid \tau) \cdot P_A(\sigma \mid \rho)$$

Note that, if ρ does not enable σ, $P_A(\sigma \mid \rho) = 0$. For any uncertain trace $\tau \in \mathcal{T}$, it holds that $\sum_{\sigma \in \mathcal{R}(\tau)} P(\sigma \mid \tau) = 1$, since both $P_O(\cdot)$ and $P_A(\cdot)$ are each constructed to be (independent) probability distributions.

We will now compute $P_A(\sigma \mid \rho)$ using the information on the activity labels uncertainty. Let us write f_A^e as a shorthand for $\pi_a(e)$. If there is uncertainty in activities, then for each event $e \in \rho$ and activity label $a \in \pi_a^{set}(e)$, the probability that e executes a is given by $f_A^e(a)$. Thus, for every $\rho = \langle e_1, ..., e_n \rangle \in \mathcal{R}_O(\tau)$ and $\sigma = \langle a_1, ..., a_n \rangle \in \mathcal{R}_O(\tau)$, the value P_A can be aggregated from these distributions in the following way:

$$P_A(\sigma \mid \rho) = \prod_{i=1}^{n} f_A^i(a_i)$$

Through the value of P_A, we can assess the likelihood that any given order-realization executes a particular realization. The next step is to estimate the probability of each order-realization ρ from the set $\mathcal{R}_O(\tau)$. The probability of observing ρ needs to be aggregated from the probability that the corresponding set of events appears in the given particular order, which is determined by the timestamp intervals and, if applicable, the distributions over them; and the probability that the order-realization contains the corresponding specific set of events, which is determined by the uncertainty information on the indeterminacy. Multiplying the two values obtained above to yield a probability estimate for the order-realization reflects our independence assumption. Let us firstly focus on uncertainty on timestamps, which causes the events to be partially ordered.

We will write $f_T^e(t)$ as a shorthand for $\pi_t(e)(t)$. For every event e, the value of $f_T^e(t)$ yields the probability that event e happened on timestamp t. This value is always 0 for all $t < \pi_{t_{min}}(e)$ and $t > \pi_{t_{max}}(e)$ (see $\pi_{t_{min}}$ and $\pi_{t_{max}}$ in Definition 3). Given the continuous domain of timestamps, $P_O(\cdot)$ is assessed by using integrals. For a trace $\tau \in \mathcal{T}$ and an order-realization $\rho = \langle e_1, ..., e_n \rangle \in \mathcal{R}_O(\tau)$, let $a_i = \pi_{t_{min}}(i)$ and $b_i = \pi_{t_{max}}(i)$ for all $1 \leq i \leq n$. Then, we define:

$$\begin{aligned} I(\rho) = &\int_{a_1}^{min\{b_1,...,b_n\}} f_T^{e_1}(x_1) \int_{max\{a_2,x_1\}}^{min\{b_2,...,b_n\}} f_T^{e_2}(x_2) \cdots \\ &\int_{max\{a_i,x_{i-1}\}}^{min\{b_i,...,b_n\}} f_T^{i}(x_i) \cdots \int_{max\{a_n,x_{n-1}\}}^{b_n} f_T^{e_n}(x_n)\, d_{x_n} \dots d_{x_1} \\ = &\int_{a_1}^{min\{b_1,...,b_n\}} \int_{max\{a_2,x_1\}}^{min\{b_2,...,b_n\}} \cdots \int_{max\{a_i,x_{i-1}\}}^{min\{b_i,...,b_n\}} \cdots \\ &\int_{max\{a_n,x_{n-1}\}}^{b_n} \prod_{i=1}^{n} f_T^{i}(x_i)\, d_{x_n} \dots d_{x_1} \end{aligned}$$

This chain of integrals allows us to compute the probability of a specific order among all the events in an uncertain trace. Now, to compute the probability of each realization from $\mathcal{R}_e$ accounting for indeterminate events, we combine both the probability of the events having appeared in a particular order and the probability that the sequence contains exactly those events. For simplicity, we will use a function that acquires the value 1 if an event is not indeterminate. Let us define $f_O^e \colon O \rightarrow [0,1]$ such that $f_O^e(?) = \pi_o(e)(?)$ if $\pi_o(e) \neq \varnothing$ and $f_O^e(?) = 1$ otherwise. More precisely, given $\tau \in \mathcal{T}$ and $\rho \in \mathcal{R}_O(\tau)$, we compute:

$$P_O(\rho \mid \tau) = I(\rho) \cdot \prod_{\substack{e \in \tau \\ e \in \rho}} (1 - f_O^e(?)) \cdot \prod_{\substack{e \in \tau \\ e \notin \rho}} f_O^e(?)$$

We now have at our disposal all the necessary tools to compute a probability distribution over the trace realizations of any uncertain process instance in any possible uncertainty scenario. Let us then apply this method to compute the probabilities of all realizations of the trace τ in Table 1, and to analyze its conformance to the process in Fig. 1.

Each order-realization of τ enables two realizations, because event e_5 has two possible activity labels. Since for events $e \in \tau \setminus \{e_5\}$, we have f_A^e equal to 1 for their corresponding unique activity label, the probability that an order-realization $\rho \in \mathcal{R}_O(\tau)$ has some realization $\sigma \in \mathcal{R}_A(\rho)$ only depends on whether the trace σ contains activity f or t. Thus, for traces $\sigma^{1'}, \sigma^{2'}, \sigma^{3'}, \sigma^{4'}, \sigma^{5'}, \sigma^{6'}$ and their unique enabling sequences, we always have $P_A(\sigma^{i'} \mid s_e^i) = f_A^{e_5}(f) = 0.3$, where $i \in \{1, \dots, 6\}$. Similarly, for traces $\sigma^{1''}, \sigma^{2''}, \sigma^{3''}, \sigma^{4''}, \sigma^{5''}, \sigma^{6''}$ and their unique enabling sequences, we always have $P_A(\sigma^{i''} \mid \rho^i) = f_A^{e_5}(t) = 0.7$, where $i \in \{1, \dots, 6\}$. Next, we calculate the $P_O(\cdot)$ values for the 6 possible order-realizations in $\mathcal{R}_O(\tau)$, which are displayed in Table 2.

One can notice that the I values only depend on the ordering of the first three events, which are also the only ones with overlapping timestamps. Since the indeterminate event e_6 does not overlap with any other event, pairs of sequences where the first three events have the same order also have the same probability. This reflects our assumption that the occurrence and non-occurrence of e_6 are

Table 2. The possible order-realizations of the process instance from Table 1 and their probabilities.

Order-realization ρ	$I(\rho)$	$P_O(\rho)$
ρ^1:$\langle e_1, e_2, e_3, e_4, e_5, e_6\rangle$	0.140	0.074
ρ^2:$\langle e_1, e_3, e_2, e_4, e_5, e_6\rangle$	0.780	0.390
ρ^3:$\langle e_3, e_1, e_2, e_4, e_5, e_6\rangle$	0.072	0.036
ρ^4:$\langle e_1, e_2, e_3, e_4, e_5\rangle$	0.149	0.074
ρ^5:$\langle e_1, e_3, e_2, e_4, e_5\rangle$	0.780	0.390
ρ^6:$\langle e_3, e_1, e_2, e_4, e_5\rangle$	0.072	0.036

Table 3. The set of possible realizations of the example from Table 1, their enablers, their probabilities, and their conformance scores. The conformance score is equal to the cost of the optimal alignment between the trace and the Petri net in Fig. 1.

Realization σ	ρ	$P(\sigma \mid \tau)$	$conf$
$\sigma^{1'}$:$\langle h, c, r, i, f, v\rangle$	ρ^1	$P_O(\rho^1){\cdot}P_A(\sigma^{1'} \vert \rho^1) = 0.022$	1
$\sigma^{1''}$:$\langle h, c, r, i, t, v\rangle$	ρ^1	$P_O(\rho^1){\cdot}P_A(\sigma^{1''} \vert \rho^1) = 0.052$	0
$\sigma^{2'}$:$\langle h, r, c, i, f, v\rangle$	ρ^2	$P_O(\rho^2){\cdot}P_A(\sigma^{2'} \vert \rho^2) = 0.117$	3
$\sigma^{2''}$:$\langle h, r, c, i, t, v\rangle$	ρ^2	$P_O(\rho^2){\cdot}P_A(\sigma^{2''} \vert \rho^2) = 0.273$	2
$\sigma^{3'}$:$\langle r, h, c, i, f, v\rangle$	ρ^3	$P_O(\rho^3){\cdot}P_A(\sigma^{3'} \vert \rho^3) = 0.011$	3
$\sigma^{3''}$:$\langle r, h, c, i, t, v\rangle$	ρ^3	$P_O(\rho^3){\cdot}P_A(\sigma^{3''} \vert \rho^3) = 0.025$	2
$\sigma^{4'}$:$\langle h, c, r, i, f\rangle$	ρ^4	$P_O(\rho^4){\cdot}P_A(\sigma^{4'} \vert \rho^4) = 0.022$	0
$\sigma^{4''}$:$\langle h, c, r, i, t\rangle$	ρ^4	$P_O(\rho^4){\cdot}P_A(\sigma^{4''} \vert \rho^4) = 0.052$	1
$\sigma^{5'}$:$\langle h, r, c, i, f\rangle$	ρ^5	$P_O(\rho^5){\cdot}P_A(\sigma^{5'} \vert \rho^5) = 0.117$	2
$\sigma^{5''}$:$\langle h, r, c, i, t\rangle$	ρ^5	$P_O(\rho^5){\cdot}P_A(\sigma^{5''} \vert \rho^5) = 0.273$	3
$\sigma^{6'}$:$\langle r, h, c, i, f\rangle$	ρ^6	$P_O(\rho^6){\cdot}P_A(\sigma^{6'} \vert \rho^6) = 0.011$	2
$\sigma^{6''}$:$\langle r, h, c, i, t\rangle$	ρ^6	$P_O(\rho^6){\cdot}P_A(\sigma^{6''} \vert \rho^6) = 0.025$	3

both equally possible. Table 3 displays the calculations for the computation of the $P(\sigma \mid \tau)$ values for all realizations. Now we can compute the expected conformance score for the uncertain process instance $\tau = \{e_1, \ldots, e_6\}$. We can do so by computing alignments [5] for each realization of τ:

$$\begin{aligned}\overline{conf}(\tau) = \sum_{\sigma \in \mathcal{R}(\tau)} P(\sigma \mid \tau) \cdot conf(\sigma, M) &= 0.022 \cdot 1 + 0.05 \cdot 0 + 0.117 \cdot 3 + 0.273 \cdot 2 + 0.011 \cdot 3 \\ &+ 0.025 \cdot 2 + 0.022 \cdot 0 + 0.052 \cdot 1 + 0.117 \cdot 2 + 0.273 \cdot 3 + 0.011 \cdot 2 + 0.025 \cdot 3 \\ &= 2.204.\end{aligned}$$

Given the information on uncertainty available for the trace, this conformance score is a more realistic estimate of the real conformance score compared to taking the best, worst or average scores with values 0, 3 and 1.75 respectively.

6 Validation of Probability Estimates

In this section, we compute the probability estimates for the realizations of an uncertain trace, and then show a validation of those estimates by Monte Carlo simulation on the behavior net of the trace. The process instance of our example has strong uncertainty in timestamps and weak uncertainty in activities and indeterminacy. It consists of 4 events: e_1, e_2, e_3 and e_4, where e_2 and e_3 have overlapping timestamps. Event e_2 executes b (resp., c) with probability 0.9 (resp., 0.1). There is a probability of 0.2 that e_3 did not occur. Figure 2 shows the corresponding behavior graph, an uncertain event data visualization that represents the time relationships between events with a directed acyclic

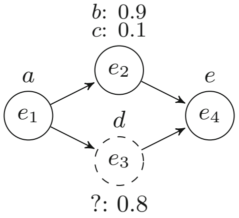


Fig. 2. The behavior graph of the uncertain trace considered as example for validation.

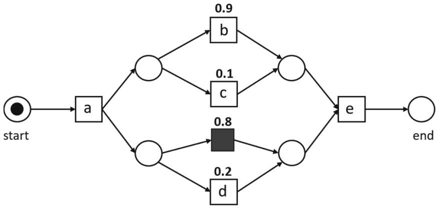


Fig. 3. The behavior net obtained from the behavior graph in Fig. 2.

Table 4. The set of realizations of the trace from Fig. 2, their enablers, and their probabilities.

Realization σ	ρ	$P(\sigma\|\tau)$
σ^1:$\langle a, b, e\rangle$	ρ^1:$\langle e_1, e_2, e_4\rangle$	$P_O(\rho^1)\cdot P_A(\sigma^1\|\rho^1) = 0.8\cdot 0.9 = 0.72$
σ^2:$\langle a, b, d, e\rangle$	ρ^2:$\langle e_1, e_2, e_3, e_4\rangle$	$P_O(\rho^2)\cdot P_A(\sigma^2\|\rho^2) = (0.5\cdot 0.2)\cdot 0.9 = 0.09$
σ^3:$\langle a, d, b, e\rangle$	ρ^3:$\langle e_1, e_3, e_2, e_4\rangle$	$P_O(\rho^3)\cdot P_A(\sigma^3\|\rho^3) = (0.5\cdot 0.2)\cdot 0.9 = 0.09$
σ^4:$\langle a, c, e\rangle$	ρ^4:$\langle e_1, e_2, e_4\rangle$	$P_O(\rho^4)\cdot P_A(\sigma^4\|\rho^4) = 0.8\cdot 0.1 = 0.08$
σ^5:$\langle a, c, d, e\rangle$	ρ^5:$\langle e_1, e_2, e_3, e_4\rangle$	$P_O(\rho^5)\cdot P_A(\sigma^5\|\rho^5) = (0.5\cdot 0.2)\cdot 0.1 = 0.01$
σ^6:$\langle a, d, c, e\rangle$	ρ^6:$\langle e_1, e_3, e_2, e_4\rangle$	$P_O(\rho^6)\cdot P_A(\sigma^6\|\rho^6) = (0.5\cdot 0.2)\cdot 0.1 = 0.01$

graph [8]. Lastly, Table 4 list all the possible realizations, their probabilities, and the order-realizations enabling them.

We now validate our obtained probability estimates quantitatively by means of a Monte Carlo simulation approach. First, we construct the behavior net [10] corresponding to the uncertain process instance, which is shown in Fig. 3. The set of replayable traces in this behavior net is exactly the set of realizations for the uncertain instance. Then, we simulate realizations on the behavior net, dividing the accumulated count of each realization by the number of runs, and compare those values to our probability estimates. Here, we use the *stochastic simulator* of the PM4Py library [3]. In every step of the simulation, the stochastic simulator chooses one enabled transition to fire according to a stochastic map, assigning a weight to each transition in the Petri net (here, the behavior net).

To simulate uncertainty in activities, events and timestamps, we do the following: possible activities executed by the same event appearing in an XOR-split in the behavior net are weighted so to reflect the probability values of the activity labels. Indeterminacy is equivalently modeled as an XOR-choice between a visible transition and a silent one in the behavior net, so to model a "skip". If there are two or more possible activities for an indeterminate event, then the sum of the weights of the visible transitions in relation to the weight of the silent transition should be the same as in the distribution given in the event

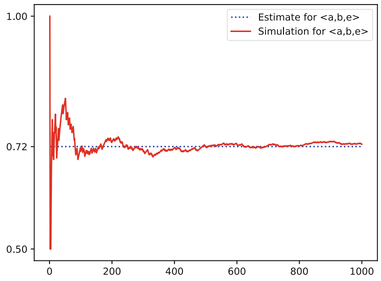


Fig. 4. Plot showing how the frequency of trace $\langle a, b, e\rangle$ converges to the expected value of 0.72 over 1000 runs.

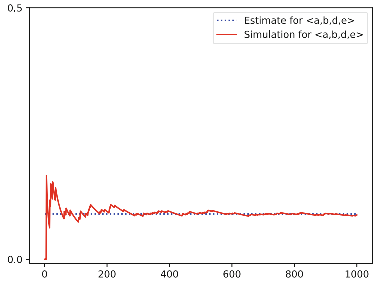


Fig. 5. Plot showing how the frequency of trace $\langle a, b, d, e\rangle$ converges to the expected value of 0.09 over 1000 runs.

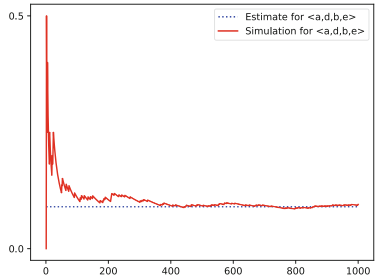


Fig. 6. Plot showing how the frequency of trace $\langle a, d, b, e\rangle$ converges to the expected value of 0.09 over 1000 runs.

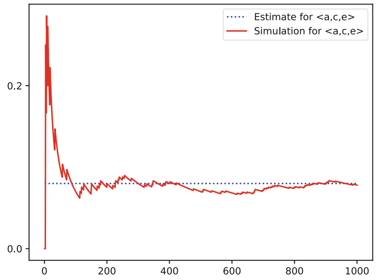


Fig. 7. Plot showing how the frequency of trace $\langle a, c, e\rangle$ converges to the expected value of 0.08 over 1000 runs.

type uncertainty information. Whenever there are events with overlapping timestamps, these appear in an AND-split in the behavior net. The (enabled) path of the AND-split which is taken first signals which event is executed at that moment.

Let $bn(\tau) = (P, T)$ be the behavior net of trace τ. Let $(e, a) \in T$ be a visible transition related to some event $e \in \tau$. We weight (e, a) the following way:

$$\mathit{weight}((e,a)) = \begin{cases} f_A^e(a) & \text{if } \pi_o(e) = \varnothing, \\ (1 - f_O^e(?)) \cdot f_A^e(a) & \text{otherwise.} \end{cases}$$

If $e \in \tau$ is an indeterminate event, then $\mathit{weight}((e, \epsilon)) = f_O^e(?)$.

Note that according to the weight assignment function, if e is determinate, then $\sum_{a \in \pi_a^{set}(e)} \mathit{weight}((e, a)) = 1$. Otherwise, $\sum_{a \in \pi_a^{set}(e)} \mathit{weight}((e, a)) = 1 - f_O^e(?) = 1 - \mathit{weight}((e, \tau))$. By construction of the behavior net, any transition related to an event in τ can only fire in accordance with the partial order of uncertain timestamps. Additionally, all transitions representing events with overlapping timestamps appear in an AND construct. By definition of our weight function, whenever the transitions of some $e \in \tau$ are enabled (in an XOR construct), the probability of firing one of them is $1/k$, where k is the number of events from τ for which none of the corresponding transitions have fired yet. This way, there is always a uniform distribution over the set of enabled transitions representing overlapping events. Assigning the weights according to this distribution allows to decorate the behavior net with probabilities that reflect the chances of occurrence of every possible value in uncertain attributes.

Applying the stochastic simulator n times yields n realizations. For each of the 6 possible realizations for the uncertain process instance, we obtain a probability measurement by dividing its simulated frequency by n. Figures 4 through 7 show how for greater n, this measurement converges to the probability estimates shown in Table 4, which were computed with our method.

To conclude, the Monte Carlo simulation shows that our estimated probabilities for realizations match their relative frequencies when one simulates the behavior net of the corresponding uncertain trace.

7 Conclusion

Uncertain traces inherently contain behavior, allowing for many realizations; these, in turn, correspond to diverse possible real-life scenarios, that may have different consequences on the management and governance of a process. In this paper, we presented a method to quantify the probability of each realization of an uncertain trace. This enables process analysts to weigh the impact of specific insights gathered with uncertainty-aware process mining techniques, such as conformance checking using alignments. As a consequence, information from process analysis techniques can be associated with a quantification of risk or opportunity for specific scenarios, making them more trustworthy.

Multiple avenues for future work on this topic are possible. These include inferring probabilities for uncertain traces from sections of the log not affected by uncertainty, adopting certain traces or fragments of traces as ground truth. Moreover, inferring probabilities by examining evidence against a ground truth can also be achieved with a normative model that includes information concerning the probability of error or noise in specific parts of the process.

References

1. van der Aa, H., Leopold, H., Weidlich, M.: Partial order resolution of event logs for process conformance checking. Decis. Support Syst. **136**, 113347 (2020). https://doi.org/10.48550/arXiv.2007.02416
2. Ao, X., Luo, P., Li, C., Zhuang, F., He, Q.: Online frequent episode mining. In: 2015 IEEE 31st International Conference on Data Engineering. IEEE (2015). https://doi.org/10.1109/ICDE.2015.7113342
3. Berti, A., van Zelst, S.J., van der Aalst, W.M.P.: Process mining for Python (PM4Py): bridging the gap between process- and data science. In: ICPM Demo Track (CEUR 2374) (2019)

4. Busany, N., van der Aa, H., Senderovich, A., Gal, A., Weidlich, M.: Interval-based queries over lossy IoT event streams. Trans. Data Sci. **1**(4), 1–27 (2020). https://doi.org/10.1145/3385191
5. van Dongen, B., Carmona, J., Chatain, T., Taymouri, F.: Aligning modeled and observed behavior: a compromise between computation complexity and quality. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 94–109. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_7
6. Leemans, M., van der Aalst, W.M.P.: Discovery of frequent episodes in event logs. In: Ceravolo, P., Russo, B., Accorsi, R. (eds.) SIMPDA 2014. LNBIP, vol. 237, pp. 1–31. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-27243-6_1
7. Lu, X., Fahland, D., van der Aalst, W.M.P.: Conformance checking based on partially ordered event data. In: Fournier, F., Mendling, J. (eds.) BPM 2014. LNBIP, vol. 202, pp. 75–88. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15895-2_7
8. Pegoraro, M., van der Aalst, W.M.P.: Mining uncertain event data in process mining. In: International Conference on Process Mining (ICPM). IEEE (2019). https://doi.org/10.1109/ICPM.2019.00023
9. Pegoraro, M., Uysal, M.S., van der Aalst, W.M.P.: Discovering process models from uncertain event data. In: Di Francescomarino, C., Dijkman, R., Zdun, U. (eds.) BPM 2019. LNBIP, vol. 362, pp. 238–249. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-37453-2_20
10. Pegoraro, M., Uysal, M.S., van der Aalst, W.M.P.: Conformance checking over uncertain event data. Inf. Syst. **102**, 101810 (2021). https://doi.org/10.1016/j.is.2021.101810
11. Zhu, H., Wang, P., He, X., Li, Y., Wang, W., Shi, B.: Efficient episode mining with minimal and non-overlapping occurrences. In: 2010 IEEE International Conference on Data Mining. IEEE (2010). https://doi.org/10.1109/ICDM.2010.25

Visualizing Trace Variants from Partially Ordered Event Data

Daniel Schuster[1,2](✉), Lukas Schade[2], Sebastiaan J. van Zelst[1,2], and Wil M. P. van der Aalst[1,2]

[1] Fraunhofer Institute for Applied Information Technology FIT, Sankt Augustin, Germany
{daniel.schuster,sebastiaan.van.zelst}@fit.fraunhofer.de
[2] RWTH Aachen University, Aachen, Germany
wvdaalst@pads.rwth-aachen.de

Abstract. Executing operational processes generates event data, which contain information on the executed process activities. Process mining techniques allow to systematically analyze event data to gain insights that are then used to optimize processes. Visual analytics for event data are essential for the application of process mining. Visualizing unique process executions—also called trace variants, i.e., unique sequences of executed process activities—is a common technique implemented in many scientific and industrial process mining applications. Most existing visualizations assume a total order on the executed process activities, i.e., these techniques assume that process activities are atomic and were executed at a specific point in time. In reality, however, the executions of activities are *not* atomic. Multiple timestamps are recorded for an executed process activity, e.g., a start-timestamp and a complete-timestamp. Therefore, the execution of process activities may overlap and, thus, cannot be represented as a total order if more than one timestamp is to be considered. In this paper, we present a visualization approach for trace variants that incorporates start- and complete-timestamps of activities.

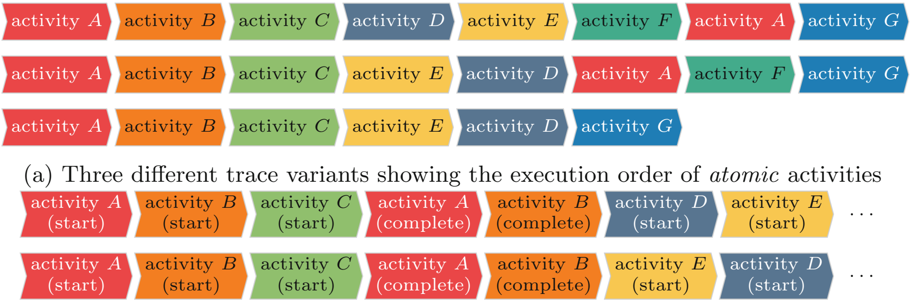
Keywords: Process Mining · Visual analytics · Interval order

1 Introduction

The execution of operational processes, e.g., business and production processes, is often supported by information systems that record process executions in detail. We refer to such recorded information as *event data*. The analysis of event data is of great importance for organizations to improve their processes. *Process mining* [1] offers various techniques for systematically analyzing event data, e.g., to learn a process model, to check compliance, and to obtain performance measures. These insights into the processes can then be used to optimize them.

As in other data analysis applications, *visual analytics* for event data are important in the application of process mining. A state-of-the-art process mining methodology [6] lists *process analytics* including visual analytics as a key

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 34–46, 2022.
https://doi.org/10.1007/978-3-030-98581-3_3



(a) Three different trace variants showing the execution order of *atomic* activities

(b) Two different trace variants showing the execution order of *non-atomic* activities, i.e, each activity is split into start and complete

Fig. 1. Classic trace variant visualizations for *(non)-atomic* process activities

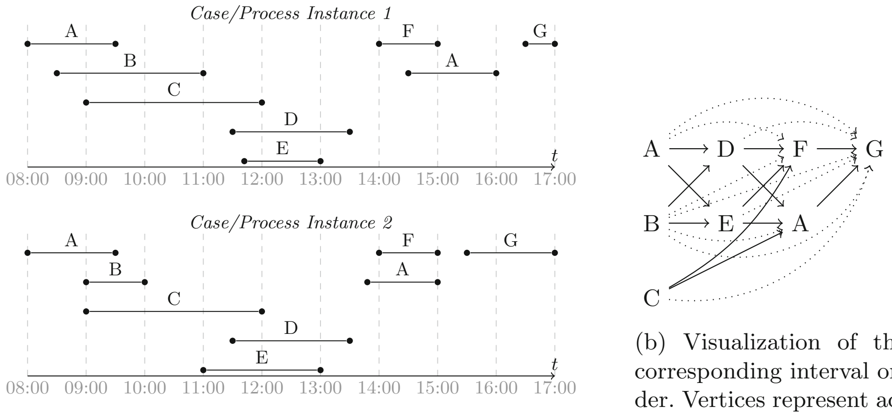
component next to the classic fields of process mining: process discovery, conformance checking, and process enhancement.

A visualization approach that is used across various process mining tools, ranging from industry to scientific tools, is called the *variant explorer*. Consider Fig. 1a for an example. In classic trace variant visualizations, a *variant* describes a unique sequence of executed process activities. Thus, a *strict total order* on the contained activities is required to visualize such sequence. Recorded timestamps of the executed activities are usually used for ordering them.

This classic trace variant visualization has two main limitations. (1) Assume atomic process activities, i.e., a single timestamp is recorded for each process activity. A *strict* total order cannot be derived if multiple activities have the same timestamp. In such cases, the sequential visualization, indicating temporal execution, of process activities is problematic because a second-order criteria is needed to obtain a strict total order. (2) In many real-life scenarios, process activities are performed over time, i.e., they are *non*-atomic. Thus, the execution of activities may intersect with each other. Consider Fig. 2a for an example. Considering both start- and complete-timestamps, a strict total order cannot be obtained if the executions of activities overlap. The classic trace variant explorer usually splits the activities in start and complete as shown in Fig. 1b to obtain atomic activities. However, the parallel behavior of activities is not easily discernible from the visualization. In addition, the first limitation remains.

In this paper, we propose a novel visualization of trace variants to overcome the two aforementioned limitations. We define a variant as an *interval order*, which can be represented as a graph. For instance, Fig. 2b shows the interval order of the two process executions shown in Fig. 2a. The graph representation of an interval order (cf. Fig. 2b) is, however, not easy to read compared to the classic trace variant explorer (cf. Fig. 1). Therefore, we propose an approach to derive a visualization from interval orders representing trace variants.

The remainder of this paper is structured as follows. Section 2 presents related work. Section 3 introduces concepts and definitions used throughout this paper.



(a) Time plots visualizing activity instances, i.e., each activity has a start-timestamp and a complete-timestamp, executed within two different cases/process instance

(b) Visualization of the corresponding interval order. Vertices represent activity instances. Arcs indicate an ordering between activity instances

Fig. 2. Visualizing partially ordered event data. Each interval shown in Fig. 2a, i.e., an activity instance, describes the execution of an activity. $A, \ldots, G$ represent activity labels. Both visualized cases/process instances (Fig. 2a) correspond to the same interval order (Fig. 2b). Note that we consider two activity instances to be unrelated if they overlap in time

Section 4 introduces the proposed visualization approach. Section 5 presents an experimental evaluation, and Sect. 6 concludes this paper.

2 Related Work

For a general overview of process mining, we refer to [1]. Note that the majority of process mining techniques assume totally ordered event data. For example, in process discovery few algorithms exist that utilize life cycle information, i.e., more than one timestamp, of the recorded process activities. For instance, the Inductive Miner algorithm has been extended in [9] to utilize start- and complete-timestamps of process activities. Also in conformance checking there exist algorithms that utilize life cycle information, e.g., [10]. A complete overview of techniques utilizing life cycle information is outside the scope of this paper.

In [6], the authors present a methodology for conducting process mining projects and highlight the importance of visual analytics. In [8], open challenges regarding visual analytics in process mining are presented. The visualization of time-oriented event data—the topic of this paper—is identified as a challenge.

The classic variant explorer as shown in Fig. 1 can be found in many different process mining tools, e.g., in ProM[1], which is an open-source process mining software tool. In [3], the authors present a software tool to visualize event data.

[1] https://www.promtools.org.

Table 1. Example of event data

Event-ID	Case-ID	Activity Label	Start-timestamp	Complete-timestamp	Resource	...
1	1	activity A	07/13/2021 08:00	07/13/2021 09:30	staff	...
2	1	activity B	07/13/2021 08:30	07/13/2021 11:00	staff	...
3	1	activity C	07/13/2021 09:00	07/13/2021 12:00	staff	...
4	1	activity D	07/13/2021 11:30	07/13/2021 13:30	staff	...
5	1	activity E	07/13/2021 11:40	07/13/2021 13:00	supervisor	...
6	1	activity F	07/13/2021 14:00	07/13/2021 15:00	manager	...
7	1	activity A	07/13/2021 14:30	07/13/2021 16:00	staff	...
8	1	activity G	07/13/2021 16:30	07/13/2021 17:00	staff	...
9	2	activity A	07/13/2021 08:00	07/13/2021 09:30	staff	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮

Various visualizations of event data are offered; however, a variant explorer, as considered in this paper, is not available. In [2], the authors present a plugin for ProM to visualize partially ordered event data. The approach considers events to be atomic, i.e., an event representing the start and an event representing the completion of an activity are considered to be separate events. Based on a user-selected time granularity, events within the same time segment are aggregated, i.e., they are considered and visualized to be executed in parallel. This offers the advantage that the user can change the visualization depending on how accurately the timestamps are to be interpreted. Compared to our approach, we consider non-atomic activity instances, i.e., we map start and complete events of a process activity to an activity instance. Next, we relate these activity instances to each other instead of atomic events as proposed in [2]. Therefore, both approaches, the one presented in [2] and the one presented in this paper, can coexist and each have their advantages and disadvantages.

3 Preliminaries

In this section, we present concepts and definitions used within this paper.

Event data describes the historical execution of processes. Table 1 shows an example of said event data. Each row corresponds to an event, i.e., in the given example an *activity instance*.[2] For example, the first event, identified by event-id 1, recorded that activity A has been executed from 08:00 until 09:30 at 07/13/2021 within the process instance identified by case-id 1.

In general, activity instances describe the execution of a process activity within a specific case. A *case* describes a single execution of a process, i.e., a process instance, and it is formally a set of activity instances that have been executed for the same case. Activity instances consist of at least the following attributes:

[2] Note that in some event logs, the start and the completion of an activity are separate events (i.e., separate rows). Observe that such records are easily transformed to our notion of event data.

an identifier, a case-id, an activity label, a start-timestamp, and a complete-timestamp. Since we are only interested in the order of activity instances within a case and not in possible additional attributes of an activity instance, we define activity instances as a 5-tuple.

Definition 1 (Universes). *$\mathcal{T}$ is the universe of totally ordered timestamps. $\mathcal{L}$ is the universe of activity labels. $\mathcal{C}$ is the universe of case identifiers. $\mathcal{I}$ is the universe of activity instance identifiers.*

Definition 2 (Activity Instance). *An activity instance $(i, c, l, t_s, t_c) \in \mathcal{I} \times \mathcal{C} \times \mathcal{L} \times \mathcal{T} \times \mathcal{T}$ describes the execution of an activity labeled l within the case c. The start-timestamp of the activity's execution is t_s, and the complete-timestamp is t_c, where $t_s \leq t_c$. Each activity instance is uniquely identifiable by i. We denote the universe of activity instances by $\mathcal{A}$.*

Note that any event log with only one timestamp per executed activity can also be easily expressed in terms of activity instances, i.e., $t_s = t_c$. For a given activity instance $a = (i, c, l, t_s, t_c) \in \mathcal{A}$, we define projection functions: $\pi^i(a) = i$, $\pi^c(a) = c$, $\pi^l(a) = l$, $\pi^{t_s}(a) = t_s$, and $\pi^{t_c}(a) = t_c$.

Definition 3 (Event Log). *An event log E is a set of activity instances, i.e., $E \subseteq \mathcal{A}$ such that for $a_1, a_2 \in E \wedge \pi^i(a_1) = \pi^i(a_2) \Rightarrow a_1 = a_2$. We denote the universe of event logs by $\mathcal{E}$.*

For a given event log $E \in \mathcal{E}$, we refer to the set of activity instances executed within a given case $c \in \mathcal{C}$ as a *trace*, i.e., $T_c = \{a \in E \mid \wedge \pi^c(a) = c\}$. As shown in Fig. 2a, we can visualize a trace and its activity instances in a time plot.

Note that each activity instance $a = (i, c, l, t_s, t_c) \in \mathcal{A}$ defines an interval on the timeline, i.e., $[t_s, t_c]$. A collection of intervals—in this paper we focus on traces—defines an *interval order*. In general, given two activity instances $a_1, a_2 \in \mathcal{A}$, we say $a_1 < a_2$ iff $\pi^{t_c}(a_1) < \pi^{t_s}(a_2)$. Note that interval orders are a proper subclass of strict partial orders [7]; hence, interval orders satisfy: irreflexivity, transitivity, and asymmetry. Interval orders additionally satisfy the *interval order condition*, i.e., for any $x, y, w, z : x < w \wedge y < z \Rightarrow x < z \vee y < w$ [7].

In this paper, we represent an interval order as a directed, labeled graph that consists of vertices V, representing activity instances, and directed edges $V \times V$, representing ordering relations between activity instances. Figure 2b shows the interval order of the traces shown in Fig. 2a. We observe that the first two activity instances labeled with A and B are incomparable to each other because there is no arc from either A to B or vice versa. Thus, the first execution of A and B are executed in parallel, i.e., their intervals overlap. For example, activity C is related to F, G and the second execution of A. Thus, C is executed before F, G and the second execution of A. Next, we formally define the construction of the directed graph representing the interval order of a trace.

Definition 4 (Interval Order of a Trace). *Given a trace $T_c \subseteq \mathcal{A}$, we define the corresponding interval order as a labeled, directed graph (V, E, λ) consisting of vertices V, directed edges $E = (V \times V)$, and a labeling function $\lambda : V \to \mathcal{L}$. The*

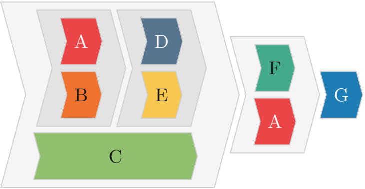


Fig. 3. Proposed visualization for the interval order shown in Fig. 2b

set of vertices is defined by $V{=}T_c$ *with* $\lambda(a){=}\pi^l(a)$*. Given two activity instances* $a^1, a^2{\in}T$*, there is a directed edge* $\big(\pi^i(a_1), \pi^i(a_2)\big){\in}E$ *iff* $\pi^{t_c}(a_1){<}\pi^{t_s}(a_2)$*. We denote the universe of interval orders by* $\mathcal{P}$*.*

Next, we define the induced interval order.

Definition 5 (Induced Interval Order). *Given* $(V, E, \lambda){\in}\mathcal{P}$*. For* $V'{\subseteq}V$*, we define the induced interval order, i.e., the induced subgraph,* $(V', E', \lambda'){\in}\mathcal{P}$ *with* $E'{=}E{\cap}(V'{\times}V')$ *and* $\lambda'(v){=}\lambda(v)$ *for all* $v{\in}V'$*.*

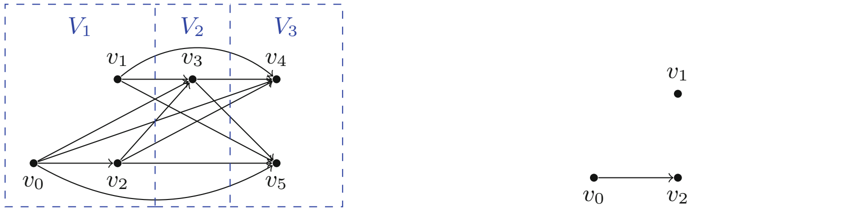
4 Visualizing Trace Variants

This section introduces the proposed approach to visualize trace variants from partially ordered event data. Section 4.1 introduces the approach, and Sect. 4.2 proves that the approach is deterministic. Section 4.3 discusses the potential limitations of the approach. Finally, Sect. 4.4 covers the implementation.

4.1 Approach

The proposed visualization approach of trace variants is based on chevrons, a graphical element known from classical trace variant visualizations (cf. Fig. 1). Figure 3 shows an example of the proposed visualization for the interval order given in Fig. 2b. The interpretation of a chevron as indicating sequential order is maintained in our approach. Additionally, chevrons can be nested and stacked on top of each other. Stacked chevrons indicate parallel/overlapping execution of activities. Nested chevrons relate groups of activities to each other. In the given example, the first chevron indicates that C is executed in parallel to A, B, D, and E. The two upper chevrons indicate that A and B are executed in parallel, but are executed before D and E, both of which are also executed in parallel.

The proposed approach assumes an interval order, representing a trace variant, as input and *recursively* partitions the interval order by applying cuts to compute the layout of the visualization (cf. Fig. 3). In general, a cut is a partition of the nodes of a given interval order. Based on the partition, induced interval orders are derived. Each application of such a cut corresponds to chevrons and their positioning in the final visualization, e.g., stacked or side-by-side chevrons.



(a) Interval order and a maximal ordering cut (b) Induced interval order based on V_1

Fig. 4. Example of an ordering cut, i.e., a partition of the nodes into $V_1 {=} \{v_0, v_1, v_2\}$, $V_2 {=} \{v_3\}, V_3 {=} \{v_4, v_5\}$, and one corresponding induced interval order for V_1

Nested chevrons result from the recursive manner. Next, we define the computation of the proposed layout, i.e., we define two types of cuts.

An *ordering cut* partitions the activity instances into sets such that these sets can be totally ordered, i.e., all activity instances within a set can be related to all other activity instances from other sets. In terms of the graph representation of an interval order, this implies that all nodes from one partition have a directed edge to all nodes from the other partition(s). We depict an example of an ordering cut in Fig. 4. Note that all nodes in V_1 are related to all nodes in V_2 and V_3. Next, we formally define an ordering cut for an interval order.

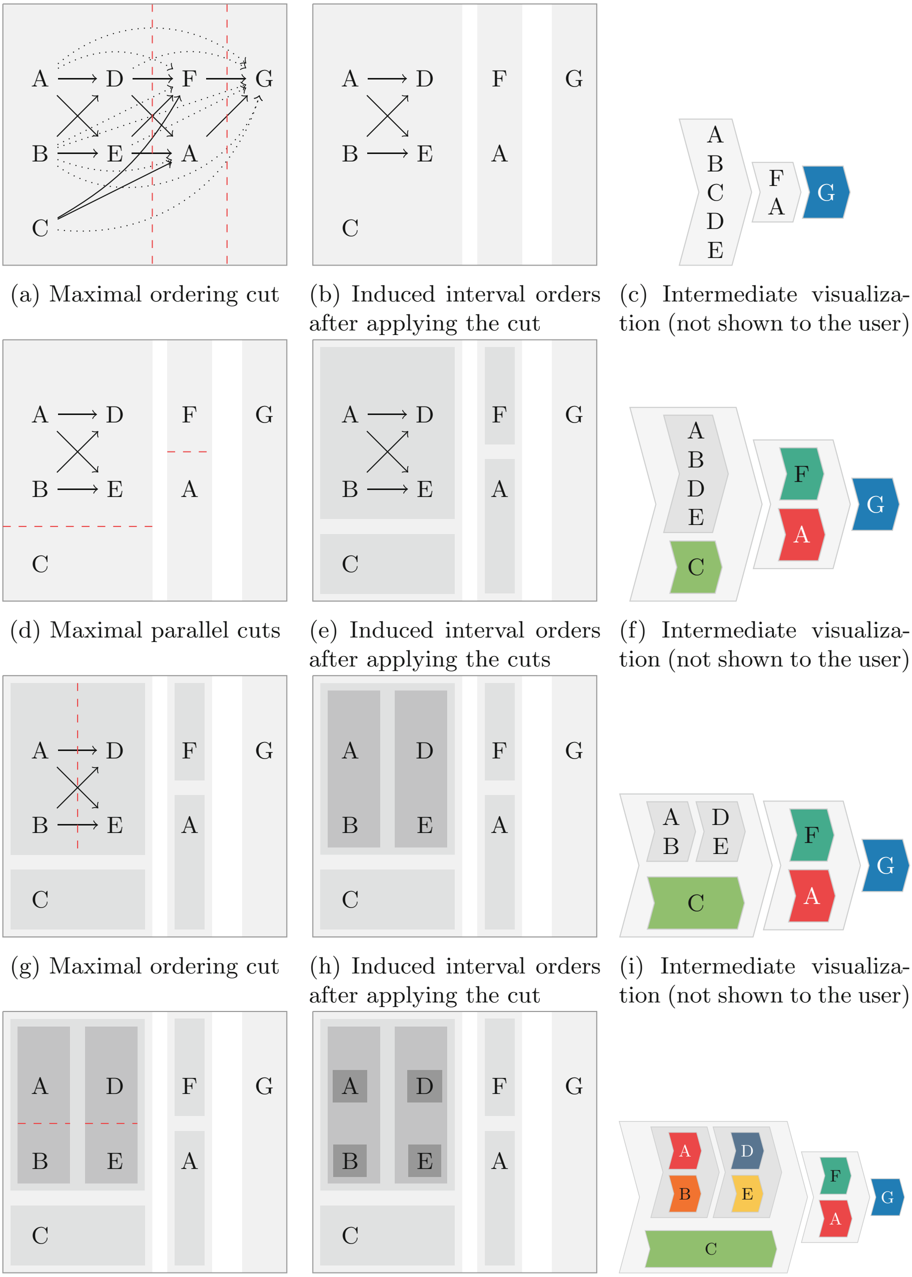
Definition 6 (Ordering Cut). *Assume an interval order* $(V, E, \lambda) {\in} \mathcal{P}$*. An ordering cut describes a partition of the nodes* V *into* $n{>}1$ *non-empty subsets* $V_1, \ldots, V_n$ *such that:* $\forall 1 {\leq} i {<} j {\leq} n \Big(\forall v {\in} V_i, v' {\in} V_j \big((v, v') {\in} E \big) \Big)$.

A *parallel cut* indicates that activity instances from one partition overlap in time with activity instances in the other partition(s), i.e., activity instances from different partitions are unrelated to each other. Thus, we are looking for *components* in the graph representation of an interval order.

Definition 7 (Parallel Cut). *Assume an interval order* $(V, E, \lambda) {\in} \mathcal{P}$*. A parallel cut describes a partition of the nodes* V *into* $n{\geq}1$ *non-empty subsets* $V_1, \ldots, V_n$ *such that* $V_1, \ldots, V_n$ *represent connected components of* (V, E, λ)*, i.e.,* $\forall 1 {\leq} i {<} j {\leq} n \Big(\forall v {\in} V_i \forall v' {\in} V_j \big((v, v') {\notin} E \wedge (v', v) {\notin} E \big) \Big)$.

We call a cut *maximal* if n, i.e., the number of subsets, is maximal.

Figure 5 shows an example of the proposed visualization approach. We use the interval order from Fig. 2b as input. The visualization approach recursively looks for a maximal ordering or parallel cut. In the example, we initially find an ordering cut of size three (cf. Fig. 5a). Given the cut, we create three induced interval orders (cf. Fig. 5b). As stated before, each induced interval order created by a cut represents a chevron. In general, an ordering cut indicates the horizontal alignment of chevrons while a parallel cut indicates the vertical alignment of chevrons. Since we found an ordering cut of size three, the intermediate visualization consists of three horizontally-aligned chevrons (cf. Fig. 5c). If an induced



(a) Maximal ordering cut

(b) Induced interval orders after applying the cut

(c) Intermediate visualization (not shown to the user)

(d) Maximal parallel cuts

(e) Induced interval orders after applying the cuts

(f) Intermediate visualization (not shown to the user)

(g) Maximal ordering cut

(h) Induced interval orders after applying the cut

(i) Intermediate visualization (not shown to the user)

(j) Maximal parallel cuts

(k) Induced interval orders after applying the cuts

(l) Final visualization

Fig. 5. Example of recursively applying ordering and parallel cuts to an interval order and the corresponding visualization

interval order only consists of one element (e.g., the third induced interval order in Fig. 5b), we fill the corresponding chevron with a color that is unique for the given activity label (cf. Fig. 5c). As in the classic trace variant explorer, colors are used to better distinguish different activity labels.

We now recursively apply cuts to the induced interval orders. In the first two interval orders, we apply a parallel cut (cf. Fig. 5d). The third interval order consists only of one node labeled with G; thus, no further cuts can be applied. Figure 5e shows the induced interval orders after applying the two parallel cuts. As stated before, time-overlapping activity instances are indicated by stacked chevrons. Since both applied parallel cuts have size two, we create two stacked chevrons each within the first and the second chevron (cf. Fig. 5f). After another ordering cut (cf. Fig. 5g–5i) and two more parallel cuts (cf. Fig. 5j), the visualization approach stops because all induced interval orders consist of only one activity instance. Figure 5l shows the final visualization.

4.2 Formal Guarantees

Next, we show that the proposed approach is deterministic, i.e., the same visualization is always returned for the same interval order. We therefore show that different cuts cannot coexist, i.e., either a parallel cut, an ordering cut, or no cut exists in an interval order. Further, we show that maximal cuts are unique.

Lemma 1 (Cuts Cannot Coexist). *In an interval order* $(V, E, \lambda){\in}\mathcal{P}$ *a parallel and an ordering cut cannot coexist.*

Proof. Let $(V, E, \lambda){\in}\mathcal{P}$ be an interval order with an ordering cut $V_1, \dots, V_n$ for some $n{\geq}2$. Assume there exists a parallel cut, too, i.e., $V'_1, \dots, V'_m$ for some $m{\geq}2$. For $1{\leq}j{\leq}m$, assume that for an arbitrary $v{\in}V$ it holds that $v{\in}V'_j$ such that $v{\in}V_i$ for some $i{\in}\{1, \dots, n\}$. Since an ordering cut exists, we know that $\forall w{\in}V_{i+1}{\cup}\dots{\cup}V_n\big((v, w){\in}E\big)$ and $\forall w'{\in}V_1{\cup}\dots{\cup}V_{i-1}\big((w', v){\in}E\big)$. Since $V'_1, \dots, V'_m$ is a parallel cut, i.e., each $V'_k{\in}\{V'_1, \dots, V'_m\}$ represents a connected component (Definition 7), also all w and w' must be in V'_j. Hence, $V'_j{=}\{v\}{\cup}V_1{\cup}\dots{\cup}V_{i-1}{\cup}V_{i+1}{\cup}\dots{\cup}V_n$. Further, since $\forall w'{\in}V_1{\cup}\dots{\cup}V_{i-1}\forall w{\in}V_{i+1}{\cup}\dots{\cup}V_n\big((w', w){\in}E \wedge (w', v){\in}E \wedge (v, w){\in}E\big)$ it follows by Definition 7 that $V'_j{=}V_1{\cup}\dots{\cup}V_n{=}V$. Hence, $\forall V'_k{\in}\{V'_1, \dots, V'_m\}\setminus\{V'_j\}$ $(V'_k{=}\emptyset)$ since $V'_1, \dots, V'_m$ is a partition of V. This contradicts our assumption that there exists a parallel cut, too. The other direction is symmetrical. □

Since cuts cannot coexist (cf. Lemma 1), one cut is applicable for a given interval order at most. Next, we show that maximal cuts are unique.

Lemma 2 (Maximal Ordering Cuts Are Unique). *If an ordering cut exists in a given interval order* $(V, E, \lambda){\in}\mathcal{P}$*, the maximal ordering cut is unique.*

Proof. Proof by contradiction. Assume an interval order $(V, E, \lambda){\in}\mathcal{P}$ having two *different* maximal ordering cuts, i.e., $V_1, \dots, V_n$ and $V'_1, \dots, V'_n$.
$\Rightarrow \exists i{\in}\{1, \dots, n\}\forall j{\in}\{1, \dots, n\}\big(V_i{\neq}V'_j\big) \Rightarrow V_i{\neq}V'_i$

$\Rightarrow \exists v{\in}V_i{\cup}V_i'\big((v{\in}V_i \wedge v{\notin}V_i') \vee (v{\notin}V_i \wedge v{\in}V_i')\big)$
Assume $v{\in}V_i \wedge v{\notin}V_i'$ (the other case is symmetric)
$\Rightarrow v{\in}V_1'{\cup}\ldots{\cup}V_{i-1}'{\cup}V_{i+1}'{\cup}\ldots{\cup}V_n'{=}V{\setminus}V_i'$

1) Assume $v{\in}V_1'{\cup}\ldots{\cup}V_{i-1}'$
$\xRightarrow{\textit{Definition } 6} \forall v'{\in}V_i\big((v,v'){\in}E\big)$

2) Assume $v{\in}V_{i+1}'{\cup}\ldots{\cup}V_n'$
$\xRightarrow{\textit{Definition } 6} \forall v'{\in}V_i\big((v',v){\in}E\big)$

$\xRightarrow{v\in V_i} (v,v){\in}E$ contradicts the assumption (V,E,λ) represents an interval order because irreflexivity is not satisfied. □

Lemma 3 (Maximal Parallel Cuts Are Unique). *If a parallel cut exists in a given interval order $(V,E,\lambda){\in}\mathcal{P}$, the maximal parallel cut is unique.*

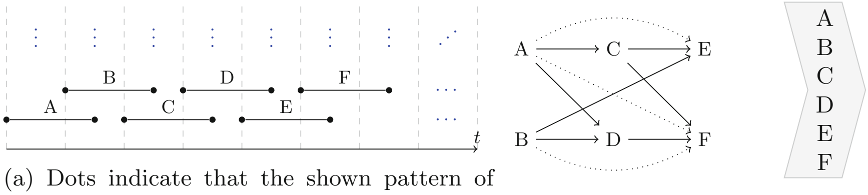
Proof (Lemma 3). By definition, components of a graph are unique. □

Lemma 2 and Lemma 3 show that maximal cuts, both ordering and parallel, are unique. Together with Lemma 1, we derive that the proposed visualization approach is *deterministic*, i.e., the approach always returns the same visualization for the same input, because for a given interval order only one cut type is applicable at most and if a cut exists, the maximal cut is unique.

4.3 Limitations

In this section, we discuss the limitations of the proposed visualization approach.

Reconsider the example in Fig. 5. Cuts are recursively applied until one node, i.e., an activity instance, remains in each induced interval order (cf. Fig. 5k). However, there are certain cases in which the proposed approach cannot apply cuts although more than one node exists in an (induced) interval order.



(a) Dots indicate that the shown pattern of chained activity instances can be extended arbitrarily, horizontally as well as vertically

(b) Corresponding interval order

(c) Corresponding visualization

Fig. 6. Example trace and interval order in which no cuts are applicable

Consider Fig. 6a, showing an example of a trace where no cuts can be applied. Since each activity instance is overlapping with some other activity instance, we cannot apply an ordering cut. Also, since there is no activity instance that overlaps with all other activity instances, we cannot apply a parallel cut. Note that the visualized pattern of chained activity instances can be arbitrarily extended by adding more activity instances vertically and horizontally, indicated by the dots in Fig. 6a. Figure 6b shows the corresponding interval order.

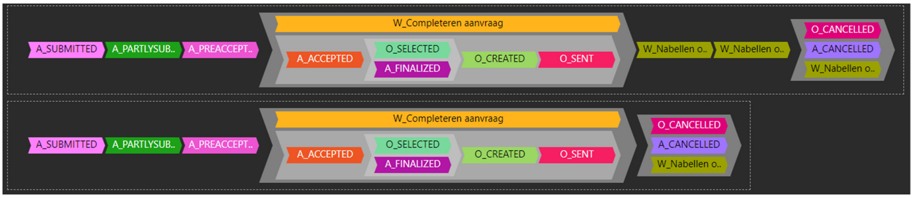


Fig. 7. Screenshot of Cortado's variant explorer showing real-life event data [4]

Table 2. Evaluation results based on real-life event logs

	Log statistics		**Calculation time (s) of interval ordered variants**				**Variants statistics**		
Event log	#cases (avg. #events per case)	multiple timestamps per activity available	Total calculation	Pre-processing event data	Creating interval orders	Cutting interval orders	#classic variants (only start-time-stamp considered)	#interval ordered variants	#interval ordered variants with limitations
BPI Ch. 2017 [5]	31,509 (≈38)	yes	≈39	≈21.6	≈5.7	≈11.3	15,930	5,854	335 (≈6%)
BPI Ch. 2012 [4]	13,087 (≈20)	yes	≈22.6	≈5.9	≈5.4	≈11.2	4,366	3,830	0 (≈0%)
Sepsis [11]	1,050 (≈14)	no	≈1.9	≈0.3	≈0.3	≈1.2	846	690	0 (≈0%)

For the example trace, the proposed approach visualizes the activities $A, \dots, F$ within a single chevron, indicating that the activities are executed in an unspecified order (cf. Fig. 6c). Thus, the visualization highly simplifies the observed process behavior in such cases. Alternatively, it would be conceivable to show the interval order within a chevron if an (induced) interval order cannot be cut anymore. However, we decided to keep the visualization simple and show all activities within a single chevron. Note that this design decision entails that the expressiveness of the proposed visualization is lower than the graphical notation of interval orders, i.e., different interval orders can have the same visualization.

4.4 Implementation

The proposed visualization approach for partially ordered event data has been implemented in *Cortado* [12][3], which is a standalone tool for interactive process discovery. Figure 7 shows a screenshot of Cortado visualizing an event log with partially ordered events. The implemented trace variant explorer works for both, partially and totally ordered event data. The tool assumes an event log in the `.xes` format as input. If the provided event log contains start- and complete-timestamps, the visualization approach presented in this paper is applied.

[3] Available from version 1.2.0, downloadable from: https://cortado.fit.fraunhofer.de/.

5 Evaluation

In this section, we evaluate the proposed visualization approach. We focus thereby on the performance aspects of the proposed visualization. Further, we focus on the limitations, i.e., no cuts can be applied anymore, although the (induced) interval order has more than one element, as discussed in Sect. 4.3.

We use publicly available, real-life event logs [4,5,11]. Table 2 shows the results. The first three columns show information about the logs. Two logs [4,5] contain start- and complete-timestamps per activity instance while one log [11] contains only a single timestamp per activity instance. Regarding the total calculation time, we note that the duration of the visualization calculation is reasonable from a practical point of view. We observe that the recursive application of cuts takes up most of the computation time in all logs, as expected. Regarding the variants, we observe that the number of classic variants is higher compared to the number of variants derived from the interval order for all event logs. We observe this even for the third event log [11] because some activities within the cases share the same timestamp. Regarding the limitations of the approach, as discussed in Sect. 4.3, we observe that only in the first log [5] approximately in 6% of all trace variants patterns occur where it was not possible to apply cuts anymore. Note that the limitation cannot occur in event logs where only a single timestamp per activity is available, e.g., [11].

6 Conclusion

This paper introduced a novel visualization approach for partially ordered event data. Based on chevrons, known from the classic trace variant explorer, our approach visualizes the ordering relations between process instances in a hierarchical manner. Our visualization allows to easily identify common patterns in trace variants from partially ordered event data. The approach has been implemented in the tool *Cortado* and has been evaluated on real-life event logs.

References

1. van der Aalst, W.M.P.: Data science in action. In: Process Mining, pp. 3–23. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4_1
2. van der Aalst, W.M.P., Santos, L.: May i take your order? On the interplay between time and order in process mining. arXiv preprint arXiv:2107.03937 (2021). https://doi.org/10.1007/978-3-030-94343-1_8
3. Bodesinsky, P., Alsallakh, B., Gschwandtner, T., Miksch, S.: Exploration and assessment of event data. In: EuroVis Workshop on Visual Analytics (EuroVA). The Eurographics Association (2015). https://doi.org/10.2312/eurova.20151106
4. van Dongen, B.: BPI Challenge 2012 (2012). https://data.4tu.nl/articles/dataset/BPI_Challenge_2012/12689204
5. van Dongen, B.: BPI Challenge 2017 (2017). https://data.4tu.nl/articles/dataset/BPI_Challenge_2017/12696884

6. van Eck, M.L., Lu, X., Leemans, S.J.J., van der Aalst, W.M.P.: PM^2: a process mining project methodology. In: Zdravkovic, J., Kirikova, M., Johannesson, P. (eds.) CAiSE 2015. LNCS, vol. 9097, pp. 297–313. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19069-3_19
7. Fishburn, P.C.: Intransitive indifference with unequal indifference intervals. J. Math. Psychol. **7**(1), 144–149 (1970). https://doi.org/10.1016/0022-2496(70)90062-3
8. Gschwandtner, T.: Visual analytics meets process mining: challenges and opportunities. In: Ceravolo, P., Rinderle-Ma, S. (eds.) SIMPDA 2015. LNBIP, vol. 244, pp. 142–154. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-53435-0_7
9. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Using life cycle information in process discovery. In: Reichert, M., Reijers, H.A. (eds.) BPM 2015. LNBIP, vol. 256, pp. 204–217. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-42887-1_17
10. Lu, X., Fahland, D., van der Aalst, W.M.P.: Conformance checking based on partially ordered event data. In: Fournier, F., Mendling, J. (eds.) BPM 2014. LNBIP, vol. 202, pp. 75–88. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15895-2_7
11. Mannhardt, F.: Sepsis cases - event log (2016). https://data.4tu.nl/articles/dataset/Sepsis_Cases_-_Event_Log/12707639
12. Schuster, D., van Zelst, S.J., van der Aalst, W.M.P.: Cortado—An interactive tool for data-driven process discovery and modeling. In: Buchs, D., Carmona, J. (eds.) PETRI NETS 2021. LNCS, vol. 12734, pp. 465–475. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-76983-3_23

Analyzing Multi-level BOM-Structured Event Data

Tobias Brockhoff[1](✉), Merih Seran Uysal[1], Isabelle Terrier[2], Heiko Göhner[2], and Wil M. P. van der Aalst[1]

[1] Process and Data Science Chair, RWTH Aachen University, Aachen, Germany
{brockhoff,uysal,wvdaalst}@pads.rwth-aachen.de
[2] Heidelberger Druckmaschinen AG, Heidelberg, Germany

Abstract. With the advent of Industry 4.0, increasing amounts of data on operational processes (e.g., manufacturing processes) become available. These processes can involve hundreds of different materials for a relatively small number of manufactured special-purpose machines rendering classical process discovery and analysis techniques infeasible. However, in contrast to most standard business processes, additional structural information is often available—for example, Bills of Materials (BOMs), listing the required materials, or Multi-level Manufacturing Bills of Materials (M^{2}BOMs), which additionally show the material composition. This work investigates how structural information given by Multi-level Bills of Materials (M^{2}BOMs) can be integrated into a top-down operational process analysis framework to improve special-purpose machine manufacturing processes. The approach is evaluated on industrial-scale printer assembly data provided by *Heidelberger Druckmaschinen AG*.

Keywords: Process Mining · Bill of Materials · Operational Processes · Industry 4.0 · Offset Printing

1 Introduction

With the advent of digitalization, data on an increasing number of processes are recorded. *Process mining* is the emerging key discipline concerned with the analysis of such data to provide insights into processes and, eventually, to improve them. Traditionally, *event data*, i.e., a set of discrete events that are linked by a certain case notion, have been recorded in business management systems, which, for example, handle *order-to-cash* or *purchase-to-pay* processes. However, with the rise of *Industry 4.0*, more and more event data from manufacturing and assembly processes become available. The analysis of these so-called *operational processes* [1] using process mining is therefore key to not only remove friction from companies' administrative workflows but also to optimize and steer their manufacturing processes.

In contrast to standard business processes, *operational* processes frequently provide additional structural information. A common approach to structure

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 47–59, 2022.
https://doi.org/10.1007/978-3-030-98581-3_4

the production, particularly for complex products, is by means of a **M**ultilevel **M**anufacturing **B**ill **o**f **M**aterials (M^{2}BOM) that shows the hierarchical composition of required materials. These models are for example supported by manufacturing ERP systems such as *SAP* [13]. Moreover, *operational processes* often involve a large number of materials and assembly tasks that render fully-automatic model discovery infeasible. However, model discovery plays a central role in classical process mining-based process analysis frameworks such as PM^2 [6], where the mining & analysis stage implementation comprises automatic model discovery, conformance checking, and model enhancement. Thus, the adaptability of standard analysis approaches to *operational processes* is limited. Thus, given *operational* event data where each assembly case is endowed with its corresponding M^{2}BOM, we propose a two-stage refinement of PM^2's mining & analysis step. The first substage targets a general and comprehensive performance overview exploiting additional structural information; the second substage concerns the analysis of subprocesses of interest identified in the first stage. To implement the first substage, we propose a method that discovers a tree-based assembly model close to the original M^{2}BOM and, therefore, well-suited to convey results to stakeholders from engineering. In doing so, we particularly focus on performance. Due to practical constraints, the actual process usually adheres to the provided M^{2}BOM (e.g., parts cannot be missing and dependencies must be respected) and, thus, conformance checking tends to be less interesting.

Our main contributions are the investigation of so-called M^{2}BOM-structured event logs and how multiple M^{2}BOMs, with the help of domain knowledge and special types of material options, can be unified into a single common data model. We propose to detect bottlenecks based on this unified representation and to analyze the latter using a top-down approach. Finally, we illustrate the feasibility of our approach on an industrial-scale printer assembly use case provided by *Heidelberger Druckmaschinen AG*.

The remainder of this paper is structured as follows: Sects. 2 and 3 cover the related work and preliminaries, respectively. Section 4 presents the analysis approach, in particular, the discovery of M^{2}BOM-based models in Sect. 4.2. We evaluate the approach in Sect. 5 and conclude our work in Sect. 6.

2 Related Work

There are many papers on improving the performance of manufacturing processes—for example, based on the principles of lean management [11]. We, however, focus on the more recent approach of using process mining to analyze and improve operational assembly processes. For a more detailed review on process mining for assembly-related processes (e.g., procurement), we refer the reader to [5]. One of the first case studies, conducted by Rozinat et al. [12], investigates the testing procedure of wafer scanners in terms of idle times and repeated tests. In this work, little additional structural information has been exploited. More similar to our use case in terms of independently manufactured parts is the ship manufacturing process in [9], where multiple ship blocks are manufactured simultaneously. In contrast to our work, this work focuses on individual blocks only, applying trace clustering to identify similar intra-block assembly work flows. A comparison between planned and de facto schedules in block-level ship manufacturing processes can be found in [10]. Besides, model discovery for a coffee machine manufacturing process using standard automatic mining approaches has been investigated in [3]. Recently, Uysal et al. [15] analyzed the performance and evolution of an automotive production line. While this work also focuses on the performance of an assembly line, we consider more complex production lines and do not require a ground truth production model. However, both use cases share a similar assembly structure in common (i.e., major assembly steps, linked by additional structural information, and a set of unstructured assembly activities related to each major assembly step). More recently, Lorenz et al. [8] analyzed deviations between de jure and de facto models in sanitary product manufacturing using conformance checking. They emphasize that the major advantage of process mining over traditional methods is its adaptability to dynamic processes and that it can comprehensively consider entire cases. This strength is further underpinned by its application to production change point detection in [4]. Finally, a first framework for the end-to-end analysis of production processes using process mining has been proposed in [14].

3 Preliminaries

Throughout this paper, we use *out-trees* to model a bill of materials. Given a set of vertices V, a directed acyclic and weakly connected graph $T = (V, E)$ with $E \subseteq V \times V$ is a *tree*. We denote the set of vertices by $T_v = V$. A *rooted tree* is a tree with designated root vertex ρ_T and an *out-tree* is a tree where each edge points away from ρ_T. An s-t path for $s, t \in V$ is a sequence of edges $\langle e_1 = (s, v_1), e_2 = (v_1, v_2), \ldots, e_k = (v_{k-1}, t)\rangle$, $e_i \in E, i = 1, \ldots, k$.

In this work, we use restricted, loop-free, process trees to describe execution/replay semantics.

Definition 1 (Loop-free Process Tree). *Let $\mathcal{A}$ denote a universe of activity labels such that $\tau \notin \mathcal{A}$. Let $\oplus = \{\rightarrow, \times, \wedge, \vee\}$ be the set of tree operators. A loop-free process tree is defined by the following production rules:*

- *$a \in \mathcal{A} \,\dot{\cup}\, \{\tau\}$ is a loop-free process tree*
- *$\bullet(T_1, \ldots, T_n)$ for process trees T_i, $i = 1, \ldots, n$, $n \geq 1$, and $\bullet \in \oplus$ is a loop-free process tree*

Besides, given a process tree PT, we assume standard operator semantics for the defined language $\mathcal{L}(\mathrm{PT})$ (compare [7]). Furthermore, to model the operational event data, we introduce the following universes and event projections.

Definition 2 (Event Universes). *To model the manufacturing event data, we define the universes of event identifiers, $\mathbb{U}_{e_{id}}$; product identifiers, $\mathbb{U}_{p_{id}}$; manufacturing activities, $\mathbb{U}_a$; timestamps, $\mathbb{U}_{time}$; material types, $\mathbb{U}_{m_{typ}}$; material identifiers, $\mathbb{U}_{m_{id}}$; material id to material type mappings,* $\mathrm{mat} \in \mathbb{U}_{m_{id}} \rightarrow \mathbb{U}_{m_{typ}}$*; and events, $\mathcal{E} = \mathbb{U}_{e_{id}} \times \mathbb{U}_{p_{id}} \times \mathbb{U}_a \times \mathbb{U}_{time} \times \mathbb{U}_{m_{id}}$.*

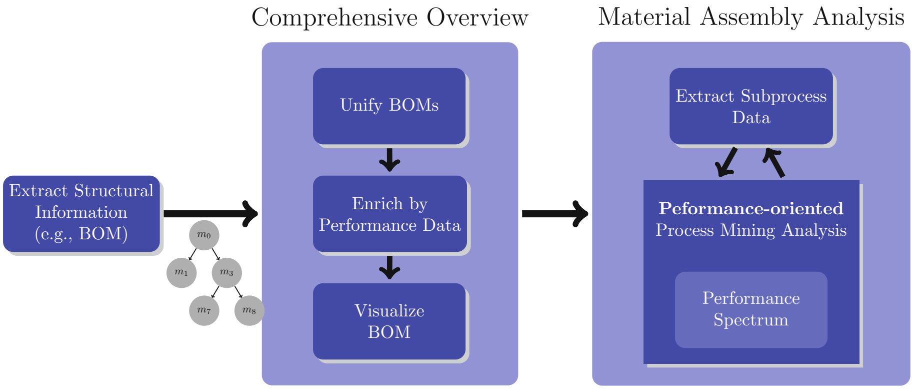

Comprehensive Overview
Material Assembly Analysis
Extract Structural Information (e.g., BOM)
m_0
m_1
m_3
m_7
m_8
Unify BOMs
Enrich by Performance Data
Visualize BOM
Extract Subprocess Data
Peformance-oriented Process Mining Analysis
Performance Spectrum

Fig. 1. Analysis methodology for M^{2}BOM-structured event logs.

Notice that each event is related to a *single* material following the bill of materials-inspired idea that tasks can be attributed to a specific material where the assembly of multiple materials is attributed to the created new material. Given an event $e = (e_{\mathrm{id}}, p_{\mathrm{id}}, \mathrm{a}, t, m_{\mathrm{id}}) \in \mathcal{E}$, we denote the projection on the event id, product id, activity, timestamp, and material id by $\pi_{e_{\mathrm{id}}}(e) = e_{\mathrm{id}}$, $\pi_{p_{\mathrm{id}}}(e) = p_{\mathrm{id}}$, $\pi_{\mathrm{a}}(e) = \mathrm{a}$, $\pi_{\mathrm{time}}(e) = t$, and $\pi_{m_{\mathrm{id}}}(e) = m_{\mathrm{id}}$, respectively. Furthermore, in a slight abuse of notation, we generalize the projection to sets, yielding multisets of attribute values, for example, $\pi_{\mathrm{a}}(E) = [\pi_{\mathrm{a}}(e) \,|\, e \in E]$ for $E \subseteq \mathcal{E}$. Finally, we introduce the following standard definition of an event log with the additional requirement that materials are not shared among different products.

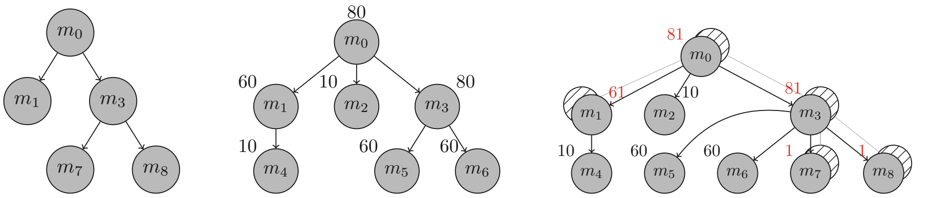
Definition 3 (Assembly Event Log). *An assembly event log* $(E, \leq_E)$ *is a finite tuple of events* $E \subseteq \mathcal{E}$ *endowed with an ordering relation* $\leq_E \subseteq E \times E$ *such that: (i)* $\leq_E$ *is a partial order, (ii) event identifiers are unique, i.e.,* $\forall e_1 \forall e_2 : \pi_{e_{id}}(e_1) = \pi_{e_{id}}(e_2) \Rightarrow e_1 = e_2$*, (iii) ordering respects time, i.e.,* $\forall e_1 \forall e_2 \in E : e_1 \leq_E e_2 \Rightarrow \pi_{time}(e_1) \leq \pi_{time}(e_2)$*, and (iv) no materials are shared, i.e.,* $\forall e_1 \forall e_2 \in E : \pi_{m_{id}}(e_1) = \pi_{m_{id}}(e_2) \Rightarrow \pi_{p_{id}}(e_1) = \pi_{p_{id}}(e_2)$.

4 Methods

In this section, we propose a top-down methodology for analyzing operational processes providing additional structural information and illustrate how a multi-level manufacturing bill of materials (M^{2}BOM) can be exploited.

4.1 Analysis Methodology

A major challenge when analyzing operational processes is the potentially large number of assembly activities and materials. Additionally, particularly in special-purpose machine manufacturing, the number of orders is usually small. This generally negatively affects automatic process discovery techniques, yielding huge



(a) M^2BOM (b) Initial shared M^2BOM (c) Merged shared M^2BOM

Fig. 2. Example of iteratively merging M^2BOMs into a shared M^2BOM.

and incomprehensible models. However, for humans, even with little domain knowledge, these processes are clearly structured and a lot of effort went into planning. In doing so, one material dependency modeling approach is by means of M^2BOM. In the proposed process analysis methodology, depicted in Fig. 1, we therefore exploit this additional structural information to be able to visualize the process as a whole. To this end, we first extract the structural information. Then, a tree-based performance-aware overview over the entire process where vertices correspond to the materials, is created to show bottlenecks and to identify other points of interest (e.g., similar materials with relatively large performance differences). After identifying points of interest, particularly performance bottlenecks, a refined analysis of the associated subprocesses is conducted. Usually, manufacturing subprocesses are designed to be independent and, thus, little information is lost by focusing on a specific subprocess. Besides, the assembly of a specific material is often fairly sequential thereby facilitating the analysis. In doing so, a control-flow and conformance analysis tends to be less interesting; instead, the major focus must be on the performance. Given the reduced complexity of the subprocess, the performance spectrum [2], with time relative to timestamp of case start, is a well-suited tool because it allows for a high-resolution performance analysis. Finally, the subprocess analysis can be iterated drilling down further.

4.2 M^{2}BOM-Structured Assembly Processes

A common approach to structure assembly processes is by means of multi-level manufacturing bills of materials (M^2BOMs). Unfortunately, M^2BOMs cannot be directly used to visualize the performance of multiple cases because products are often configurable and therefore have different bills of materials. Even though these configurations might be modeled for the customer in the ordering system, this information is lost when creating the actual manufacturing bill of materials. Thus, to provide a comprehensive assembly overview, this section proposes a method to discover an option-aware M^2BOM from the data. To this end, we first merge a collection of M^2BOMs into a common representation (compare Fig. 2) and then extend it into a proper configuration model (compare Fig. 3).

Conceptually, a M^2BOM can be modeled by a tree as follows:

Definition 4 (Multi-level Manufacturing Bill of Materials (M^{2}BOM)). *Given a finite list of materials $M \subseteq \mathbb{U}_{m_{id}}$, a multi-level manufacturing bill of materials is an out-tree (M, D).*

An example M^{2}BOM is depicted in Fig. 2(a). Since our approach operates on material types, we label the vertices by their type in our illustrations. In addition, to relate the event data of a particular product to its M^{2}BOM, we define $\sigma_{m_{\text{id}}}(p, E) = \{\pi_{m_{\text{id}}}(e) \,|\, e \in E, \pi_{p_{\text{id}}}(e) = p_{\text{id}}\}$, which selects the materials used in the assembly of the product p_{id} for the assembly event log $(E, \leq_E)$. Next, we combine the classical event log and M^{2}BOM into a *M^{2}BOM-structured event log*.

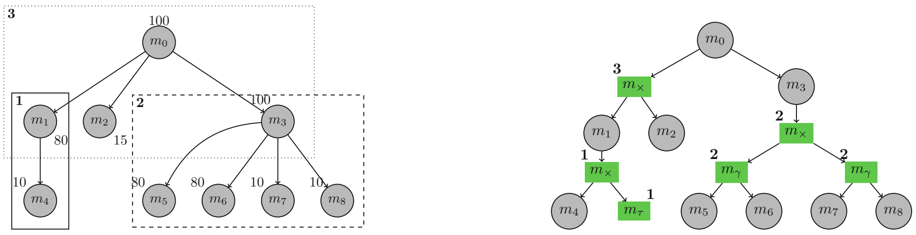
Definition 5 (M^{2}BOM-structured event log). *Let $(E, \leq_E)$ denote an assembly event log. Let $BOM_E : \mathbb{U}_{p_{id}} \to (\mathcal{P}(\mathbb{U}_{m_{id}}) \times (\mathcal{P}(\mathbb{U}_{m_{id}} \times \mathbb{U}_{m_{id}}))\,, p_{id} \mapsto (M, D)$ be a function that assigns M^{2}BOM to each product such that $M = \sigma_{m_{id}}(p, E)$. An M^{2}BOM-structured event log is a tuple $((E, \leq_E), BOM_E)$.*

We deliberately keep the event log and M^{2}BOMs separate—requiring that material from the log occurs in the M^{2}BOM and vice versa—to facilitate the use of other process mining techniques. Even though the performance of the assembly for a single product can be measured and projected onto the corresponding M^{2}BOM using the M^{2}BOM-structured event log, this does not provide aggregated statistics. Therefore, we first merge the M^{2}BOMs into one shared M^{2}BOM.

Definition 6 (Shared M^{2}BOM). *Let $E_{BOM} = ((E, \leq_E), BOM_E)$ be an M^{2}BOM-structured event log. Let $M^\sigma \subseteq \mathbb{U}_{m_{id}} \times \mathbb{U}_{m_{typ}} \times \mathcal{P}(\mathbb{U}_{p_{id}})$ be a vertex set with id, material type, and product set projections $\pi_{m_{id}}(v) = m_{id}$, $\pi_{m_{typ}}(v) = m_{typ}$, and $\pi_{p_{id}}(v) = s_{p_{id}}$ for $v = (m_{id}, m_{typ}, s_{p_{id}}) \in M^\sigma$. $B^\sigma = (M^\sigma, D^\sigma)$ with $D^\sigma \subseteq M^\sigma \times M^\sigma$ is a shared M^{2}BOM iff:*

- *B^σ is an out-tree (compare Definition 4)*
- *B^σ contains exactly the bills of materials present in E_{BOM}:*
  - *Each M^{2}BOM is contained: for every product id $p_{id} \in \pi_{p_{id}}(E)$ there exists an injective homomorphism $h_{p_{id}}\colon (BOM_E(p_{id}))_v \to M^\sigma$ between B and B^σ that respects the material types and product id sets, i.e., $\forall m \in (BOM_E(p_{id}))_v \left(\text{mat}(m) = \pi_{m_{typ}}(h(m)) \wedge p_{id} \in \pi_{p_{id}}(h(m))\right)$.*
  - *B^σ contains only M^{2}BOMs from the event log, i.e., $\forall v \in M^\sigma(\{p_{id} | p_{id} \in \pi_{p_{id}}(E), h_{p_{id}}^{-1}(v) \neq \emptyset\} = \pi_{p_{id}}(v))$.*

In the shared M^{2}BOM, every vertex has an id (guaranteeing uniqueness), a type, and a set of products containing this material. Since M^{2}BOM allows multiple materials instances having the same type, corresponding vertices between M^{2}BOM and the shared M^{2}BOM must be consistent in the type and location within the tree. We enforce this by the injective—no two material instances are mapped to the same vertex—homomorphism $h_{p_{\text{id}}}$. It ensures that every M^{2}BOM can be type-consistently embedded into the shared M^{2}BOM and that a vertex contains a product if and only if one of its materials is mapped to this vertex.



(a) Option contexts

(b) Option resolution

Fig. 3. Resolving material count mismatches by introducing options. (a) The option contexts and the order of retrieval. (b) The applied resolutions.

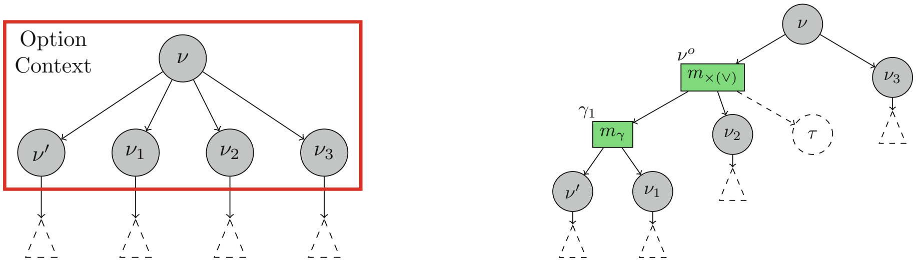
While the declarative definition does not provide a recipe for constructing the shared M^{2}BOM, there is a straightforward iterative approach that merges vertices v, v' of trees T, T' if their and their ancestors' types are consistent (i.e., the material types on the $\rho_T - v$ and $\rho_{T'} - v'$ paths coincide). An example is depicted in Fig. 2, which, for simplicity, shows the cardinality of the product id sets instead of the actual sets. Furthermore, Fig. 2(c) also shows the homomorphism between the M^{2}BOM (Fig. 2(a)) and the initial shared M^{2}BOM (Fig. 2(b)).

While the shared M^{2}BOM allows for a visualization of aggregated projected statistics, it cannot properly capture material frequency differences in terms of certain materials being optional or choices between materials. Besides, it is also desirable to link the shared M^{2}BOM to a proper process model to be able to apply other process mining techniques. For example, process simulation can be used for production planning. To this end, we transform the shared M^{2}BOM into an option M^{2}BOM that, in turn, can be directly related to a process tree. The option M^{2}BOM models optional materials and choices using dedicated special material types m_γ, $m_\vee$, $m_\times$, and m_τ. While m_γ is used to create material groups; $m_\vee$, $m_\times$, and m_τ directly correspond to their pendants in process trees. An example of the transformation is depicted in Fig. 3(b), showing that, for example, a customer may choose between m_1 and m_2.

Definition 7 (Option M^{2}BOM). *Let $\mathbb{U}^o_{m_{typ}} = \mathbb{U}_{m_{typ}} \dot{\cup} \{m_\gamma, m_\times, m_\vee, m_\tau\}$ denote an extended material type universe. An option M^{2}BOM is an out-tree $B^o = (M^o, D^o)$ with $M^o \subseteq \mathbb{U}^o_{m_{typ}} \times \mathbb{U}_{m_{id}}, D^o \subseteq M^o \times M^o$ such that m_τ only occurs as leaf vertex adjacent to a choice vertex of type $m_\times$ or $m_\vee$.*

An option M^{2}BOM directly corresponds to a process tree where material groups are modeled by concurrency and non-leaf materials as sequences of concurrent child material manufacturing followed by the assembly of the parents themselves. Accordingly, we define the process tree of an option M^{2}BOM as follows:

Definition 8 (Process Tree of an Option M^{2}BOM). *Given an option M^{2}BOM $B^o = (M^o, D^o)$ and a vertex $v \in M^o$, the process tree* $\mathrm{PT}_{B^o}(v)$*, rooted*



(a) Option context

(b) Resolution concept

Fig. 4. Transforming a shared M^{2}BOM into an option M^{2}BOM by (a) identifying option contexts and (b) applying different resolution strategies.

at v, of the option M^{2}BOM is recursively defined as follows:

$$PT_{B^o}(v) = \begin{cases} \overset{(\vee)}{\times}(\mathrm{PT}_{B^o}(c_1), \ldots, PT_{B^o}(c_n)) & \mathit{if}\ v = m_{\times(\vee)} \\ \wedge(\mathrm{PT}_{B^o}(c_1), \ldots, PT_{B^o}(c_n)) & \mathit{if}\ v = m_\gamma \\ \pi_{m_{typ}}(v) & \mathit{if}\ v\ \mathit{is\ a\ leaf} \\ \rightarrow(\wedge(\mathrm{PT}_{B^o}(c_1), \ldots, PT_{B^o}(c_n)), \pi_{m_{id}}(v)) & \mathit{if}\ v \in \mathbb{U}_{m_{typ}} \end{cases} \qquad (1)$$

where $(c_1, \ldots, c_n)$ is an arbitrary enumeration of the children of v. We denote the process tree obtained for the root of B^o by PT_{B^o} (i.e., $PT_{B^o} := PT_{B^o}(\rho_{B^o})$).

To relate an option M^{2}BOM to a concrete M^{2}BOM, we, first, introduce the material reduction of an option M^{2}BOM B^o that reduces B^o to an M^{2}BOM. It is obtained by repeatedly replacing edges $(s, u), (u, t)$ with $s, t \in \mathbb{U}^o_{m_{\mathrm{typ}}}, u \in \{m_\gamma, m_\times, m_\vee\}$ by (s, t) and removing m_τ leaves and vertices without adjacent edges. We denote the material reduction by $B^o_{|\mathbb{U}_{m_{\mathrm{typ}}}}$. For example, Fig. 4(a) shows the material reduction of the resolution depicted in Fig. 4(b). Using the material reduction and Definition 7, we can establish the link between M^{2}BOM-structured event logs and option M^{2}BOMs. M^{2}BOM B is compatible with an option M^{2}BOM if for each material in B there is a corresponding material in $B^o_{|\mathbb{U}_{m_{\mathrm{typ}}}}$ and if B is a valid combination of materials w.r.t. the options modeled in B^o (e.g., no mandatory material is missing or exclusive material options are respected). To this end, we require that a potentially valid production plan of B (i.e., materials are ordered such that child materials are manufactured before their parent materials), is contained in the language of the process tree PT_{B^o}.

Definition 9 (Option M^{2}BOM Compatibility). *Given M^{2}BOM $B = (M, D)$ and an option M^{2}BOM $B^o = (M^o, D^o)$ with its material reduction $B^o_{|\mathbb{U}_{m_{typ}}} = (M^o_r, D^o_r)$, B realizes B^o if there exists an injective homomorphism $h\colon M \rightarrow M^o_r$ between B^o and $B^o_{|\mathbb{U}_{m_{typ}}}$ satisfying the following conditions: (i) material types are respected, i.e., $\forall m \in M \pi_{m_{typ}}(m) = \pi_{m_{typ}}(h(m))$ and (ii) the post-order traversal $\langle v_1, \ldots, v_n \rangle$ of vertices in B, $\langle h(v_1), \ldots, h(v_n) \rangle$ is in the language of the process tree of B^o, i.e., $\langle h(v_1), \ldots, h(v_n) \rangle \in \mathcal{L}(PT_{B^o})$.*

Finally, a BOM-structured event log is compatible with an option M^{2}BOM if M^{2}BOM of every product is compatible. Similar to the construction of the shared M^{2}BOM, there is a straightforward approach based on comparing product sets to construct a compatible option M^{2}BOM from the shared M^{2}BOM of a BOM-structured event log. Figure 4 illustrates the major steps; first, option contexts induced by product count mismatches between parent and child vertices are iteratively retrieved in bottom-up order. Given a count mismatch between the products associated with parent v and child v' (i.e., $|\pi_{p_{\mathrm{id}}}(v)| \neq |\pi_{p_{\mathrm{id}}}(v')|$), the option context comprises v and *all* its children as these might be in a, so far undiscovered, choice relation with v'. This mismatch can then be resolved by introducing group, exclusive choice, or non-exclusive choice nodes as well as the possibility to skip certain materials; the concept of the resolution is depicted in Fig. 4(b). The resolution usually requires domain knowledge because the data may not contain all valid configurations. For example, two configurations might, by incident, never occur together even though they could. Therefore, making both optional should be preferred over an exclusive choice. Finally, the set of product ids covered by a newly introduced vertex is equal to the union of its successor's cover. Notice that for optional subtrees, the counting argument above has to be slightly modified so that these vertices are not handled repeatedly. Eventually, we obtain a valid option M^{2}BOM after resolving all options. Figure 3(b) shows a complete option resolution, including the order of steps. First, m_4 is found to be an optional part of m_2. Next, we discover an occurrence mismatch between m_3 and its child materials that can be resolved by a choice between two material groups. Finally, an exclusive choice between m_1 and m_2 is introduced.

5 Case Study

We evaluated the proposed methodology using real-world data from *Heidelberger Druckmaschinen AG*—a global manufacturer of offset, digital, and label printing presses. The company not only offers special-purpose machines but also provides services for the entire industrial printing value chain. The event data comprises events for several of hundred offset printers of different models and configurations. In agreement with the company's confidentiality policy, we anonymized the data (i.e., the activities, materials, and time spans). However, to give a high-level intuition, we assigned the materials in the first four M^{2}BOM levels expressive names. As depicted in Fig. 5, the root element is the printer, the second level comprises logistics materials, the third level's material is required to finalize the machine (*Final Comp.*), and the fourth level comprises the major large components of an offset printer (*Large Comp.*). In addition, each event contains a reference to its and its parent's material id, which was used for the automated M^{2}BOM construction.

BOM-Based Overview. In coordination with the stakeholders, we applied the option M^{2}BOM discovery approach to the most frequently sold product. We obtained an option M^{2}BOM containing more than 250 different materials and

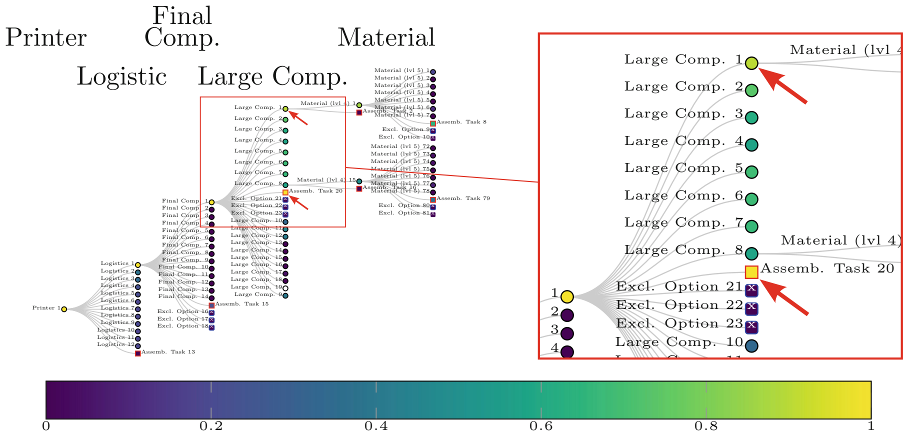


Fig. 5. An excerpt of our visualization of the option M^{2}BOM, discovered for the most frequently sold printer model (anonymized time scale).

approximately 25 choices. For each vertex v, we computed the median assembly time, i.e., the timespan between the start and complete timestamp of the first and the last event related to a material in the subtree rooted at v. Moreover, we included the business hours and the factory calendar in the computations. An excerpt of the resulting option M^{2}BOM colored by the median assembly times is depicted in Fig. 5. Starting at the root node, we expanded each level's most performance-relevant material up to a depth of three. Each circular vertex corresponds to a material, while squares correspond to options or a special *activity material* that subsumes all assembly tasks related to the parent vertex. For example, *Assembly Task 20* subsumes the events required to assemble *Final Comp. 1* using the materials on the fourth level. Besides, *Excl. Option 21* shows an optional printer part. Considering the performance of the assembly, this visualization clearly shows the most time-consuming operations—namely, *Assembly Task 20* and *Large Comp. 1*. In contrast to a plain list of assembly times, Fig. 5 also depicts the relations between the materials, facilitating performance comparison. Knowing that *Large Comp. 1-8* are similar materials, Fig. 5 shows median assembly time differences between these components. In particular, the increased assembly duration of *Large Comp. 1* compared to *Large Comp. 8* is due to a slightly increased complexity of the respective assembly tasks. However, we will focus on *Assembly Task 20*, the most time-consuming step.

Bottleneck Analysis. Next, we investigated the major bottleneck, *Assembly Task 20*. To this end, we extracted the corresponding events for all printers of the considered model and discovered a process model using the default Inductive Miner infrequent [7] algorithm. As expected, the resulting model was mostly sequential and exhibits only little concurrency. Using this model, we created the token flow-based *performance spectrum* [2]. In doing so, we exploited additional

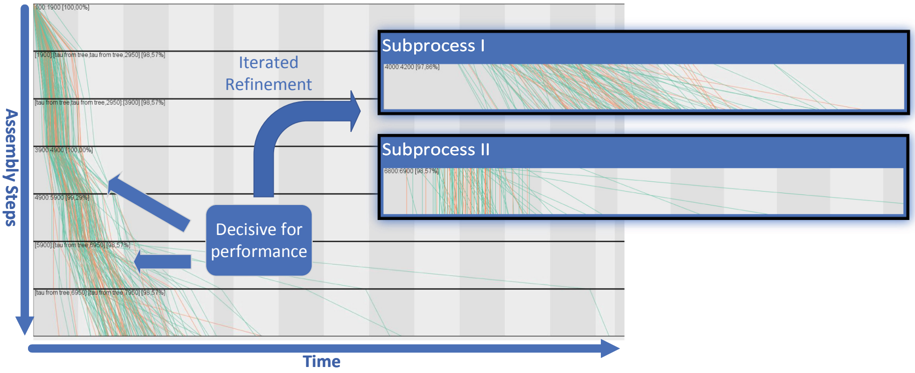


Fig. 6. Performance Spectrum for the most critical assembly activity. (Color figure online)

domain knowledge to identify sections in the subprocess. The resulting performance spectrum is depicted at the left hand side of Fig. 6; the vertical axis shows the flow of cases through the identified sections, while the horizontal axis shows the time relative to the start of the assembly. We further differentiate between standard machines (orange) and machines with additional customization and features (cyan). Using the performance spectrum, we identified two crucial sections, where times differ significantly among various machines. By iterating the subprocess analysis step, we were finally able to identify the decisive assembly tasks in terms of overall performance within the two sections. The performance spectra for these activities are depicted on the right hand side of Fig. 6.

6 Conclusion

In this work, we propose an analysis methodology for conducting a process mining analysis in operational (assembly) processes that provide additional structural information in terms of multi-level manufacturing bills of materials. Our analysis methodology uses a top-down approach that first creates an overview over the entire process, exploiting the available additional structural information, and then analyzes subprocesses in more detail. In particular, we propose an option BOM-based visualization and provide a method to discover an option M^2BOM from the assembly event data. We demonstrate the applicability of the analysis methodology, particularly the discovery and visualization of the option M^2BOM, on a real-world industrial-scale printer manufacturing use case.

For future work, we plan to extend the option M^2BOM mining approach to incorporate different printer models and to apply it to additional manufacturing domains. Moreover, incorporating process variant comparison approaches, particularly w.r.t. performance, would be interesting. Finally, as even subprocesses can be quite large, we aim to investigate methods that automatically detect performance-critical parts in performance spectra.

Acknowledgements. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy—EXC-2023 Internet of Production—390621612. We thank the Alexander von Humboldt (AvH) Stiftung for supporting our research.

References

1. van der Aalst, W.M.P., Brockhoff, T., Ghahfarokhi, A.F., Pourbafrani, M., Uysal, M.S., van Zelst, S.J.: Removing operational friction using process mining: challenges provided by the internet of production (IoP). In: Hammoudi, S., Quix, C., Bernardino, J. (eds.) DATA 2020. CCIS, vol. 1446, pp. 1–31. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-83014-4_1
2. van der Aalst, W.M.P., Tacke Genannt Unterberg, D., Denisov, V., Fahland, D.: Visualizing token flows using interactive performance spectra. In: Janicki, R., Sidorova, N., Chatain, T. (eds.) PETRI NETS 2020. LNCS, vol. 12152, pp. 369–380. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51831-8_18
3. Bettacchi, A., Polzonetti, A., Re, B.: Understanding production chain business process using process mining: a case study in the manufacturing scenario. In: Krogstie, J., Mouratidis, H., Su, J. (eds.) CAiSE 2016. LNBIP, vol. 249, pp. 193–203. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39564-7_19
4. Chiò, E., Alfieri, A., Pastore, E.: Change-point visualization and variation analysis in a simple production line: a process mining application in manufacturing. Procedia CIRP **99**, 573–579 (2021)
5. Dreher, S., Reimann, P., Gröger, C.: Application fields and research gaps of process mining in manufacturing companies. In: LNI, pp. 621–634 (2021)
6. van Eck, M.L., Lu, X., Leemans, S.J.J., van der Aalst, W.M.P.: PM^2: a process mining project methodology. In: Zdravkovic, J., Kirikova, M., Johannesson, P. (eds.) CAiSE 2015. LNCS, vol. 9097, pp. 297–313. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19069-3_19
7. Leemans, S.J.J.: Robust Process Mining with Guarantees. Ph.d. thesis, Eindhoven University of Technology (2017)
8. Lorenz, R., Senoner, J., Sihn, W., Netland, T.: Using process mining to improve productivity in make-to-stock manufacturing. IJPR **59**(16), 1–12 (2021)
9. Park, J., Lee, D., Zhu, J.: An integrated approach for ship block manufacturing process performance evaluation: case from a Korean shipbuilding company. Int. J. Prod. Econ. **156**, 214–222 (2014)
10. Park, M., Song, M., Baek, T.H., Son, S.Y., Ha, S.J., Cho, S.W.: Workload and delay analysis in manufacturing process using process mining. In: Bae, J., Suriadi, S., Wen, L. (eds.) AP-BPM 2015. LNBIP, vol. 219, pp. 138–151. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19509-4_11
11. Pfeiffer, W., Weiß, E.: Lean Management: Grundlagen der Führung und Organisation lernender Unternehmen. Erich Schmidt Verlag GmbH & Co KG (1994)
12. Rozinat, A., de Jong, I., Günther, C., van der Aalst, W.M.P.: Process mining applied to the test process of wafer steppers in ASML. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. **39**(4), 474–479 (2009)
13. SAP: Multilevel bill of material (BOM) (2021). https://blogs.sap.com/2021/05/07/multilevel-bill-of-material-bom

14. Schuh, G., Gützlaff, A., Schmitz, S., van der Aalst, W.M.P.: Data-based description of process performance in end-to-end order processing. CIRP Ann. **69**(1), 381–384 (2020)
15. Uysal, M.S., et al.: Process mining for production processes in the automotive industry. In: BPM Industry Forum (2020)

Linac: A Smart Environment Simulator of Human Activities

Gemma Di Federico(✉), Erik Ravn Nikolajsen, Mamuna Azam, and Andrea Burattin

Technical University of Denmark, Kgs. Lyngby, Denmark
gdfe@dtu.dk

Abstract. The identification and construction of datasets of human activities is an extremely time-consuming and resource intensive task, yet researchers cannot refrain from such datasets. The publicly available datasets may not reflect all the researchers' requirements and are not scrupulously documented. In addition, these datasets can cope with just a limited and predefined set of behaviors. To address these challenges, we developed an instrument that allows to simulate the behavior of agents interacting with an environment. The environment is a customized configuration, equipped with sensors. The simulation generates as output a stream of events stemming from activated sensors. In addition, the agents behavior is not fully deterministic, so as to reflect the dynamic nature of human beings and to be as realistic as possible.

1 Introduction

In this work we describe *Linac*, a smart environment simulator. The simulator combines the non-deterministic behavior of human beings with a controlled simulation system, with the aim of generating data streams for research purposes. During the last decade there has been a notable diffusion of sensor systems. These systems allow the collection and analysis of data in real time, gaining the attention of researchers in the field of process mining [11,14]. As a result, the application of sensor systems has spread to many fields with the aim of collecting data, opening up the opportunity to derive new processes. One of the most attractive and innovative application is the derivation of processes related to human behavior [13], paving the way into the world of industry and healthcare [9]. In order to include human beings in process analysis, algorithms need to consider all the specific characteristics derived from this new application. To evaluate, extend, develop and test process mining algorithms, data is needed. There are several ways to collect real-life data, but they are expensive and time consuming. As well as there are scenarios that cannot be replicated (such as accidents or borderline situations). In addition, a thorough understanding of the underlying data as well as its underlying execution is required, and the most appropriate way to do the work is by generating *ad hoc* data (i.e., where the ground truth is known beforehand). The simulation and data generation instrument proposed in this paper allows the configuration of a custom and controlled

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 60–72, 2022.
https://doi.org/10.1007/978-3-030-98581-3_5

simulation. Different agents populate a smart environment, and the process to be carried out is defined by the user. The scenario is equipped with sensors. The simulation consists of the behavior of the agents interacting with the system.

The rest of the paper is organized as follows: existing solutions, the problems of testbeds, public datasets and simulators are presented in Sect. 2. In Sect. 3 we describe our proposed solution, and in Sect. 4 its implementation. In Sect. 5 an evaluation of the approach is presented, and Sect. 6 concludes the paper.

2 Existing Solutions

To evaluate process mining algorithms, data is needed [18]. In particular, when the process comprises the analysis of human behavior, the construction of a dataset becomes an extremely time-consuming and resource intensive task [20]. Three solutions can be considered: testbeds, public datasets and simulators.

The first option consists of constructing a physical testbed. Testbeds are physical environments equipped with sensors, controllers and network components, capable of capturing the state of the environment. Once the testbed has been constructed, a participant performs a predefined list of activities. The acquisition of such datasets is subject to limitations related to the cost and configuration of the actual environment. The layout of the environment must be carefully studied and verified, then all the necessary materials must be acquired, configured and installed. After the construction, the real execution process could start. This process is usually long-lasting, since participants need time to carry out the activities. Large datasets acquired with these techniques are therefore very complicated and expensive to obtain.

The second option to obtain data consists of using publicly available datasets. The main issue is to find a dataset that describes exactly the scenario needed. Additionally, the understanding of a dataset is limited to the documentation provided by authors. Since datasets are usually made up of thousands of events, it is hard to have a detailed description of their content. Consequently, it takes a long time to understand it. Orange4Home and CASAS are two datasets widely used in the literature for the analysis of routines and daily activities. The Orange4Home dataset [7] reports the activities of daily living in a smart apartment, for 4 weeks of recording. The dataset contains recording from 236 data sources. The log is not annotated, and the documentation is limited to the routine plan followed by the occupant during the experiment. The CASAS project [6] provides 66 different datasets, describing labeled and unlabeled activities performed in a smart environment such as assisted care apartments or box offices. The data refers to both single and multi-resident. The documentation provides the list of activities performed and the list of sensors. For the labeled datasets, authors do not specify how the activities were recognized. In addition to what has been said so far, datasets publicly available usually describe common scenarios, characterized by the execution of regular activities. However, we may be interested into processes related to extreme or anomalous cases both for a case study and to include all possible scenarios. Based on the experiment under investigation, there might be the need to examine specific cases that are difficult or expensive to replicate in reality or that may rise ethical concerns: an accident in an industry or an elderly person falling to the ground. Additionally, to verify the correctness of an algorithm we might want to evaluate it against as many cases as possible. A ready-to-use dataset hardly includes borderline cases, limiting the testing possibility.

The most effective way to obtain data is by using a simulator that replicates each specific use case. In this way we could have a complete control over the environment and the actions, a complete knowledge of the activities carried out and the data produced, i.e., the ground truth. Synonott et al. [19] distinguish between model-based and interactive approaches. Model-based approaches [3, 12, 17] consist of the specification of the reference model that the simulation should follow. The abstraction level in the definition of the activities describes the accuracy of the modeled behavior. Renoux et al. [17] propose a model-based approach in which inhabitants interact with a "sensorized" apartment. The user does not script out the actions that need to be carried out but rather provides an idea of how the simulated world works. The abstraction of the model does not allow to represent subtle but significant differences in the behaviors execution. To one hand they are suitable for long running simulations, on the other hand they do not provide clock simulation. Interactive approaches [2, 5] consist of a virtual environment (2D or 3D scenario) where the user can operate: the agent interacting with the environment is an "avatar" guided by the user who can interact with each individual sensor. The simulation is precise and realistic, since there is a real human behind the movements. However, the generation of large amounts of data is very expensive as it requires a great effort by the user: these approaches are suitable for testing single activities or short runs. An example is the work of Buchmayr et al. [5] which presents a 2D floor map equipped with sensors. The main objective of the tool is to generate and visualize sensors behavior, in particular the simulation of faulty or unexpected cases. The simulation is performed by the user interacting with the sensors via the mouse.

The majority of the simulators cannot be adapted to simulate several different environments. For example, it is challenging to represent a smart factory scenario with a 3D smart home simulator as it would become very complex in terms of dimensions and objects to have integrated [2]. Or it would be impossible to be able to replicate exact behaviors when the simulator only requires a general behavioral model as input [17]. The tools provided to draw the environment are often limited to the intended objective [3]; as well as they are not always oriented to the generation of datasets. Furthermore, there is a lack of simulators capable of reproducing, in a realistic way, the behavior of a human being [8] interacting with an environment. For this reason, we have developed a simulator capable of representing different environments and scenarios. The agent's behavior is the central focus of the tool. The behavior of the agent is as realistic as possible by introducing different walking speeds and non-deterministic movements. The process carried out by the agent is defined by the user, and it controls the expected result of the simulation.

3 Proposed Simulation Solution

To study processes related to human beings, process mining algorithms need to be evaluated. The data used in the evaluation phase should faithfully represent the reality and the human behavior. During the development of Linac, we have identified several key features that a simulator must consider. The most relevant is that human beings are flexible in their movements [8]: they do not perform movements in a fixed way but introducing variability. Furthermore, we cannot assume that human beings are all equal, i.e. elderly are slower in the movements, while young people are faster. Another factor that gains the attention of our analysis is the amount of data generated: sensors systems tend to produce large amount of data. In the following sections we explain how these challenges have been addressed.

3.1 Configuration of the Smart Environment

The simulation platform allows the configuration of a smart environment. The application is not limited to represent specific domains since it offers a blank canvas where to build up the floor plan, in form of a grid. The drawing tools are walls, entities and sensors. *Walls* are used to physically constrain the environment, while *entities* are objects that are part of the environment (that are not sensors). The agent can interact with these objects. An example of entity is a chair: the agent can move around such an object, but if she is instructed to go to it, then she can "stand" on top of it. Both sensors and entities have a physical and an interact area. Fig. 1a shows an example of floor plan designed as two rooms and two agents (agents represented with green tiles). The purple tiles are non-walkable sensors, while the blue tiles are entities. The walkable sensors are not shown on the map, but they can be inspected on the application. A pre-build selection of sensors is included, but new sensors could easily be defined as Java classes. *Sensors* can be active or passive. While active sensors are activated by the direct interaction by the agent, passive sensors are continuously running to detect changes in their statuses, producing an output at fixed time intervals. How the agent interacts with the sensor is defined via a command in the agent instructions. The trigger frequency of each sensor is configurable.

3.2 Configuration of the Agents' Behavior – AIL Language

The simulation consists of human beings moving and interacting with the objects of the smart environment. The simulation is carried out by one or more agents, each of them representing a person. Sensors and entities are shared among the agents. During the configuration it is possible to define a specific movement speed (i.e. meters per second) for each agent. The definition of the speed allows the simulation of the behavior of people with different ability levels. Furthermore, each agent has a specific set of activities to perform. The list of instructions is specified by the user during the configuration phase, using an application specific script language called Agent Instruction Language (AIL). AIL has been implemented in order to facilitate the definition of a list of activities to be performed. In fact, the definition of a long set of instructions is a time consuming and complex task. In addition, human behavior is characterized by the repetition of activities, always performed following the same set of actions. As a consequence, the list of instructions is composed by redundant code, e.g., the procedure for preparing a cup of tea will always be executed in the same way. To facilitate the task, we introduced primitive instructions that and can be grouped into macros. The AIL language comprises four primitive instructions which describe basic behaviors:

- **`goto`**`(x,y)`: instructs the agent to move to a specific position;
- **`goto`**`(name)`, where `name` refers to an entity or an active sensor: instructs the agent to move to a random tile in the interaction area of the entity/sensor;
- **`wait`**`(seconds)`: instructs the agent to remain stationary for the specified amount of seconds;
- **`interact`**`(activeSensor, command)`: instructs the agent to move to a random tile within the interact area of sensor `activeSensor`, and interact with it as specified by the command. The command of interaction is reflected in the data produced when the sensor triggers.

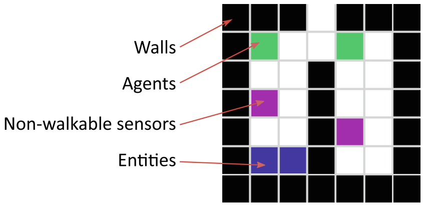
Macro instructions describe complex behaviors. These instructions include a sequence of primitive and/or macro instructions. Macros are a powerful tool to avoid errors, limit redundant code and establish groups of activities that form richer behavior. These instructions are defined as follows:

- **`macro`**`(m) {list of primitive instructions}`: defines the macro;
- `m()`: executes the macro `m`.

Figure 1b shows an example of primitives and macro. For each agent, a list of instructions is reported, and both agents share the macro called `makeTea`. Then the macro is then used only by agent `a1`. The language is designed to be intuitive: an external application can be used to automatically generate instructions starting from an ideal behavior.

3.3 Simulation Execution

The simulation models human behavior, which consists of movements and interactions with the objects. The movements, in turns, comprise journeys between the agent's current position and the target position. To find the path the agent must follow, we implemented a path-finding algorithm. To comply with the flexibility and stochasticity of human behavior, we extended the A* path-finding algorithm [10], constructing a sub-optimal and non-deterministic version of it. Actually, we started from an optimal and deterministic implementation of A*: since we defined the floor plan as a grid of tiles, it is easy to translate the grid of tiles into a graph of nodes needed for the A* algorithm. The A* algorithm uses a heuristic function to calculate a path between two nodes on a graph.



(a) Floor plan example

```
macro(makeTea) {
    interact(faucet,TURN_ON); wait(3);
    interact(faucet,TURN_OFF);
    interact(kettle,TURN_ON); wait(60);
}
agent(a1) {
    makeTea();
    goto(couch);
}
agent(a2) {
    goto(3,4);
    wait(100);
}
```

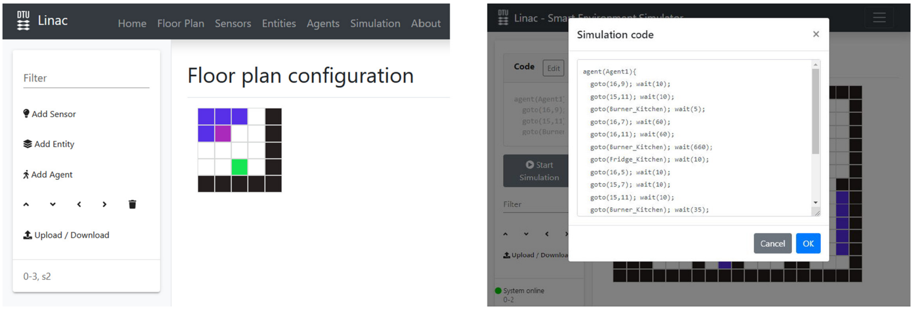
(b) AIL example

Fig. 1. A floor plan and a possible list of instructions for two agents

The heuristic function can be either admissible or inadmissible: in the first case it always calculates a distance that is shorter than or equal to the actual distance of reaching the goal node (optimal path); in the second case, it can calculate a distance that is longer than the actual distance of reaching the goal node (sub-optimal path). Furthermore, since neither the A* algorithm itself nor the commonly used heuristic functions contains a random variable, the calculated path is deterministic. However, a path-finding algorithm that is deterministic and optimal is not a good model for human movement: *(i)* a human does not take the same path every time between two points, *(ii)* a human does not take a random walk between two points and *(iii)* humans do not always take a shortest path between two points [4]. To tackle these problems, we defined an inadmissible heuristic function that includes a random variable. To obtain this heuristic function H, we considered the Euclidean distance between the two points plus a value $R \sim U(0, n \cdot L)$, randomly drawn from the uniform distribution between 0 and $n \cdot L$. In this case, L is the length of the sides of the tiles in the floor plan and n is a parameter indicating the degree of sub-optimality (the higher the value the less optimal the path). Adding the random variable to an otherwise admissible heuristic, overestimates the distance, thus making the heuristics inadmissible and hence sub-optimal. Furthermore, since it contains a random variable, it will be non-deterministic. We then experimented with increasing the threshold value n as much as we could, while avoiding the agent making too many counterproductive movements. Here we defined counterproductive movements as a move that leaves the agent at the same distance to the goal node or a longer distance away from the goal node.

3.4 Clock Simulation

A fundamental aspect of Linac is the clock simulation. Being able to run the simulation in real time is valuable when evaluating online algorithms and, on the other hand, not practical when simulating multiple days, as this would take multiple days in the real life as well. One way to lessen this impracticality is to scale how fast time progresses in the simulation relative to the time progression



```
agent(Agent1){
  goto(16,9); wait(10);
  goto(15,11); wait(10);
  goto(Burner_Kitchen); wait(5);
  goto(16,7); wait(60);
  goto(16,11); wait(60);
  goto(Burner_Kitchen); wait(660);
  goto(Fridge_Kitchen); wait(10);
  goto(16,5); wait(10);
  goto(15,7); wait(10);
  goto(15,11); wait(10);
  goto(Burner_Kitchen); wait(35);
```

Fig. 2. Screenshots of the Linac floor-plan page (left) and simulation (right)

in the reality. This can be achieved by defining how many real time seconds a simulated second should take.

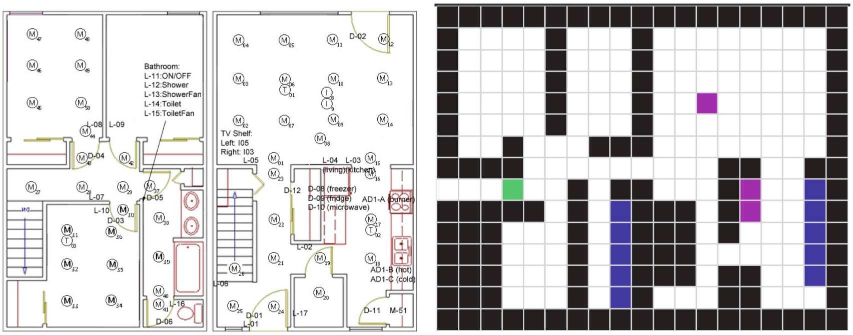
3.5 MQTT Output

The output produced by Linac is in the form of MQTT messages. The MQTT protocol [15] is based on a publish/subscribe model that decouples the publisher that sends the message from the subscribers that receive the message by the use of a broker. This architecture involves one or more publishing clients that publish messages under a topic. In our context each sensor would constitute a publishing client. Using this system, we could also focus only on a specific sensor by subscribing to a specific topic.

4 Implementation

Linac is implemented as a web application[1]. The application comprises a frontend and backend. The latter[2] is a server-side application implemented in Java. The former[3] is a web application implemented in TypeScript and Vue.js. The web application communicates with the server using a restful interface. The backend, in turns, exposes a set of APIs that can be triggered also from other applications. For example, a script can be used to generate and execute multiple simulations by programmatically generating the corresponding AIL code. The web application is organized in three main components: the floor plan, the sensors/entities/agents pages and the simulation page. A screenshot of the main page is shown in Fig. 2. The implementation allows the configuration and simulation of the environment. Once a simulation is running, it is possible to see the movements of the agents on the map in real time. Further details on the implementation are available on the report [16].

[1] See http://linac.compute.dtu.dk.
[2] Source code available at https://github.com/DTU-SPE/linac-backend.
[3] Source code available at https://github.com/DTU-SPE/linac-frontend.



(a) Floor plan of CASAS [1]

(b) Floor plan in Linac (floor sensors not visible)

Fig. 3. The two floor maps

5 Evaluation

To evaluate the behavior of the simulator, we decided to replicate an existing dataset, derived from a real scenario, and compare the results. The dataset chosen is one from CASAS. The dataset represents sensor events collected in a smart apartment testbed. The apartment has two residents performing their normal daily activities. The dataset provides both the raw and the annotated events, but we used the annotated one to recognize activities. Our evaluation comprises three phases: in the first phase we tried to replicate the floor plan, after that we analyzed the annotated dataset to identify how each activity has been performed, concluding with a running simulation. Being able to successfully replicate the CASAS dataset would allow us to show the capabilities of Linac in terms of realism of the data, thus allowing us to derive new datasets where specific situations or behaviors appear.

5.1 Configuration

Floor Plan Design. The Linac tool allows the configuration of an environment by means of a grid of variable size. On the grid, wall entities and sensors are distinguished by colors: black, blue, and purple respectively. The CASAS environment is composed of a 6 rooms apartment, equipped with more than 50 sensors (motion, item, door, water, temperature, electricity sensors). The floor plan was designed by transforming the sensors layout into the form of a grid. Then, all the sensors have been configured, and for each of them the physical area, the interaction area and the trigger frequency have been defined. Once the two maps matched, we moved on to the next phase. Figure 3 shows the two floor plans: Fig. 3a illustrates the original map provided by CASAS while Fig. 3b refers to the grid layout designed using Linac.

Table 1. Datasets structure and results comparison

	Duration		# of sensors		# of events	
	CASAS	**Linac**	**CASAS**	**Linac**	**CASAS**	**Linac**
Simulation1 (`Bed_to_Toilet`)	2 min	2 min	12	13	47	51
Simulation2 (`Meal_Preparation`)	19 min	19 min	14	14	360	440

Agent Instructions. The CASAS dataset is labeled, but no information regarding how the activities were carried out is provided. For this reason, we choose to focus only on two activities, rather than analyzing all the 13 proposed. The activities are `Bed_to_Toilet`, referred as Simulation1, and `Meal_Preparation`, named Simulation2. Starting from the list of triggered sensors in the CASAS dataset, we reconstructed the path followed by the agent. At this point, we drawn up a list of instructions that the agent had to follow to carry out the specific activity. A key feature of our simulator is the **`goto`** primitive statement which allows to instruct the agent to reach a specific tile, entity or active sensor, without having to provide coordinates for each step. In this way, it is easier to define the activity list. Once completed the list of instruction, we moved on the simulation phase.

The Simulation. The simulation tool of Linac allows for defining the date, the relative time and the configuration of MQTT. The choice of date and time let you to place the simulation at a specific moment in time. The simulation could be performed in real time speed or in a specific relative time, that is how many real seconds a simulated one should take. For this evaluation task, we performed the two simulations (`Bed_to_Toilet` and `Meal_Preparation`) on the same floor plan. The first simulation refers to a movement between two rooms, that is the path followed to go from the bed to the kitchen. The second refers to the activity of meal preparation. We used the relative time to run the simulations, which took less than 1 minute to execute. The data for both simulations is available for download[4].

5.2 Results

The four datasets (2 simulations × 2 datasets) are structured as reported in Table 1. For each couple of simulations, the durations are the same. The total amount of unique sensors activated differs in Simulation1 and this is caused by the non-deterministic path-finding algorithm implemented in Linac. In other words, the path that agents follow is characterized by a certain random variability, which led to the activation of an additional sensor. A gap could be observed in the number of events generated, especially in Simulation2. This discrepancy does not imply differences in the behavior: the index that causes this spread is the trigger frequency of each sensor. In Linac, each sensor has a fixed trigger frequency (for this simulation, configured to 5 seconds). In the CASAS dataset,

[4] See https://doi.org/10.5281/zenodo.5386318.

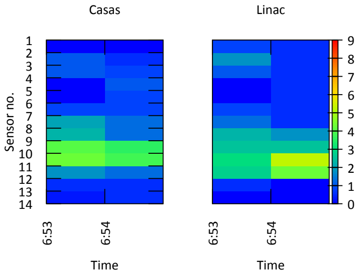


Fig. 4. Sensors triggered in Simulation1 in CASAS and Linac

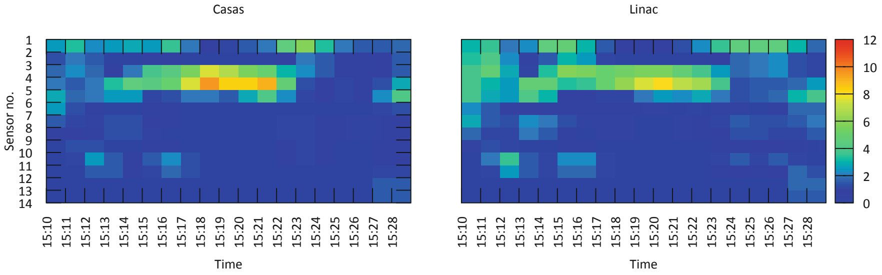


Fig. 5. Sensors triggered in Simulation2 in CASAS and Linac

on the other hand, this information is not provided. To better evaluate the differences between the two simulations, we plot them on heat maps.

The first map, reported in Fig. 4 refers to Simulation1. The maps show the number of times that each relevant sensor is triggered during the simulation period (with a time grouping of 1 min). As we can notice from the color variations, the two maps seem to behave very similarly over the same sensors/time. However, some discrepancies could be observed in the intensity color. Since the simulation is spread over just two minutes, and the number of events generated is small, we cannot consider the impact of sensors triggered only few times. In fact, the agent did not spend long periods in those zones, but was only passing through them, generating a single trigger for each sensor.

The second simulation lasts for 19 min, and this makes the heat maps in Fig. 5 more accurate. In fact, the average amount of triggers for each sensor is higher than the previous simulation. Therefore, as the simulation lasts longer, the total number of events generated is much greater. Looking at the maps we have to consider that the values on the x-axes are one minute units. Therefore, there could be sensors activated a minute before or a minute after others (e.g. 15:11:58 and 15:12:01), which could be considered misalignment in the graphs but, for simulation purposes, are absolutely tolerable. Both the maps in Fig. 5 intensify in colors in the time interval 15:15–15:23, suggesting that the main behaviors occurred during that interval.

To better evaluate the resulting simulations, we computed the Pearson's coefficient for the two scenarios. The Pearson's coefficient is a measure of the strength of a linear correlation between two variables. We computed the correlation for each sensor, and then we calculated the average value for all sensors. Both Simulation1 and Simulation2 resulted in a coefficient of 0.93. These results depict a strong correlation between the sensors activation during the simulations. Therefore, we can state that, in both simulations, the behavior of the simulated sensors (by Linac) and the behavior of the reference sensors (CASAS) agree and hence we can conclude that Linac is capable of effectively mimicking a real dataset.

The objective of the simulations is to replicate the behavior of human beings (in CASAS) as movements of agents (in Linac) through a simulation. The two simulations conducted, and the resulting evaluation, highlighted that Linac is able to replicate a real behavior such as that collected in the CASAS dataset.

6 Conclusions and Future Works

We presented a smart environment simulator for the generation of datasets, in the form of streams. The simulator could be used to configure different environments, thanks to its structure made up of walls, entities and sensors. The behavior of the agents is composed of movements inside the environment and of interactions with entities and sensors. The behavior is dictated by means of a list of instructions that the agent must follow and that can be described using the language AIL, that we created for this purpose. The simulation uses a non-deterministic sub-optimal algorithm to replicate the stochasticity of human behavior. The simulator, additionally, offers the functionality for setting the speed of movement for each agent. A simulated clock is used to solve problems related to the long running of the simulations. The clock is fully configurable, and the emulation consists of running real simulations but in which time passes much faster. The last aspect to be summarized is the output that uses the MQTT protocol to stream the data, that is, sensors readings.

The behavior of the Linac simulator has been compared with a dataset describing the behavior of an actual person inside a testbed environment. The analysis gave very positive results suggesting that Linac is able to reproduce the same movements.

All things considered, it can be said that the Linac simulator is suitable for the generation of realistic datasets referring to the human behavior. The data generated can be used to test and evaluate algorithms, thus resulting in a valuable tool for researchers.

Aspects to improve comprise the definition of the library of sensors available and the extension of the AIL language. For example, considering the `wait` statement, used to instruct the agent to remain stationary for a certain period, there are cases in which we want the agent should not remain exactly in the same tile, but randomly move nearby. These aspects could contribute towards an ever higher level of realism.

References

1. CASAS - Daily Life Spring 2009. http://casas.wsu.edu/datasets/twor.2009.zip
2. Alshammari, N., Alshammari, T., Sedky, M., Champion, J., Bauer, C.: OpenSHS: open smart home simulator. Sensors **17**(5), 1003 (2017)
3. Ariani, A., Redmond, S.J., Chang, D., Lovell, N.H.: Simulation of a smart home environment. In: ICICI-BME 2013, pp. 27–32 (2013)
4. Banovic, N., Buzali, T., Chevalier, F., Mankoff, J., Dey, A.K.: Modeling and understanding human routine behavior. In: Proceedings of CHI, pp. 248–260 (2016)
5. Buchmayr, M., Kurschl, W., Küng, J.: A simulator for generating and visualizing sensor data for ambient intelligence environments. Procedia **5**, 90–97 (2011)
6. Cook, D.J., Crandall, A.S., Thomas, B.L., Krishnan, N.C.: CASAS: a smart home in a box. Computer **46**(7), 62–69 (2012)
7. Cumin, J., Lefebvre, G., Ramparany, F., Crowley, J.L.: A dataset of routine daily activities in an instrumented home. In: Ochoa, S.F., Singh, P., Bravo, J. (eds.) UCAmI 2017. LNCS, vol. 10586, pp. 413–425. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67585-5_43
8. Di Federico, G., Burattin, A., Montali, M.: Human behavior as a process model: Which language to use? ITalian forum on Business Process Management (2021)
9. Seoane, F., Traver, V., Hazelzet, J.: Value-driven digital transformation in health and medical care. In: Fernandez-Llatas, C. (ed.) Interactive Process Mining in Healthcare. HI, pp. 13–26. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-53993-1_2
10. Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE SMCS **4**(2), 100–107 (1968)
11. Janisch, C., et al.: The internet-of-things meets business process management. A manifesto. IEEE Syst. Man Cybern. Mag. **6**(4), 34–44 (2020)
12. Kormányos, B., Pataki, B.: Multilevel simulation of daily activities: why and how? In: CIVEMSA, pp. 1–6. IEEE (2013)
13. Leotta, F., Mecella, M., Mendling, J.: Applying process mining to smart spaces: perspectives and research challenges. In: Persson, A., Stirna, J. (eds.) CAiSE 2015. LNBIP, vol. 215, pp. 298–304. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19243-7_28
14. Mandal, S., Hewelt, M., Oestreich, M., Weske, M.: A classification framework for IoT scenarios. In: Daniel, F., Sheng, Q.Z., Motahari, H. (eds.) BPM 2018. LNBIP, vol. 342, pp. 458–469. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11641-5_36
15. Naik, N.: Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. In: ISSE, pp. 1–7 (2017)
16. Nikolajsen, E.R., Azam, M.: A platform to simulate agent interactions with IoT devices to facilitate process mining algorithm research. Technical Report, DTU (2021). https://findit.dtu.dk/en/catalog/2691894192
17. Renoux, J., Klugl, F.: Simulating daily activities in a smart home for data generation. In: WSC 2018, pp. 798–809 (2018)
18. Rozinat, A., De Medeiros, A., Günther, C., Weijters, A., van der Aalst, W.: Towards an evaluation framework for process mining algorithms. BPMcenter.org (2007)
19. Synnott, J., Nugent, C., Jeffers, P.: Simulation of smart home activity datasets. Sensors **15**(6), 14162–14179 (2015)
20. Vinciarelli, A., et al.: Open challenges in modelling, analysis and synthesis of human behaviour in human-human and human-machine interactions. Cogn. Comput. **7**(4), 397–413 (2015)

Root Cause Analysis in Process Mining with Probabilistic Temporal Logic

Greg Van Houdt[1](✉), Benoît Depaire[1], and Niels Martin[1,2]

[1] Research Group Business Informatics, UHasselt - Hasselt University, Hasselt, Belgium
{greg.vanhoudt,benoit.depaire,niels.martin}@uhasselt.be
[2] Research Foundation Flanders (FWO), Brussels, Belgium

Abstract. Process mining is a research domain that enables businesses to analyse and improve their processes by extracting insights from event logs. While determining the root causes of, for example, a negative case outcome can provide valuable insights for business users, only limited research has been conducted to uncover true causal relations within the process mining field. Therefore, this paper proposes AITIA-PM, a novel technique to measure cause-effect relations in event logs based on causality theory. The AITIA-PM algorithm employs probabilistic temporal logic to formally yet flexibly define hypotheses and then automatically tests them for causal relations from data. We demonstrate this by applying AITIA-PM on a real-life dataset. The case study shows that, after a well-thought-out hypotheses definition and information extraction, the AITIA-PM algorithm can be applied on rich event logs, expanding the possibilities of meaningful root cause analysis in a process mining context.

Keywords: Process Mining · Root Cause Analysis · Probabilistic Temporal Logic · Event Log

1 Introduction

Process mining is a research domain that enables businesses to analyse and improve their processes by extracting insights from event logs [1]. The foundation is the event log, which records the real execution of a business process. It can then be used for, among other goals, process discovery [2] and conformance checking [6]. However, merely discovering how a process is actually executed and where it differs from the normative model might not be sufficient. Insights in, for example, why an event was triggered or why a trace ended with an exception can be of more interest to business users, and thus, accurate root cause analyses (RCA) are desired.

Identifying root causes can be a complex task [17]. Each process involves many different steps, and for each step many factors can be of influence. Add to this that many traces in a business process can show unique behaviour, as well as

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 73–84, 2022.
https://doi.org/10.1007/978-3-030-98581-3_6

influence each other by having to share resources. Previous research has proposed techniques to conduct RCA in process mining, e.g. [7,10,11], however, there are clear limitations. First, they often put forward a correlation analysis instead of a true RCA. However, when a process characteristic is correlated with a particular undesirable outcome, this does not imply that this characteristic caused the phenomenon. In that sense, one must acknowledge confounding factors can exist, which might cause spurious associations to arise [24]. Second, existing RCA techniques that build upon causality theory impose heavy assumptions on the underlying data. Think of only being able to handle linear causal relations, for example.

Against this background, this paper proposes the AITIA-PM algorithm. This algorithm is a new way of executing an RCA in process mining, inspired by the work of Kleinberg [13,14]. Not only is AITIA-PM based on causality theory, this technique does not impose assumptions on the required data, making it more reliable in the real world. We propose the use of probabilistic temporal logic (PTL) to formally define hypotheses about causal relations, which offers great flexibility. Additionally, we explicitly take confounding factors into account. As such, AITIA-PM is a new addition to the current state-of-the-art of meaningful RCA in process mining. Our contributions are best summarised as follows:

– We propose a novel method in AITIA-PM, adding a new technique to the mix for effective root cause analysis in the process mining domain which is fully based on existing causality theory.
– The demonstration on a real-life event log shows the value of AITIA-PM, mainly found in the flexibility of PTL when identifying specific causal relations and how statistical significance can be computed. It also shows the importance of a theoretical foundation regarding the philosophy surrounding causality, as results are easy to interpret.

The remainder of this paper is structured as follows. Section 2 describes the related work in root cause analysis from a process mining standpoint, after which Sect. 3 introduces the AITIA-PM algorithm which is employed in the demonstration as discussed in Sect. 4. Finally, we conclude our paper in Sect. 5.

2 Related Work

An RCA is not bound to a specific family of techniques. Examples are (i) classification techniques as seen in, for example, [3,8,10,22,23], and (ii) rule mining algorithms like association rules [5] and subgroup discovery [19]. Unfortunately, in most applications, there is too little attention given towards the differentiation between correlation and causality.

Hompes et al. [11] proposed a graph-based approach resulting in a time series analysis to detect cause-effect relations by testing for Granger causality [9], thus explicitly considering causation instead of correlation between features. However, it is not perfect either. Granger causality, as it is originally defined, cannot account for instantaneous or nonlinear causal relations, and cannot deal with confounding effects either. Also, Granger causality makes strong assumptions on the underlying data which are rarely met in the real world [15].

Finally, Qafari and van der Aalst have recently published research on structural equation models for RCA [17] which was later extended with counterfactual reasoning [18]. One of the foundations here is that the structure of causal relations can be provided by the domain expert if available and, as such, there can be no discussion about causality or correlation. The counterfactual reasoning extension allows the authors to produce recommendations that indicate how specific cases could have been handled differently to avoid problems in the future [18]. However, the authors acknowledge that using a machine learning technique imposes the risk of obtaining wrong or imprecise recommendations, or even miss out on the correct ones, regardless of the model's accuracy. Narendra et al. [16] also show how to answer the what-if questions via structural causal models and counterfactual reasoning, proving the effectiveness of the methods, yet they acknowledge it lacks intuitiveness.

The causality measure and complementary algorithm introduced by Kleinberg [13,14] pays great attention towards determining causality by building on the philosophical foundations of causality theory [12,21]. To that end, the algorithm is able to detect the genuine causal relations from data separate from spurious ones. This is achieved by implementing probabilistic temporal logic (PTL) for defining hypotheses, which are then tested based on probability theory and statistical significance. Additionally, Kleinberg's technique explicitly tackles confounding variables.

3 The AITIA-PM Algorithm

As described in Sect. 2, Kleinberg's work found its basis in causality theory. The measure and complementary algorithm allow for extraction of causal relations from data rather than a predefined model of how a system evolves in terms of states it is in. AITIA-PM tailors the ideas of Kleinberg to the process mining field. The following paragraphs describe the necessary background followed by a step-by-step guide of the algorithm. For more information, we refer the reader to Kleinberg [13].

3.1 Background

The Concept of Causality. In this paper, consistent with the work of Kleinberg [13], the following properties must hold to establish a causal relationship between a cause and an effect: (i) the cause must precede the effect in time [12] and (ii) a cause must raise the probability of the effect [21]. Property (ii) is also known as the prima facie condition. Several pitfalls must be taken into account, however.

First of all, there might be causality without raising the probability of the effect or vice versa. For example, yellow stained fingers and lung cancer can be the result of a common earlier cause: smoking. Without considering smoking, one would observe that having yellow stained fingers would increase the probability of lung cancer. However, when holding the common cause fixed, that relationship between the effects would disappear. Controlling for common causes is known as screening off, or dealing with confounding factors [24].

Second, event logs carry a case notion. However, process instances can influence each other. Think of resources being shared or scarce materials suddenly becoming unavailable because the last item was just consumed, thus impacting how a different case can continue. Therefore, we add another property to AITIA-PM one must meet, namely that (iii) each case is defined by the events which can possibly be a cause of the effect within that specific case.

Clearly, unlike the heavy assumptions made in Granger causality which are, among others, that there is no confounding variable present, causal relations are linear and time series are stationary [15], our understanding of causality imposes less restrictions on the input data. The first two properties, as will be made clear in the following subsections, are also easy to infer from an event log automatically, making inference practically feasible as well.

Probabilistic Temporal Logic. PTL allows reasoning on the likelihood of an event within a certain time interval. For example: how likely is it that a train arrives at the station within 2 to 10 min. As such, properties should not hold *eventually*, as they are bound in time so it can be quantified how likely it will happen. By allowing to freely define the cause, effect, type of relation between cause and effect, and the time window, PTL is highly flexible in execution.

AITIA-PM uses PTL as language to define the hypotheses the business user desires to test for cause-effect relations. Each hypothesis comprises a logical formula describing both the time bounds as well as the likelihood of a potential cause c triggering an effect e: $c \leadsto^{\geq r, \leq s}_{\geq p} e$. This is also called a leads-to formula where r, s represent the time bounds and p the minimum probability for the cause triggering the effect in the time window in order for the formula to evaluate to true. c and e here are state formulas: properties which hold for the system at a certain point in time. Such a property can be an activity that was executed. For example, with $\neg H$ and F being not doing homework and failing a test respectively, $\neg H \leadsto^{\geq 1, \leq 3}_{\geq 0.40} F$ would describe that when a student neglects the necessary homework, the probability of the student failing a test between 1 and 3 time units would be at least 40%. From the practical viewpoint of AITIA-PM, the probabilities are calculated from data and do not need to be passed by the user.

The state formulas for the cause and effect are not limited to contain one element each. PTL allows for each state formula to be a path formula too. A path formula can express properties along a path (or trace) in the dataset. For example, a path formula can be that an activity B must follow activity A in a trace within 5 time units, like so:

$$[AF^{\leq 5}_{\geq p_1} B] \leadsto^{\geq r, \leq s}_{\geq p_2} e \tag{1}$$

where F represents the path operator *Finally*, indicating that at some state of the path the property will hold, and p_1 being the probability that B should follow A within 5 time units. The evaluation of such a path formula in itself is also a state formula which is true at a certain moment in time for the trace. Having defined such state and path formulas, one knows which information to extract from the event log to employ as system states. These system states, along with their case notions and timestamps, then serve as input for the algorithm.

AITIA-PM uses only a subset of PTL by, for example, neglecting the notion of time windows. We do so because long-term dependencies in business processes need to be acknowledged. The interested reader is referred to [13] for more details about PTL.

3.2 Algorithmic Procedure

AITIA-PM guides the user in detecting meaningful root causes supported by causal theory. It consists of the following five steps: (i) input data preparation, (ii) generating causal hypotheses, (iii) testing for prima facie causes, (iv) calculation of epsilon values, and (v) testing for causal significance.

Step 1 – Input Data Preparation. The AITIA-PM algorithm focuses on *system states* and how they change over *time* for each *case* in the event log. As such, these are the three required attributes in the input data structure. The definition of the system states depends on the potential causes and effects the business user is interested in, and thus, has defined in PTL hypotheses. For example, let's assume that we know that when resource x (R_x) is involved in a case, the case will result in an error (E). In other words, you define your hypothesis as

$$R_x \leadsto E. \tag{2}$$

Remember that the probability of this leads-to formula actually occurring is inferred from data in a later stage. Given this hypothesis, the data analyst knows which system states to extract from or enrich the event log with: the resources involved with the case at each time unit, and whether or not the error E was registered. As such, the input data consists of these three columns: the case ID, the system state, and the timestamp.

One can also opt to convert all timestamps in the data set to a specific time unit, where the first observation in the event log would start at time unit 0. This would easily allow the reintroduction of time windows in PTL leads-to formulas.

Step 2 – Generating Hypotheses. Having defined the system states, one can now generate the different hypotheses: which causes might have a significant impact on the likelihood of the effect triggering? AITIA-PM takes a list of plausible causes and effects to combine them into the complete set of hypotheses: does cause c trigger effect e within the time bounds $[r, s]$? All combinations are considered a hypothesis except where $c = e$.

In this step, it is important to consider adding all system states as a possible cause for the effect of interest. This way, you also check for the other states as potential confounding factors, even though you might not expect them to have a causal relationship with the effect. In the example of R_x triggering an error E, a hypothesis will be generated for every resource R_r with $r \in R$ to trigger the effect E.

Step 3 – Testing for Prima Facie Causes. The hypotheses generated before contain all combinations of cause-effect we are interested in. However, they probably also describe causal relations which might not meet the prima facie condition. In order for a cause to be a prima facie cause of an effect, it must satisfy the following three conditions:

1. the cause must have occurred before the effect,
2. the cause must increase the probability of the effect occurring, and
3. the cause and effect when checking the above requirements must belong to the same case in the event log.

With the timestamps and case IDs provided along with the system states, it is relatively straightforward to determine whether or not a cause is a prima facie cause for an effect from the event log. Only the hypotheses fulfilling the above requirements are considered to be genuine potential causes for the effect.

In order to accomplish this prima facie test, the following pieces of information are required: (i) when and for which case was the cause observed, (ii) when and for which case was the effect observed, and (iii) how often did the effect occur after the cause given they both belong to the same case. The prima facie condition is then probabilistically checked from the data as follows:

$$P(e|c) > P(e) \tag{3}$$

where

$$P(e) = \frac{\#e}{\#events} \tag{4}$$

and

$$P(e|c) = \frac{\#(e \wedge c)}{\#c}. \tag{5}$$

It is important to remember that $\#(e \wedge c)$ takes the timing of events and case ID into account. This computation therefore checks if there exists a c before e within the same case, and if not, the hypothesis is automatically classified as false. For example, resource R_y is only involved after the case already produced error E. As such, $P(E|R_y) = 0$, meaning that R_y cannot be a prima facie cause of E.

Step 4 – Calculation of Epsilon Values. Having determined all prima facie causes of the effect of interest, we now want to separate the genuine causes from the spurious ones. To that end, we use epsilon values as a measure of causality that can be statistically tested. The measure ϵ_{avg}, introduced by Kleinberg [13], describes the average change of probability of effect e given the presence of cause c while keeping another factor x constant. This factor x is also a prima facie cause of e which is deemed to be present. As such, for each other factor x, an ϵ_x is calculated after which the average describes the impact of c on e.

Formally, the measure is then expressed as follows:

$$\epsilon_{avg}(c,e) = \frac{\sum_{x \in X \backslash c} \epsilon_x(c,e)}{|X \backslash c|} \tag{6}$$

where X represents the set of prima facie factors of e and

$$\epsilon_x(c,e) = P(e|c \wedge x) - P(e|\neg c \wedge x). \tag{7}$$

Determining these probabilities correctly requires that the case notion is identical for pairs of e, c and x. While keeping x constant, the probability change of e is of interest when the cause c is present or not. Property (iii) of causality in AITIA-PM dictates that all information regarding causal relationships within a case is available in that same case. As such, the case ID must be identical for c and x when counting the occurrences of $(c \wedge x)$ and $(\neg c \wedge x)$.

The probabilities are defined as follows:

$$P(e|c \wedge x) = \frac{\#(e \wedge c \wedge x)}{\#(c \wedge x)} \tag{8}$$

and

$$P(e|\neg c \wedge x) = \frac{\#(e \wedge \neg c \wedge x)}{\#(\neg c \wedge x)} \tag{9}$$

where e must occur at a later time than $(c \wedge x)$ or $(\neg c \wedge x)$. As soon as this information is available, it is a simple matter of counting how often an effect does or does not take place in the related time windows. For each hypothesis that passed the prima facie test, an ϵ_{avg} is obtained. These average epsilons are the foundation of the statistical test performed next.

Step 5 – Determining Causal Significance. Up until this point, the epsilon values are computed, which express the average probability changes of the effect e occurring given the presence or absence of a prima facie cause c. A statistical test can then separate the genuine causes from the spurious ones. To that end, the AITIA-PM algorithm uses the concept of false discovery rates (FDR) as implemented by the R-package `fdrtool` [20]. Saving the technical details, the procedure is as follows:

1. start by calculating z-values: $z = (\epsilon_{avg} - \mu)/\sigma$ where μ and σ represent the average and the standard deviation of the set of ϵ_{avg}, respectively;

2. Next, fit a mixture model to the observed data, the z-values;
3. Determine the FDR of z.

The causal relations where the FDR is below a certain threshold are deemed significant causes. This threshold is chosen freely by the business user depending on how acceptable a false discovery is. For example, with a threshold of 0.01, one would expect 1% of causes to be significant.

4 Demonstration

In this section, we demonstrate how AITIA-PM learns causes for process delay by applying it on a real-life dataset, namely the "receipt phase of an environmental permit application process (WABO) CoSeLoG project" event log [4][1]. This event log contains the receiving phase execution records of the building permit application process in an undisclosed Dutch municipality. It consists of 1.434 traces and 8.577 events spread over 27 activity classes.

Similar to Qafari and van der Aalst [17], we consider as effect the delay observed in some cases. This delay threshold is set to 3% of the maximum duration of all traces. As the maximum duration is 275.8813 days, the threshold is equal to 8.2764 days, or 198.6345 h. As the average duration of a trace is about 2% of the maximum duration, the threshold of 3% seems appropriate. We add a new event "Case Delayed" to each case that exceeds the threshold duration at the moment the case reaches a duration of 198.6345 h. This ensures that events occurring after that moment in time can no longer be considered a cause for the delay in that case. As Qafari and van der Aalst [17], we investigate if the combination of a specific activity A_i performed by a specific resource R_j causes process delay.

Remember the five steps of AITIA-PM: (1) data preparation, (2) generating causal hypotheses, (3) testing for prima facie causes, (4) calculation of epsilon values, and (5) testing for causal significance. Steps 1 and 2 both relate to the PTL hypothesis definition. In our example, an initial set of 397 hypotheses is constructed as there are 397 distinct activity-resource pairs in the event log. Each hypothesis for a specific activity A_i and a specific resource R_j can be described with PTL as follows:

$$A_i \wedge R_j \rightsquigarrow delay \tag{10}$$

Consequently, the system states to extract from the event log are all the activities per case with the associated resource that executed them. The first ten rows of the input dataset are shown in Table 1, along with the first observation of process delay.

All initial 397 hypotheses were tested for the prima facie condition (step 3), and 159 of these passed the test, meaning they occurred before the delay

[1] The source code and data to reproduce the results of the demonstration are available at https://github.com/gregvanhoudt/AITIA-PM.

Table 1. Input data for AITIA-PM.

Case ID	System State	Time Unit
case-891	Confirmation of receipt - Resource26	0.0000000
case-891	T02 Check confirmation of receipt - Resource26	0.0131450
case-891	T03 Adjust confirmation of receipt - Resource26	0.1759917
case-891	T02 Check confirmation of receipt - Resource26	0.1835817
case-891	T03 Adjust confirmation of receipt - Resource26	0.1894819
case-3756	Confirmation of receipt - Resource02	71.2025831
case-3756	T06 Determine necessity of stop advice - Resource02	71.3695931
case-3756	T02 Check confirmation of receipt - Resource24	72.1805186
case-3756	T07-1 Draft intern advice aspect 1 - Resource24	72.1995269
case-3756	T06 Determine necessity of stop advice - Resource02	72.3097125
...	...	...
case-891	Case Delayed	198.6345127
...	...	...

was observed and they increase the probability of the case being delayed. After computation of the test statistics and setting the FDR threshold to 5%, we obtain output as shown in Table 2.

Table 2. AITIA-PM output.

cause	epsilon	z	fdr
T02 Check confirmation of receipt - Resource24	0.1871651	7.444724	0.0000000
T04 Determine confirmation of receipt - Resource10	0.1202274	4.364844	0.0258531
T05 Print and send confirmation of receipt - admin1	0.0736255	2.220631	0.0258531

In summary, AITIA-PM detects that, with the FDR threshold set to 0.05, three of the 159 hypotheses are genuine. It appears that the probability of the case being delayed significantly increases when specifically (i) "T02 Check confirmation of receipt" is executed by Resource24, (ii)"T04 Determine confirmation of receipt" is executed by Resource10, or (iii) "T05 Print and send confirmation of receipt -" is executed by Admin1. We can be most sure of (i), as that FDR value is equal to zero and its epsilon value is also the highest.

This epsilon is also easy to interpret. In the case of our first result, this interpretation is as follows: the average increase in probability of the effect, the case delay, occurring when the activity "T02 Check confirmation of receipt" is executed by Resource24 while controlling for alternative causal explanations equals 18.71651 pp..

5 Conclusion

This paper introduced a novel root cause analysis method in process mining named AITIA-PM. It complements the state-of-the-art with respect to RCA techniques as it follows causality theory. Unlike already established techniques, AITIA-PM imposes realistic assumptions regarding the required data. This makes it a very adaptable technique to the desires of a business user. Additionally, by taking a probabilistic approach and averaging out the probability changes, the technique can easily tackle confounding factors which could cause spurious associations. This makes it a strong novel option for RCA.

The demonstration shows that AITIA-PM can flexibly tap into the vast amount of information an event log possesses. PTL allows very diverse hypotheses to be tested which makes AITIA-PM both powerful but also expressive. Due to PTL it is easy to define both simple as well as more complex hypotheses with respect to cause-effect relations in a formal manner. Finally, we have shown the strength of AITIA-PM with respect to interpretability of results.

Several future research challenges are identified in this article. First, a domain expert is required to provide the necessary states the process can semantically be in. Automatic hypothesis generation could bring insights the domain expert might not even consider. Second, state formulas in their current form are binary as they evaluate to true or false. Future work could bring an extension which supports continuous variables.

Acknowledgements. A special thanks goes to prof. dr. S. Kleinberg for sharing the source code of the original AITIA algorithm.

This research was funded by the UHasselt BOF under grant number BOF19OWB19.

References

1. van der Aalst, W.M.P.: Process Mining: Data Science in Action. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
2. van der Aalst, W.M.P., Weijters, T., Maruster, L.: Workflow mining: discovering process models from event logs. IEEE Trans. Knowl. Data Eng. **16**(9), 1128–1142 (2004)
3. Bozorgi, Z.D., Teinemaa, I., Dumas, M., La Rosa, M., Polyvyanyy, A.: Process mining meets causal machine learning: discovering causal rules from event logs. In: 2020 2nd International Conference on Process Mining (ICPM), pp. 129–136 (2020)
4. Buijs, J.: Receipt phase of an environmental permit application process (‘WABO’), CoSeLoG project. Eindhoven University of Technology (2014). https://data.4tu.nl/articles/dataset/Receipt_phase_of_an_environmental_permit_application_process_WABO_CoSeLoG_project/12709127

5. Böhmer, K., Rinderle-Ma, S.: Mining association rules for anomaly detection in dynamic process runtime behavior and explaining the root cause to users. Inf. Syst. **90** (2020). Advances in Information Systems Engineering Best Papers of CAiSE 2018
6. Carmona, J., van Dongen, B., Solti, A., Weidlich, M.: Conformance Checking. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99414-7
7. Delias, P., Lagopoulos, A., Tsoumakas, G., Grigori, D.: Using multi-target feature evaluation to discover factors that affect business process behavior. Comput. Ind. **99**, 253–261 (2018)
8. Ferreira, D.R., Vasilyev, E.: Using logical decision trees to discover the cause of process delays from event logs. Comput. Ind. **70**, 194–207 (2015)
9. Granger, C.W.: Some recent development in a concept of causality. J. Econom. **39**(1–2), 199–211 (1988)
10. Gupta, N., Anand, K., Sureka, A.: Pariket: mining business process logs for root cause analysis of anomalous incidents. In: Chu, W., Kikuchi, S., Bhalla, S. (eds.) DNIS 2015. LNCS, vol. 8999, pp. 244–263. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16313-0_19
11. Hompes, B.F.A., Maaradji, A., La Rosa, M., Dumas, M., Buijs, J.C.A.M., van der Aalst, W.M.P.: Discovering causal factors explaining business process performance variation. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 177–192. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_12
12. Hume, D.: A Treatise of Human Nature (1739)
13. Kleinberg, S.: Causality, Probability, and Time. Cambridge University Press, Cambridge (2012)
14. Kleinberg, S., Kolm, P.N., Mishra, B.: Investigating causal relationships in stock returns with temporal logic based methods (2010)
15. Maziarz, M.: A review of the granger-causality fallacy. J. Philos. Econ. Reflections Econ. Soc. Issues **8**(2), 86–105 (2015)
16. Narendra, T., Agarwal, P., Gupta, M., Dechu, S.: Counterfactual reasoning for process optimization using structural causal models. In: Hildebrandt, T., van Dongen, B.F., Röglinger, M., Mendling, J. (eds.) BPM 2019. LNBIP, vol. 360, pp. 91–106. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26643-1_6
17. Qafari, M.S., van der Aalst, W.: Root cause analysis in process mining using structural equation models. In: Del Río Ortega, A., Leopold, H., Santoro, F.M. (eds.) BPM 2020. LNBIP, vol. 397, pp. 155–167. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-66498-5_12
18. Qafari, M.S., van der Aalst, W.M.P.: Case level counterfactual reasoning in process mining. In: Nurcan, S., Korthaus, A. (eds.) CAiSE 2021. LNBIP, vol. 424, pp. 55–63. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-79108-7_7
19. Fani Sani, M., van der Aalst, W., Bolt, A., García-Algarra, J.: Subgroup discovery in process mining. In: Abramowicz, W. (ed.) BIS 2017. LNBIP, vol. 288, pp. 237–252. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59336-4_17
20. Strimmer, K.: fdrtool: a versatile R package for estimating local and tail area-based false discovery rates. Bioinformatics **24**(12), 1461–1462 (2008)
21. Suppes, P.: A probabilistic theory of causality. Br. J. Philos. Sci. **24**(4), 409–410 (1973)
22. Suriadi, S., Ouyang, C., van der Aalst, W.M.P., ter Hofstede, A.H.M.: Root cause analysis with enriched process logs. In: La Rosa, M., Soffer, P. (eds.) BPM 2012. LNBIP, vol. 132, pp. 174–186. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36285-9_18

23. Vasilyev, E., Ferreira, D.R., Iijima, J.: Using inductive reasoning to find the cause of process delays. In: 2013 IEEE 15th Conference on Business Informatics, pp. 242–249. IEEE (2013)
24. Vogt, W.P., Johnson, B.: Dictionary of Statistics & Methodology: A Nontechnical Guide for the Social Sciences. Sage, Thousand Oaks (2011)

xPM: A Framework for Process Mining with Exogenous Data

Adam Banham(✉), Sander J. J. Leemans, Moe T. Wynn, and Robert Andrews

Queensland University of Technology, Brisbane, Australia
adam.banham@hdr.qut.edu.au

Abstract. Process mining facilitates analysis of business processes using event logs derived from historical records of process executions stored in organisations' information systems. Most existing process mining techniques only consider data directly related to process execution (endogenous data). Data not directly representable as attributes of either events or traces (which includes exogenous data), are generally not considered. Exogenous data may be used by process participants in making decisions about execution paths. However, as exogenous data is not represented in event logs, its impact on such decision making is opaque and cannot currently be assessed by existing process mining techniques. This paper shows how exogenous data can be used in process mining, in particular discovery and enhancement techniques, to understand its influence on process decisions. In particular, we focus on time series which represent periodic observations of e.g. weather measurements, city health alerts or patient vital signs. We show that exogenous time series can be aligned and transformed into new attributes to annotate events in an event log. Then, we use these attributes to discover preconditions in a Petri net with exogenous data (xDPN), thus revealing the exogenous data's influence on the process. Using our framework and a real-life data set from the medical domain, we evaluate the influence of exogenous data on decision points that are non-deterministic in an xDPN.

Keywords: Process mining · Decision mining · Petri nets with data · Context awareness · Time series data

1 Introduction

Process mining is a field that uses historical event data extracted from an organisation about a business function (process) to better understand its behaviour and performance [1]. Process mining techniques rely on a single 'source of truth', an event log containing process instances (traces), and a sequence of events (the "what" happened and "when" it happened) for each process instance.

Process discovery techniques [13,25] exploit event sequences presented in an event log to recreate the structure of a business process. Conformance techniques [3,4] use a process model and an event log to create aligned event

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 85–97, 2022.
https://doi.org/10.1007/978-3-030-98581-3_7

sequences that follow the possibilities described in the process model. Enhancement techniques [1,20] enrich a process model with additional influences such as the performance or the resource utilisation of events.

In our work, we use the following definitions to distinguish between data that can, and can not be represented in event logs effectively. We define *endogenous* data as data internal to a process, meaning they have a direct link to a specific process's progress towards its goal. For example, endogenous data could include: the time that an event occurred, the resource which performed the activity, any information needed to perform the activity or the cost of completing the activity.

In contrast, we define *exogenous* data as data external to a process, meaning that they are not tied to a specific process, but record contextual data. For example, exogenous data could be the temperature and humidity readings inside a food delivery truck, or periodic readings from a sensor monitoring a patient's heart rate, or the noise levels in an employee's work space. The purpose in recording exogenous data is to describe the context as clearly as possible over time, meaning that records are taken as frequently as possible (i.e., time series) rather than more selective point-in-time recordings usually associated with endogenous data. While data-aware or context-aware techniques exist, such as techniques presented in [20,24] or [25], we have not found any studies which use exogenous data in conjunction with these techniques.

In this paper, we study the potential of exogenous data to improve our understanding of complex decision points in processes. In particular, we focus on a particular type of exogenous data, i.e., numerical time series. We proposed a novel process mining framework, *xPM*, that translates exogenous time series data and links them to relevant events in an event log for automated process discovery and enhancement. A data-aware process discovery technique can then be used to discover a process model in which the decision points are annotated with preconditions using exogenous data. Finally, an enhancement step which visualises related exogenous data for transitions on a process model is envisioned. We instantiated xPM and evaluated the influence of exogenous data on the quality of the discovered process model using a real-life data set from the medical domain.

The remainder of the paper is organised as follows: Sect. 2 outlines related work. Section 3 defines the preliminaries. Section 4 presents xPM. Section 5 discusses the evaluation, and Sect. 6 concludes the paper.

2 Related Work

Categorisation of data sources used to describe businesses has been discussed in several studies. The 'onion skin' model in [22] conceptualises the relationship between data and process as the viewpoint is moved further away from a process. This conceptualisation is then applied to process mining in [2], where data is categorised according to the likelihood of cause and effect between variables with the process. However, these frameworks are not seen as essential to process mining in recent reviews of the field, and the contextual component remains an optional consideration during event data extraction [7,9,21]. Our contribution is that we support separate entities for endogenous and exogenous data sources such that they can studied separately or in combination.

The benefits of including a variety of data categories are discussed in [16] which (i) motivates the use of data attributes for distinguishing between noise and conditional behaviour, (ii) considers if data attributes influence decision points by creating an internal state as the process executes through boolean expressions and decision trees, and (iii) studied how alignments [3] can be extended such that they balance both the control flow perspective and the data perspective. The techniques described in [16] were implemented using Petri Nets with Data (DPN) modelling language (see [6,10,16] for a complete definition).

Our study extends the concepts presented in [16] and shows how exogenous data can be incorporated (instead of being limited to only the endogenous perspective of an event log). In particular, we focus on extending guard conditions in DPNs to include external factors not represented in the endogenous event log.

Methodologies that encourage contextual data collection and log enrichment are few in number. However, some recent studies have focused on the enrichment of an event log with new types of data. In [23], the authors present a framework for intra- and inter-trace predictive monitoring and introduce the notion of *bi-dimensional coding* to deal with intra- and inter-trace dependencies. In [8], the authors suggest that not all events within an event log are about the control flow, and are instead, about the data flow of a process. They use the concept of *context events* to deal with the two types of events and show how distinguishing between the two can lead to less complex discovered models. However, this approach would incorporate exogenous context into the control flow perspective instead of clarifying whether the context influences process execution.

The benefits of having additional data attributes that can be seen in the recent evolution of techniques using such data, such as [14,25]. In [14], the authors present a discovery algorithm that uses data attributes to create a hierarchical model to improve the simplicity of outcomes. Another approach in [25], was to create a constraint operator for process trees notation, whereby data semantics can be expressed. While these techniques can create control flow sequences based on data attributes, no extensions have been proposed to use exogenous sources outside what can be found in the events within an event log.

3 Preliminaries

This section introduces event logs, Petri nets, xDPNs and exogenous data sets.

Event Logs. The execution of each process step can be recorded as an event. An end-to-end execution of a process is called a *trace*. A trace is a sequence of events $\langle e_1, \dots e_n \rangle$. An event log is a collection of such traces. Both traces and events can have attributes to store data.

Exogenous Data Sets. A *time series* is a sequence of timestamped values $\langle m_1^{@t_1} \dots m_i^{@t_i} \rangle$ for measures m_i and timestamps t_i. For example,

$\langle 22\,°\mathrm{C}_1^{01\text{-}01\text{-}2021}, 15\,°\mathrm{C}_2^{02\text{-}01\text{-}2021}, 10\,°\mathrm{C}_3^{03\text{-}01\text{-}2021}\rangle$. A time series can have attributes and uses the notation of a trace to describe the ith measurement of a time series. A collection of time series describing the same exogenous context is an *exogenous data set*. For example, a collection of time series for wind speed, where each series is recording wind speeds for a local government area is an exogenous data set.

Labelled Petri Nets. A *Petri net* is a triple $N = (P, T, F)$, where P is a finite set of places, T is a finite set of transitions such that $P \cap T = \emptyset$ and $F \subset (P \times T) \cup (T \times P)$ is a set of directed arcs, called a flow relation [1]. A *labelled Petri net* is a quintuple, $(P, T, F, \Sigma, \lambda)$, where (P, T, F) is a Petri net, Σ is a set of observed activity names and λ is a event labelling function $T \rightarrow \Sigma$ [1]. Places may hold tokens, which are produced and consumed when transitions fire according to the flow relation. A transition is *enabled* if each input place contains a token. The state of a Petri net is a *marking*, which records what places have tokens and how many. An enabled transition l can *fire*, which updates the marking according to the flow relation F and, if l is labelled by λ, denotes the execution of activity $\lambda(l)$. An initial marking denotes the initial state of a Petri net before the first transition is fired.

Petri Nets with Exogenous Data (xDPN). A *precondition* is a boolean expression describing a subset of values for attributes (e.g. temperature is higher than 20 °C). A *Petri Net with Exogenous Data* (xDPN) is a sextuple $(P, T, F, \Sigma, \lambda, \Phi)$, where $(P, T, F, \Sigma, \lambda)$ is a labelled Petri net and $\Phi : T \rightarrow \phi$ associates a transition with a precondition. A transition is *data enabled* if the precondition attached to a transition is satisfied by the current assignment of attributes or if there is no attached precondition. In an xDPN, a transition can fire if it is enabled and data enabled. The state of an xDPN is described by a marking, and an endogenous and exogenous data state. An xDPN is a sub-formalism of DPN (a complete formalisation of DPN can be found in [6,10,16]): in contrast to xDPN, DPN consider distinctions between attributes states (e.g. `read` or `written`). Furthermore, xDPNs do not enforce that transitions in the model update variable assignments, allowing exogenous data attributes to be updated during execution.

4 A Framework for Process Mining with Exogenous Data

In this section we introduce xPM, which considers how exogenous data can be used by process mining techniques. Figure 1 shows an overview of xPM. xPM takes as input an event log and a collection of exogenous data sets ($\mathcal{X}$). xPM uses a number of quadruples $(x, \mathcal{L}, \mathcal{S}, \mathcal{T})$, where $x \in \mathcal{X}$ is an exogenous data set; $\mathcal{L}$ is a linking function, which links traces to exogenous time series that are relevant for that trace; $\mathcal{S}$ is a slicing function, which, for each event, returns sub-time series relevant for that event; and $\mathcal{T}$ is a transformation function, which summarises each sub-time series into a set of transformed attributes. For each such quadruple, xPM annotates each event (that has non-empty sub-time series) with their exogenous sub-time series and transformed attributes, creating a *exogenous-annotated*

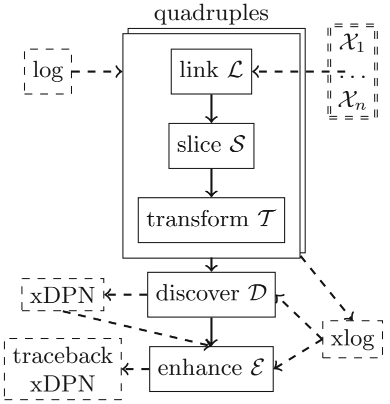


Fig. 1. The xPM framework.

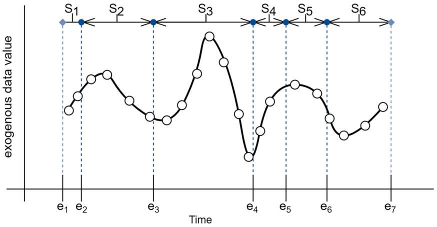


Fig. 2. Visualisation of exogenous data sliced by $\mathcal{S}$ to create sub-time series s_i. Each e_i is annotated with s_{i-1}.

log (xlog). Next, a discovery function $\mathcal{D}$ discovers an xDPN. Finally, an enhancement step $\mathcal{E}$ aligns a log and a xDPN. Then for each aligned event, we *trace back* from the exogenous transformed attribute to sub-time series. Finally, $\mathcal{E}$ visualise the subset of exogenous data set relevant for each transition (*traceback* xDPN).

4.1 Linking

The first step of xPM is to find a subset of an exogenous data set related to each trace using a linking function $\mathcal{L}$. This linking function can consider many different aspects of a trace when creating this subset, and it may also consider if a trace has or has not been linked to other exogenous data sets. For example, an event log could be capturing how an insurance company handles claims. Then, an exogenous data set could capture time series of weather predictions for local government areas. An $\mathcal{L}$ would link the time series in this data set to claims. To create a subset of time series, $\mathcal{L}$ might compare the location of a claim and the location of weather predictions. However, a more complex $\mathcal{L}$ could find adjacent government areas and interpolate between weather predictions to predict if an extreme weather event will likely occur.

In case an $\mathcal{L}$ links two or more time series to a particular trace, this $\mathcal{L}$ must merge these time series into a single time series. Simply combining all time series onto a single timeline is insufficient as multiple values could be recorded at a timestamp. Handling this case is not trivial and will require a thorough understanding of exogenous data or domain knowledge. As such, in this paper we limit the scope of exogenous data to time series of numerical data and limit $\mathcal{L}$ to link only one time series to each trace. We acknowledge that this a simplified view of exogenous data and does not account for all types of exogenous data possible; an extension of xPM could consider how an additional internal step could compress larger subsets into a single time series.

4.2 Slicing

The second step of xPM is to annotate events with relevant exogenous data. That is, events will be annotated with the sub-time series using a *slicing* function $\mathcal{S}$. Figure 2 illustrates an example of a simple slicing function: each event e_i of a trace $\langle e_1, \ldots e_n \rangle$ is annotated with the sub-time series between the previous event e_{i-1} and e_i.

More elaborate slicing functions could use a process model to ignore concurrent events when determining the previous event, or to only annotate events relevant to decision points in the model. Other possibilities include taking a fixed time window, for instance, for an event, taking the past two days of rainfall measurements to watch for flash flooding. Another possible slicing algorithm could use knowledge of activity instances (i.e. start and completion events) in order to create sub-time series observed during a execution of an activity.

These examples are not exhaustive; however, we highlight the potential of creating an extensive array of slicing algorithms to suit the needs of an analyst. Domain knowledge then informs the choice of slicing functions; assisting this choice is an interesting area of further research.

4.3 Transformation

Next, a transformation function, $\mathcal{T}$, transforms a sub-time series for an event into attributes and annotates the event with these attributes. Each new attribute created in this way for a event is referred to as a *transformed* attribute. The $\mathcal{T}$ function needs to provide a name for each attribute it creates (which can be trivially met by adding a suffix to the exogenous data set's name). Furthermore, transformations should reference sub-time series by an identifier so that outcomes that use transformations can be traced back to the original sub-time series for further analysis.

We identified three forms that a $\mathcal{T}$ can take: (i) $\mathcal{T}$ can return a single value to annotate an event; such a transformation might return the minimum, maximum or mean of a sub-time series; (ii) $\mathcal{T}$ can return a set of attributes to annotate an event; such a transformation might be the nth Taylor polynomial of the sub-time series, with each of the necessary coefficients; (iii) $\mathcal{T}$ can be recursive, which applies several recursions in order to meet either case (i) or (ii). Such a transformation finds the nth derivative of the sub-time series (where the sub-time series is a continuous function) then applies any previously mentioned functions.

4.4 Discovery

The output of several quadruples $(x, \mathcal{L}, \mathcal{S}, \mathcal{T})$ is an event log, with some events annotated with (i) sub-time series and (ii) transformed attributes. We refer to such an event log as an *exogenous-annotated* log (xlog). To this xlog, a discovery function $\mathcal{D}$ is applied. This study only considers $\mathcal{D}$ functions that use data-aware discovery techniques to obtain a process model with preconditions for transitions using the transformed attributes. Examples of such techniques are [17] and [19]. Preconditions found by these techniques do not create boolean expressions between `written` or `read` and as such can be translated into an xDPN. Furthermore, when discovering preconditions using these techniques, the attributes that have been set by preceding events inform the discovery. As data leading up to the event is not considered, future decision mining techniques capable could handle differences in exogenous and endogenous attributes.

4.5 Enhancing

As the final step of xPM an enhancement step $\mathcal{E}$ visualises the sub-time series from an xlog using the outcome of $\mathcal{D}$ to highlight points of interest. To create a connection between the events in an exogenous-annotated log and a discovered xDPN, we need to use process conformance techniques to find alignments. While data-aware alignments exist (e.g. [4,16]), [4] only considers the writing of attributes by transitions (and not whether preconditions hold) and [4,16] correct the data written by transitions using Integer Linear Programming. In contrast, in our context, exogenous data should not be adjusted in conformance techniques as it occurs outside the internal process execution. Therefore, to verify whether preconditions are met, our approach first computes alignments [3], after which we verify preconditions separately.

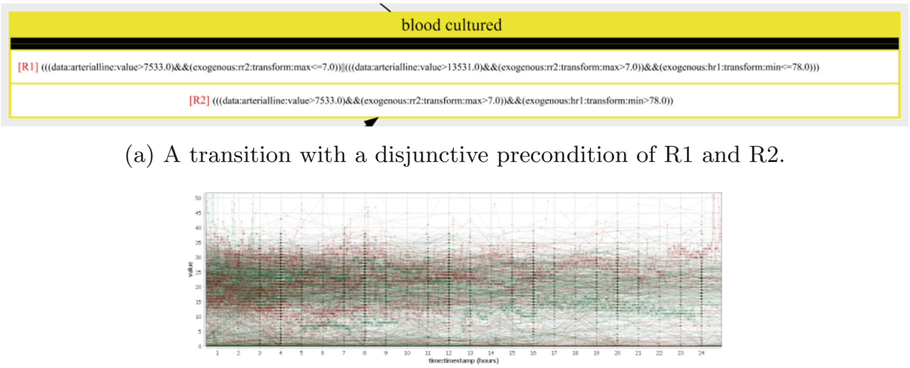
Given that the alignments proposed in [3] do not consider the data perspective, we present following example of $\mathcal{E}$ which uses alignments. First, an alignment between all traces and an xDPN is computed. Then for each aligned transition in the xDPN, we collect the most recent sub-time series in preceding events and plot all series from the same exogenous data set on a graph. Then we consider the type of alignment move that occurred in the alignment for that transition. If we see a synchronous move and this transition has a precondition, we check the following. (1) If the precondition was satisfied then sub-time series related to the aligned event of this move is plotted in green. (2) If the precondition was not satisfied then sub-time series related to the aligned event of this move is plotted in red. (3) Otherwise – e.g. a non-synchronous move – we plot the related sub-time series in black. Figure 3 is an example of such a visualisation, which has been implemented in a ProM plugin, Exogenous Data.

5 Evaluation

In this section, we instantiate xPM presented in Sect. 4. Then we evaluate, using two event logs from a real-life data set in the medical domain and existing DPN discovery techniques, the influence of exogenous data on the quality of the discovered xDPNs.

5.1 Procedure

We used the event logs either (i) as an event log with endogenous data attributes (*endo*), (ii) as an event log with exogenous attributes where endogenous attributes have been removed (*exo*), and (iii) as an event log with both endogenous and exogenous attributes (*endo+exo*).



(a) A transition with a disjunctive precondition of R1 and R2.

(b) An exogenous data set's (`RR2`) sub-time series for Figure 3a.

Fig. 3. An example of $\mathcal{E}$ for a transition, showing exogenous sliced time series.

Our instantiation of xPM is as follows:

$\mathcal{L}$ For each exogenous data set, a linking function was defined that linked data sets to the patient of the trace and that occurred during the admission.

$\mathcal{S}$ We included two slicing functions. Let $\langle e_1 \ldots e_n \rangle$ be a trace. Then, for event e_i the first slicing function (S_1) finds sub-time series between events e_{i-1} and e_i, while the second slicing function (S_2) finds the sub-time series between e_1 and e_i.

$\mathcal{T}$ We included four transformation functions: minimum, average, maximum and the cumulative sum of a Fourier transform[1] [11].

$\mathcal{D}$ To discover a control-flow model, we applied the Inductive Miner - infrequent [13] with path filtering of 0.25. To discover an xDPN, we applied two Data Petri Net discovery techniques: Mutually Exclusive Decision Tree (`dt`) [5] and Overlapping Rules Decision Tree (`or`) [19].
These techniques each take a parameter *min instances* (mi) that sets the minimum level of observed decision point instances that support a clause in a precondition. We repeated the experiment for $mi \in \{0.05, 0.15, 0.25\}$.

$\mathcal{E}$ was not part of this experiment.

Thus, in total, 18 xDPNs were discovered for each of the two logs. A visual breakdown of our instantiation can be seen in Fig. 4.

5.2 Quality Measures

We assessed the quality of the discovered xDPNs using fitness, precision and determinism. For fitness, we used balanced multi-perspective conformance checking [18]. For precision, we used the multi-perspective precision [16].

[1] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.stft.html#scipy.signal.stft.

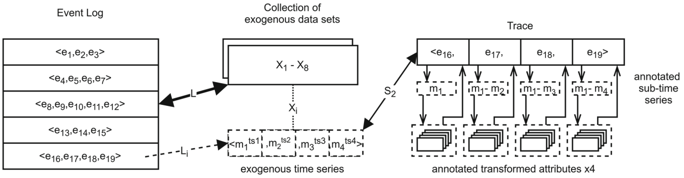

Event Log
$<e_1,e_2,e_3>$
$<e_4,e_5,e_6,e_7>$
$<e_8,e_9,e_{10},e_{11},e_{12}>$
$<e_{13},e_{14},e_{15}>$
$<e_{16},e_{17},e_{18},e_{19}>$
L
L_i
Collection of exogenous data sets
X_1 - X_8
X_i
$<m_1^{ts1}, m_2^{ts2}, m_3^{ts3}, m_4^{ts4}>$
exogenous time series
S_2
Trace
$<e_{16},$
$e_{17},$
$e_{18},$
$e_{19}>$
m_1
m_1- m_2
m_1- m_3
m_1- m_4
annotated sub-time series
annotated transformed attributes x4

Fig. 4. Visual breakdown of quadruples applied in xPM.

For determinism, we propose the following measure, which expresses the decision points in the model that are deterministic. That is, a fraction of places in the model with more than two outgoing arcs (decision points) that have at least one outgoing arc to a transition that has *no* precondition. Formally, let $N = (P, T, F)$ be a Petri net.

$$\text{decision points or } dp(P) = \{p \mid p \in P \wedge p^\bullet \geq 2\} \tag{1}$$

$$\text{weight or } w(p) = \frac{|\{t \mid t \in p^\bullet \wedge \varPhi(t) \neq \perp\}|}{|p^\bullet|} \tag{2}$$

$$\text{Determinism or } D(P) = \begin{cases} \frac{\sum_{\{p \in dp(P)\}} w(p)}{|dp(P)|} & \text{if } |dp(P)| > 0 \\ 1 & \text{otherwise} \end{cases} \tag{3}$$

A D value of 1 implies that all transitions that are involved in choices in the model have preconditions, while a value of 0 indicates that no transition that is involved in a choice has a precondition.

5.3 Event Logs and Exogenous Data

The data for our experiments is derived from the MIMIC-III data set [12]. MIMIC-III records patient demographics, admissions, ward stays, clinical observations, labs, imaging, prescriptions, caregiver notes, etc., for over forty thousand patients who stayed in critical care units between 2001 to 2012.

We created two event logs: a log of patient movements (*movements log*) and a log of procedures for respiratory failures (*procedures log*). The extraction scripts for these two event logs can be found in this repository[2]. The *movements log* captures the movements of patients between ICU wards within a single hospital admission, and contains 24 271 traces, 290 462 events, 65 activities and 6 endogenous attributes. The *procedures log* captures a process which describes the procedures that a patient received during a single hospital admission and contains 65 traces, 610 events, 34 event classes and 4 endogenous attributes. Both logs have 8 exogenous data sets (respiratory rate, 3x heart rate, 2x oxygen saturation, 2x arterial blood pressure). The *movements log* has 25 684 680 exogenous data points; the *procedures log* has 590 285 exogenous data points.

[2] https://github.com/adamBanham/icpm2021.

Table 1. Experimental Results.

			Movements log			Procedures log		
Variant	$\mathcal{D}$	*mi*	fit.	pre.	*D*	fit.	pre.	*D*
endo	dt	0.25	0.586	0.644	0.048	0.739	0.395	0.119
		0.15	0.573	**0.656**	0.108	0.739	0.392	0.119
		0.05	0.573	0.644	0.140	0.761	0.418	0.167
	or	0.25	0.586	0.657	0.048	**0.785**	0.393	0.131
		0.15	0.583	**0.656**	0.108	**0.785**	0.393	0.131
		0.05	0.587	0.647	0.140	0.761	0.419	0.167
endo +exo	dt	0.25	0.575	0.591	0.079	0.692	0.409	0.155
		0.15	0.518	0.537	0.156	0.705	0.411	0.155
		0.05	0.465	0.533	**0.283**	0.717	0.459	0.238
	or	0.25	0.586	0.649	0.048	0.731	0.445	0.214
		0.15	0.536	0.559	0.156	0.739	0.430	0.179
		0.05	0.465	0.550	0.259	0.717	0.463	0.238
exo	dt	0.25	**0.654**	0.640	0.000	0.722	0.413	0.143
		0.15	**0.654**	0.623	0.000	0.697	0.436	0.143
		0.05	0.173	0.567	0.222	0.709	0.420	0.214
	or	0.25	**0.654**	0.628	0.000	0.701	**0.512**	**0.274**
		0.15	**0.654**	0.639	0.000	0.697	0.425	0.143
		0.05	0.099	0.582	0.198	0.709	0.424	0.214

5.4 Results and Discussion

Table 1 shows the results. The best results for each log appear in **boldface**. When considering the *movements log*, using exogenous data only (exo) does not introduce preconditions in most cases, and henceforth the fitness and precision values are high. In cases where it does introduce preconditions, fitness is very low but precision is competitive. We conclude that for this log, the exogenous data by itself does not suffice. For exo+endo, typically more preconditions are discovered, which lowers fitness and precision (at most 0.11 lower than endo). This is to be expected, as adding more preconditions to the xDPN means that multi-perspective measures will consider more data attributes from the event log, thus increasing the state space on which precision is based.

When considering the *procedures log*, surprisingly, larger values of the parameter *mi* did not always decrease the number of preconditions found (*D*, or, endo+exo) as is to be expected as *mi* is a support threshold. We suspect that the rather small size of the procedures log and the nature of the overlapping rules (`or`) algorithm is at play here, which after building a first precondition, there is not enough observations left for a second precondition to meet the *mi* threshold. For this log, using the exogenous data increased the determinism and hence the number of preconditions found (exo+endo and exo vs. endo). Consequently, for endo and endo+exo, fitness goes up with *mi* for `dt` and goes down for `or`, but for exo these patterns are not there. If we consider exo and endo+exo vs. endo, then fitness consistently decreases, precision consistently increases, and determinism consistently increases. We suspect that the preconditions cover a larger fraction of the increased state space than for the *movements log*.

A possible extension of our analysis would be to understand if cohort analysis [15], separating patients into distinct care groups, would change the efficacy of our approach, allowing us to consider if observations in *procedures log* can be seen in medically relevant cohorts of patients.

6 Conclusion

In previous studies, exogenous data has been undeveloped when considering guard conditions in DPNs. As such, exogenous data's influence on process participants and decision-making in a process execution has not been considered in depth. This paper presents xPM, a framework for using exogenous data in process mining techniques that does not limit analysis opportunities. xPM allows for complex analysis of exogenous data and process executions, using existing process mining techniques and increased traceability – by means of slicing functions – between events and exogenous data. We evaluated the influence of exogenous data on process model discovery by measuring the difference in process model quality. Our evaluation showed that we could understand more decision points by including exogenous data and can improve fitness.

We see several extensions in future work. The semantics of xDPN could be expanded to introduce ways of expressing exogenous data sets alongside the process execution rather than solely within preconditions. Other data-aware process mining techniques could be used instead of the proposed techniques in our instantiation, such as [14,25]. Decision mining techniques for discovering preconditions could be extended to consider if an transformed attribute or exogenous data set correlates with the process activity before discovering a precondition. A variety of visualisation for the enhancement step could exist, and analysis could be expanded to consider more than satisfaction of a precondition to creating new modes of engagement with domain experts.

References

1. van der Aalst, W.M.P.: Process Mining. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4_16
2. van der Aalst, W.M.P., Dustdar, S.: Process mining put into context. IEEE Internet Comput. **16**(1), 82–86 (2012)
3. Adriansyah, A.: Aligning observed and modeled behavior. Ph.D. thesis, Technische Universiteit Eindhoven (2014)

4. de Leoni, M., van der Aalst, W.M.P.: Aligning event logs and process models for multi-perspective conformance checking: an approach based on integer linear programming. In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 113–129. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40176-3_10
5. De Leoni, M., van der Aalst, W.M.P.: Data-aware process mining. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing - SAC 2013. ACM (2013)
6. de Leoni, M., Felli, P., Montali, M.: A holistic approach for soundness verification of decision-aware process models. In: Trujillo, J.C., et al. (eds.) ER 2018. LNCS, vol. 11157, pp. 219–235. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00847-5_17
7. De Smedt, J., vanden Broucke, S.K.L.M., Obregon, J., Kim, A., Jung, J.-Y., Vanthienen, J.: Decision mining in a broader context: an overview of the current landscape and future directions. In: Dumas, M., Fantinato, M. (eds.) BPM 2016. LNBIP, vol. 281, pp. 197–207. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58457-7_15
8. Dees, M., Hompes, B., van der Aalst, W.M.P.: Events put into context (EPiC). In: ICPM. IEEE (2020)
9. Diba, K., Batoulis, K., Weidlich, M., Weske, M.: Extraction, correlation, and abstraction of event data for process mining. WIREs Data Mining Knowl. Disc. **10**(3), e1346 (2019)
10. Felli, P., De Leoni, M., Montali, M.: Soundness verification of decision-aware process models with var.-to-var. conditions. In: 2019 19th International Conference on ACSD. IEEE (2019)
11. Griffin, D., Lim, J.: Signal estimation from modified short-time Fourier transform. IEEE Trans. Acoust. Speech Signal Process. **32**(2), 236–243 (1984)
12. Johnson, A.E., et al.: MIMIC-III, a freely accessible critical care database. Sci. Data **3**(1), 1–9 (2016)
13. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from event logs containing infrequent behaviour. In: Lohmann, N., Song, M., Wohed, P. (eds.) BPM 2013. LNBIP, vol. 171, pp. 66–78. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06257-0_6
14. Leemans, S.J.J., Goel, K., van Zelst, S.J.: Using multi-level information in hierarchical process mining: balancing behavioural quality and model complexity. In: 2020 2nd ICPM. IEEE (2020)
15. Leemans, S.J.J., Shabaninejad, S., Goel, K., Khosravi, H., Sadiq, S., Wynn, M.T.: Identifying cohorts: recommending drill-downs based on differences in behaviour for process mining. In: Dobbie, G., Frank, U., Kappel, G., Liddle, S.W., Mayr, H.C. (eds.) ER 2020. LNCS, vol. 12400, pp. 92–102. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-62522-1_7
16. Mannhardt, F.: Multi-perspective process mining. Ph.D. thesis, Technische Universiteit Eindhoven (2018)
17. Mannhardt, F., De Leoni, M., Reijers, H.A.: The multi-perspective process explorer. In: BPM Conference Demos, vol. 1418, pp. 130–134. CEUR-WS (2015)
18. Mannhardt, F., de Leoni, M., Reijers, H.A., van der Aalst, W.M.P.: Balanced multi-perspective checking of process conformance. Computing **98**(4), 407–437 (2015). https://doi.org/10.1007/s00607-015-0441-1

19. Mannhardt, F., de Leoni, M., Reijers, H.A., van der Aalst, W.M.P.: Decision mining revisited - discovering overlapping rules. In: Nurcan, S., Soffer, P., Bajec, M., Eder, J. (eds.) CAiSE 2016. LNCS, vol. 9694, pp. 377–392. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39696-5_23
20. Mannhardt, F., de Leoni, M., Reijers, H.A., van der Aalst, W.M.P.: Data-driven process discovery - revealing conditional infrequent behavior from event logs. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 545–560. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_34
21. Marquez-Chamorro, A.E., Revoredo, K., Resinas, M., Del-Rio-Ortega, A., Santoro, F.M., Ruiz-Cortes, A.: Context-aware process performance indicator prediction. IEEE Access **8**, 222050–222063 (2020)
22. Rosemann, M., Recker, J., Flender, C.: Contextualisation of business processes. Int. J. Bus. Process Integr. Manage. **3**(1), 47–60 (2008)
23. Senderovich, A., Di Francescomarino, C., Maggi, F.M.: From knowledge-driven to data-driven inter-case feature encoding in predictive process monitoring. Inf. Systems **84**, 255–264 (2019)
24. Senderovich, A., Weidlich, M., Gal, A.: Context-aware temporal network representation of event logs: model and methods for process performance analysis. Inf. Systems **84**, 240–254 (2019)
25. Shraga, R., Gal, A., Schumacher, D., Senderovich, A., Weidlich, M.: Inductive context-aware process discovery. In: 2019 1st ICPM. IEEE (2019)

A Bridging Model for Process Mining and IoT

Yannis Bertrand(✉), Jochen De Weerdt, and Estefanía Serral

Research Centre for Information Systems Engineering (LIRIS), KU Leuven,
Warmoesberg 26, 1000 Brussels, Belgium
{yannis.bertrand,jochen.deweerdt,estefania.serral}@kuleuven.be

Abstract. Contextualisation is an important challenge in process mining. While Internet of Things (IoT) devices are collecting more and more data on the physical context in which business processes are executed, the IoT and process mining fields are still considerably disintegrated. Important concepts, such as *event* or *context*, are not understood in the same way, which causes confusion and hinders cooperation between the two domains. Based on IoT ontologies and business process context models, this paper proposes a model to bridge the conceptualisation gap between the IoT and the process mining fields. The model defines the necessary concepts and relationships to build process mining techniques that take the physical context into account. As a first validation, the model is used to describe a lifelike process example, showing how IoT data and process events are related. Using this conceptualisation, both practitioners and researchers from the IoT and the process mining communities can reason about the use of IoT data in process mining and find support for data understanding, event abstraction and IoT and process data integration.

1 Introduction

Although the potential of IoT data for process mining (PM) has been recognised, the relationships between IoT data and event logs has not been made explicit yet. This lack of deeper knowledge about these relationships, at the conceptual level, is part of a more general conceptual issue in PM boiling down to the question: what is an event? Previous works by different researchers have identified various conceptions of "event", which differ on their semantic level, e.g. micro events, high- versus low-level events, etc.; on their "scope", e.g. context events or process/control-flow events; or on whether the event includes the data associated with it, as in XES [9], etc. In addition to this, the same kind of conceptual challenges arise when using IoT data to retrieve the context of a process, as the understanding of context differs in the fields of IoT and PM.

These conceptual issues causes practical problems in PM. Process models discovered from event logs at an inadequate semantic level, i.e., too detailed or too coarse, can be too complex or too simple, which often make them unpractical and unfaithful to the reality. Then, confusing process and context events can also have an impact on the resulting process model, by either omitting activities of

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 98–110, 2022.
https://doi.org/10.1007/978-3-030-98581-3_8

the process, or over-complexifying the model. An example of this can be found in Dees et al. [4], who showed that translating three types of events of the Sepsis dataset [14] into context events, reduced the number of discovered variants to one-fourth, while annotating the model with information on these context events made the model as informative as a model considering all events as process events. Employing IoT data in PM can cause both types of problems: models at an unsuitable level of granularity, or models that confuse the context and the control-flow, as IoT data have to be abstracted to the adequate semantic level (i.e. that of the process) and data on the control-flow of the process have to be carefully distinguished from data on the context of the process.

The goal of this paper is to discuss the important concepts of event and context in PM, highlighting the difference in their understanding in the domains of IoT and PM. Using these concepts, we propose a model defining the links between these concepts and between the IoT and PM conceptual views, based on IoT ontologies, context models from business process management (BPM) and PM data models. The rest of the paper is structured as follows. Section 2 reviews the existing literature, focusing on IoT ontologies and business process (BP) context models. Next, in Sect. 3, the ambiguities in some important concepts are analysed. Section 4 presents a conceptual model defining and linking important concepts of IoT and PM. After this, a use-case of the model is presented in Sect. 5, and a comparison with some related works is done in Sect. 6 Finally, Sect. 7 provides a brief conclusion with some propositions for future works.

2 Background

In this section, previous works on the modelling of IoT and PM are introduced. First, relevant IoT ontologies are discussed, before addressing BP context models. Literature on process mining in IoT environments is discussed in detail in Sect. 6.

2.1 IoT Ontologies

Recently, the focus in IoT ontologies has shifted from the creation of ontologies that are as complete as possible (e.g. the Semantic Sensor Network (SSN) ontology[1]) to the development of new ontologies that are simpler and more practical (e.g. IoTStream [7]). Two such ontologies are the Sensor, Observation, Sample and Actuator (SOSA) ontology [10] and IoTStream [7].

SOSA proposes three perspectives: the sensor, observation and actuator perspectives [10]. IoTStream is a more specific ontology, inspired by SOSA, that focuses on the treatment of streaming data [7]. Both of these ontologies are *event-centric*, in the sense that they focus on data generation and treatment, and less attention is paid to the devices and platforms IoT relies on.

[1] https://www.w3.org/TR/2017/REC-vocab-ssn-20171019/.

2.2 Business Process Context Modelling

One of the first BP context models was proposed by Rosemann et al. [16]. In this paper, the authors described an onion model where context was split in four layers (listed from closest to farthest from the process): immediate context, internal context, external context and environment context. van der Aalst developed an akin onion model a few years later [1]. Another relevant representation was proposed by Ghattas et al. [8], who extended the generic process model (GPM) with a context model $C = <I, X>$ that links each instance of the process with 1) I, the initial state of its variables and 2) X, the inputs from the external environment that affect the instance.

In a recent review paper, Brunk [3] proposed a taxonomy for BP context data with six dimensions: time, structure, origin, relevance, process relation and runtime behaviour. The dimensions proposed describe traditional BP context accurately, but they are not suitable for IoT data. For instance, typical IoT context variables such as *temperature*, can hardly fit in the origin dimension.

Another approach was followed by van der Werf et al. [23], who represented the context of a BP in a domain model.

However, these papers do not discuss context based on sensor data in particular. This is done by Koschmider et al. [12], who model context information in a hierarchy that contains three elements: raw data, simple context information and complex context information.

3 Conceptual Ambiguity in IoT and PM

To bring the IoT and PM fields of study together, there needs to be an agreement on some common fundamental concepts. However, a recurrent issue when trying to bridge IoT and PM is that some common concepts are not understood homogeneously across both domains, such as the concept of *context*. This lack of homogeneity can create confusion and undermine the integration of the two fields. In this section, we start by defining the concept of IoT data, explaining next the concepts of *context* and *event*. We especially highlight the differences in understanding of those concepts by the IoT and PM fields.

3.1 IoT Data

To understand IoT data, we start with the concept of IoT. A profusion of definitions exists, and the one we retain is from Dorsemaine et al., which is synthetic and explicitly mentions the various aspects of IoT: IoT is a "Group of infrastructures interconnecting connected objects and allowing their management, data mining and the access to the data they generate." [6]. Relying on this definition of IoT, we can say that IoT data are all the data collected by the objects belonging to connected infrastructures. These data describe physical objects (the so called *Things*) or the physical environment. Examples of IoT data are the temperature in a refrigerated area, the location of a package in a warehouse, the heart rate of a patient, etc.

3.2 Context in PM vs Context in IoT

Context was defined by Dey [5] as: "any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." This broad definition is referred to in both IoT and PM, but what it means in practice substantially differs in each domain, as the situation is viewed from different angles.

In IoT, the notion of context refers to the physical context of a system. Data about the physical context are usually gathered by sensors that measure e.g. the location of objects, the ambient temperature, the movement of people or objects, etc. In PM and BPM, the context that is typically taken into account is the context of the BP that is analysed. In general, this translates to factors that impact the design or the execution of the process [1,16]. However, within the PM field, the understanding of context can differ from this definition. Most papers describe context-aware PM approaches (e.g. [1–3,23]), and understand context in the same way as in BPM. But we can also find some papers describing PM approaches applied to context-aware environments (e.g. [13,20]), which understand context like in IoT, as the physical context.

There is thus a discrepancy in the understanding of context. As a consequence, some IoT context data do not fit in taxonomies describing business process context variables; e.g., a variable such as the temperature in a room can hardly fit in any of the categories defined by Rosemann et al. [16]. However, such a variable can be useful to describe the context of a business process, and should be taken into account in PM.

On the other hand, not all the parameters that are measured by sensors in context-aware environments are relevant for PM: only those that impact the process are. The notion of context that should be used in PM using IoT is therefore the business context as understood in BPM, including relevant physical parameters, which have so far been largely overlooked.

3.3 Process Event vs IoT Event

Fundamental to PM, the concept of event is also very important in IoT. A distinction between the two acceptations of the term is recognised [11,20]. This distinction is usually limited to placing events from IoT at a lower abstraction level than events in PM. However, both definitions differ in more than that if we do a more detailed evaluation.

On the IoT side, an event can be defined as a time-value set [17]:

$$<key, value, destination, generation\_time, release\_time>$$

where the semantics of a particular event are specified by the key-value pair. Notice that this is a *data* definition, characterising a data construct.

On the PM side, a common definition of the term event is the one of the XES Standard: "Events represent atomic granules of activity that have been observed during the execution of a process. As such, an event has no duration" [9].

We can especially notice that an event is more broadly defined in the IoT literature than in the PM literature, as an event in IoT can represent a very wide range of things, depending on the key-value pair. An event as understood in IoT can be linked either with an event of the process (e.g. a patient taking a blood test), or with a context variable (e.g. the patient's insulin level), whereas an event as understood in PM can only correspond to the first one. This means that, as such, PM algorithms cannot simply be run on IoT data.

To connect these two concepts, and to reconcile the views of the IoT and PM domains on context, we developed a conceptual model to help to build a common comprehension of the structure of information to IoT and PM.

4 Connecting IoT and Process Mining: A Conceptual Model

In this section, we propose and discuss a conceptualisation that shows the link between IoT and PM, based on: (1) the concepts defined in the previous section, (2) IoT ontologies, and (3) BP context data models. To create this model, we took inspiration from the methodology described by Noy and McGuinness [15].

Following this methodology, as a first step, we formulated the requirement that our model had to fulfil: our goal is to model the link between data generated from, or captured by, IoT devices and PM event logs. The model should be able to represent the different concepts involved in IoT-enhanced PM, and to distinguish different concepts (i.e., different types of events) that are often confused in PM.

Then, we reviewed existing models, focusing on lightweight and event-centric IoT ontologies (IoTStream [7], SOSA [10]), as well as context data models (e.g., [3,16]).

After this, our third step was to look for recurrent terms and concepts. We searched for concepts that were often present in IoT ontologies and for concepts that were often present in the BP context literature. We proposed archetypal classes of objects in IoT and BP context, as well as concepts that were common to both IoT ontologies and BP context models. Recurrent concepts in IoT ontologies are sensor/device, observation, observable property, and analytics, while recurrent terms in BP context are context variable, event log, and data.

Events are central in both IoT and PM. However, as mentioned in Sect. 3, this concept is difficult to grasp and it is not understood in the same way in both fields. We propose to use a generic definition of event, which both IoT and PM experts can accept: "An **Event** is an actual occurrence or happening that is significant (i.e. it falls within a domain of interest to the system), instantaneous (i.e. it takes place at a specific point in time), and atomic (i.e. it either occurs or not)" [22] . This definition acknowledges that any occurrence that is actual (i.e. happens in the real world), atomic and instantaneous, only needs to be significant to a certain purpose or in a certain application to be an event. Events in PM are significant for the execution of the process, while events in IoT measure relevant factors of the physical environment. Examples of events complying with this definition include e.g. the termination of an activity, a report on daily sales, the entrance of a person in a certain area, or the switching on or off of a lamp.

The fourth step was to create the classes of our conceptual model, and to link them together. The result can be seen on Fig. 1. The model, built as a UML class diagram, is constituted of two main parts: the first one, from *Observable Property* to *Event*, describes how data are captured and managed by IoT devices (following IoT ontologies), and the second one, from Event to *Event Log Entry* (including *Process-Aware IS* and *IS data entry*) shows how data are processed in PM to create a contextualised event log. A link is made through the common construct of Event. Next, we define the different terms represented in the model.

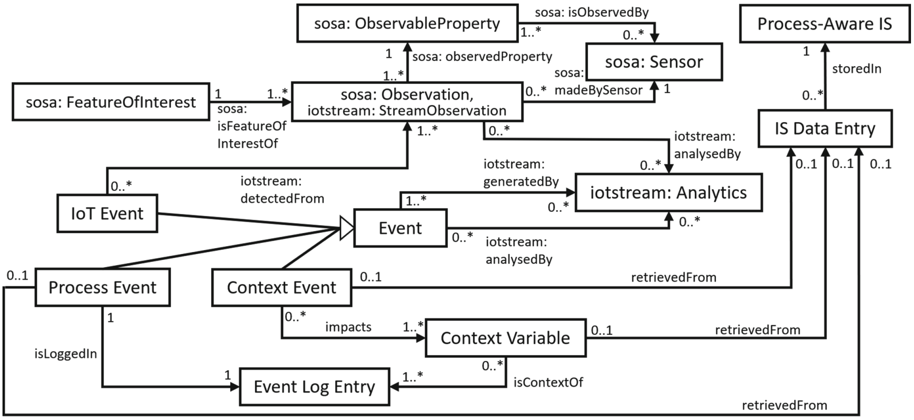


Fig. 1. Core of the model linking IoT with PM

A **Sensor** is an IoT device that measures the state of a real-world phenomenon, named Observable Property in SOSA [10]. An **Observable Property** is an observable quality (property, characteristic) of a *Feature Of Interest*. Examples of Observable Properties are: the outside temperature, the location of a truck, the weight of a container. A **Feature Of Interest** is the thing whose property is being estimated or calculated in the course of an observation (e.g. the container whose weight is measured). An **Observation** is a measurement of an Observable Property; it provides the result of estimating or calculating a value of an observable property (e.g. the measured weight of the container). The case of an actuator (an IoT device that can interact with the environment) generating the data can be modelled similarly, with actuator, actuatable property and actuation classes that mirror the sensor, observable property and observation. To avoid overloading the model, this is omitted in the figure.

IoT Events can be derived from Observations or other IoT events, and are a specialisation of Event that is defined as an instantaneous change in a real-world phenomenon that is monitored by a Sensor. Several IoT Events can be detected from the same Observation, e.g., an observation of the Observable

Property "temperature" can trigger the IoT event "temperature decreases to 0 °C" and "it is freezing".

Likewise, an IoT Event can be created directly by a change in the Observations of a Sensor (e.g. "temperature reaches 23 °C" is directly linked to the observation "23" of a temperature sensor), or it can be derived by processing one or several observation(s) from the same sensor (e.g. "temperature has increased" results from the processing of two temperature observations), with *Analytics* techniques. **Analytics** is an umbrella term from the IoTStream ontology [7] used here to describe any technique that allows the extrapolation of an Event from an Observation or another Event, such as e.g. event abstraction, complex event processing, database query, stream annotation, activity recognition, event-activity and event-case correlation, aggregation techniques, filtering techniques or machine learning algorithms.

Two other types of Events (*Context Event* and *Process Event*), which typically have richer semantics, can be derived from IoT Events using Analytics. A **Process Event** is an instantaneous change of state in the transactional lifecycle of an activity. This type of event corresponds to the usual notion of event in PM. Note that we decouple the occurrence of the change of state in the activity lifecycle and the attributes that are usually present in event data structures. Conceptually, we consider the attributes independent of the existence of the process event, and we model them separately (with the Context Event and Context Variable classes). An example of Process Event is the arrival of a package at a storage facility, which could have as attribute the size or the weight of the package.

A **Context Event** is an instantaneous change in a real-world phenomenon (deduced from an IoT Event or an IS Data Entry), that has an impact on the execution of the process (i.e. it impacts a *Context Variable*), but that does not change its control-flow state. Examples of Context Variables include the location of a package in a delivery process, the vital signs of a person in a health monitoring process, etc. An example of Context Event would be, e.g., a package has arrived at a certain area, which makes the package ready for pick up.

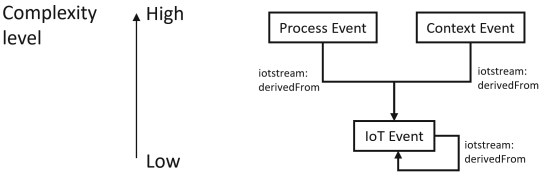


Fig. 2. Hierarchy of event specialisations in the model

Events in the model follow a hierarchy based on their complexity, as shown on Fig. 2. A higher-level event can be deduced from one or several lower-level events, and similarly a lower-level event can be the basis of one or several higher-level events. This mechanism is inspired by CEP [22], as was also suggested by

Soffer et al. [18]. IoT Events can cascade until a deduced event has a direct relationship with the process, i.e. it is a Process or a Context Event. Note that an Analytics technique, possibly trivial, is required to derive an Event from one or several other Events.

As stated earlier, a **Context Variable** is a parameter that has an influence on the execution of the process. Brunk [3] distinguished four categories of context variables, depending on their relationship with the process: activity-related, process event-related, control-flow-related, and artefact-related. Note that Context Variables might be at the level of activities, process instances, or even the overall process.

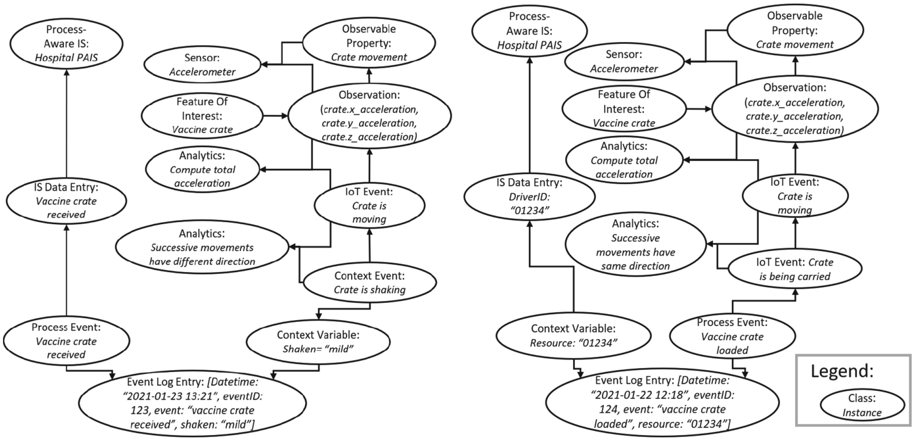
Process-Aware Information System (PAIS) and **Information System (IS) Data entry** represent the traditional PM data sources. A PAIS is an IS that records process data (i.e., IS data entries). An IS Data entry can relate to three classes: Process Event, Context Event and Context Variable. The link between PAIS and Process Event is the usual path of data used in PM, which are entered in the PAIS at runtime and later extracted to form an event log. Usually, in PM, data used as context variables are data retrieved from the IS and are considered rather static. But this does not mean that such context variables may not be subject to change. Take, for example, the amount of a claim in a claim handling process. The claim amount is usually assumed fixed, but it can actually change, as a result of, e.g., a reevaluation of the claim by an expert. This is why IS Data Entry is linked with both Context Variable and Context Event.

Finally, **Event Log Entry** is the point where Context Variables are linked with Process Events in the contextualised event log, which would contain logs of process events together with the context in which they took place. Note that, although these classes are not linked with Analytics in Fig. 1, it does not necessarily mean that Analytics are not used. Analytics is linked with Observation and Event to emphasise the importance of Analytics techniques to derive Events from Observations or other Events, but it may be that, e.g., event correlation or data fusion techniques are necessary to match a Process Event with the relevant Context Variables, or to derive an IoT Event from an Actuation. This is omitted in the figure for the sake of clarity.

5 Use Case Validation

In this section, we present a lifelike use case showing how the path between IoT data and a PM event log can be represented using this conceptualisation.

Consider the process of transporting *Moderna* vaccines from their production facility to the patients in Belgium. The vaccines are manufactured in a main production plant in the US, before being shipped to a central storage facility in Belgium. The vaccine crates are then dispatched to local vaccination centres where each dose is administered to a patient. The vaccines being particularly fragile, one would like to keep track of shocks and bumps experienced by the crates during transport, to detect during which activities most shocks are incurred, and improve the process to minimise this number. Figure 3 shows how to use our model to map different concepts from the raw output of an IoT sensor to an entry in the event log.


Process-Aware IS: Hospital PAIS
Sensor: Accelerometer
Observable Property: Crate movement
Feature Of Interest: Vaccine crate
Observation: (crate.x_acceleration, crate.y_acceleration, crate.z_acceleration)
Analytics: Compute total acceleration
IS Data Entry: Vaccine crate received
IoT Event: Crate is moving
Analytics: Successive movements have different direction
Context Event: Crate is shaking
Process Event: Vaccine crate received
Context Variable: Shaken= "mild"
Event Log Entry: [Datetime: "2021-01-23 13:21", eventID: 123, event: "vaccine crate received", shaken: "mild"]
Process-Aware IS: Hospital PAIS
Observable Property: Crate movement
Sensor: Accelerometer
Feature Of Interest: Vaccine crate
Observation: (crate.x_acceleration, crate.y_acceleration, crate.z_acceleration)
Analytics: Compute total acceleration
IS Data Entry: DriverID: "01234"
IoT Event: Crate is moving
Analytics: Successive movements have same direction
IoT Event: Crate is being carried
Process Event: Vaccine crate loaded
Context Variable: Resource: "01234"
Event Log Entry: [Datetime: "2021-01-22 12:18", eventID: 124, event: "vaccine crate loaded", resource: "01234"]
Legend:
Class: Instance

(a) (b)

Fig. 3. Example instances for the vaccine shipment process.

While the Features Of Interest *Vaccine crates* are being handled, their Observable Property *Crate movement* is recorded by an *Accelerometer* Sensor. Observations of this sensor are triplets (x, y, z) containing the acceleration in the three dimensions of space. The IoT Event *Crate is moving* can be derived from such an Observation. Comparing the movement with previous movements can tell if the crate is being shaken (which corresponds to the Context Event *Crate is shaking*, Fig. 3(a)) or if it is being displaced (which could detect a Process Event *Crate is loaded*, Fig. 3(b)), depending on the direction of consecutive movements (consecutive movements in the same direction correspond to a displacement, while consecutive movements in different directions indicate a tremor). The Context Event *Crate is shaking* impacts the Context variable *Shaken*, which after a certain amount of shocks becomes equal to "mild", to reflect the magnitude of shaking undergone by the crate (part (a) on Fig. 3).

Recording this Context Variable with each Process Event allows determining which activities shake the crates most. It can also be crossed with other Context Variables (e.g. the *Resource* driving the truck transporting the vaccines, an activity-related Context Variable that can be found as an IS Data Entry of the PAIS), to determine under which circumstances shocks are minimised (part (b) on Fig. 3).

This helps in 1) retracing the sources of the event log (IoT and PAIS), and 2) getting a deeper understanding of the links between the raw accelerometer data and the process events and context variables in the event log, as well as 3) distinguishing process events (e.g. *Vaccine crate loaded*) from context information (e.g. *Shaken*).

6 Related Work

Most of the literature that tackles PM using IoT data proposes step-by-step frameworks to extract an event log from low-level IoT data, such as those proposed by Koschmider et al. [11], Trzcionkowska and Brzychczy [21] or Soffer et al. [18]. There are differences from one framework to the other, but typical steps included in these frameworks are preprocessing the raw data, activity recognition or discovery and event abstraction. These works differ from ours as 1) they focus on the processing of the data (the “how”) while we concentrate on the data themselves (the “what”), and 2) although contextual sensor data are included, their use is limited to supporting the discovery of activities or the abstraction of events, as in [12] or [19], i.e. the IoT data are not used to mine the context of the process model.

E.g., using the framework of Koschmider et al. to model the use-case in Sect. 5 would yield the following: in step 1, accelerometer data would be correlated with activity “vaccine crate loading”. Step 2 would extract the rule that successive movements in the same direction characterise the “vaccine crate loading” activity, and step 3 would apply this rule to the whole sensor data to create an event log with the activity instead of the sensor data. The Process Events derived are similar to these described by our model, but many aspects, such as the context information, e.g. the Context Variable “shaken”, are not included. The main steps of these frameworks can also be linked with some parts of the model; see Fig. 4.

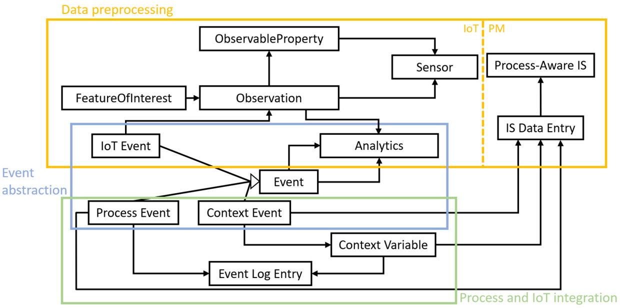


Fig. 4. Translation of typical IoT PM frameworks steps on our model

Furthermore, existing BP data models cannot model the use-case either. XES [9] is at a high level of abstraction, and is designed to store Process events only. Extensions of XES exist, among which the micro-event extension, which makes it possible to define a hierarchy of events, but it does not make it possible to link multiple higher-level events to a lower-level event[2]. The object-centric event log (OCEL)[3], a new PM data paradigm based on the concept of object, is also unsuitable for context events, as each event has to be linked with one and only one activity, which is not the case for many context events, such as e.g. the weight of a package. The context-aware GPM [8] can represent Context events, but does not distinguish them from Process events and is more coarse-grained than our model. For instance, using the context-aware GPM [8], the context in the vaccine shipment example would be modelled with: I = {{crate_shaken}, {resource}} and X = {{crate is shaking}, {vaccine crate loaded}, {vaccine crate received}}. This representation includes all the final elements of the context but, again, it misses the traceability provided by our model, and cannot include IoT metadata. Lastly, neither XES nor the context-aware GPM can represent the hierarchy of events.

7 Conclusion

In this paper, we pleaded for the use of IoT as source of context information in PM. After analysing the existing relevant models and current ambiguities affecting very important concepts, we proposed a conceptual model that defines and connects IoT and PM. As such, the model provides definitions to foster understanding between the IoT and PM community, and enables traceability between the two types of data. This is a first step towards properly understanding the relationship between IoT data and process data in order to improve their further analysis using PM. Also note that the reuse of ontologies and models from the literature automatically enables the possibility to add other additional concepts, e.g., to conceptualise the ecosystem and platforms that exist around an IoT device as described in IoT ontologies. We hope that this conceptualisation inspires others to investigate further the uncharted spaces at the intersection of IoT and PM.

In future works, we plan to complete the model by adding attributes to the classes and to make it actionable and reusable by others. To this second end, we foresee two possibilities: implementing it in OWL, or translating it into an extension of the XES Standard . The presented model also needs to be further validated. We plan to validate it with additional real-life cases and to conduct an expert-based evaluation. Finally, we also aim at researching analytics and machine learning techniques that can automatically learn the influence of IoT data on process execution and discovery.

[2] http://www.xes-standard.org/xesstandardextensions.

[3] http://ocel-standard.org/1.0/specification.pdf.

References

1. van der Aalst, W.M.P., Dustdar, S.: Process mining put into context. IEEE Internet Comput. **16**, 5 (2012)
2. Becker, T., Intoyoad, W.: Context aware process mining in logistics. Procedia CIRP **63**, 6 (2017)
3. Brunk, J.: Structuring business process context information for process monitoring and prediction. In: CBI, pp. 39–48. IEEE, June 2020
4. Dees, M., Hompes, B., van der Aalst, W.M.: Events put into context (EPiC). In: ICPM, pp. 65–72. IEEE, October 2020
5. Dey, A.K.: Understanding and using context. Pers. Ubiquit. Comput. **5**(1), 4–7 (2001)
6. Dorsemaine, B., Gaulier, J.P., Wary, J.P., Kheir, N., Urien, P.: Internet of things: a definition & taxonomy. In: NGMAST, pp. 72–77. IEEE, September 2015
7. Elsaleh, T., Enshaeifar, S., Rezvani, R., Acton, S.T., Janeiko, V., Bermudez-Edo, M.: IoT-stream: a lightweight ontology for internet of things data streams and its use with data analytics and event detection services. Sensors **20**(4), 953 (2020)
8. Ghattas, J., Soffer, P., Peleg, M.: A formal model for process context learning. In: Rinderle-Ma, S., Sadiq, S., Leymann, F. (eds.) BPM 2009. LNBIP, vol. 43, pp. 140–157. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12186-9_14
9. Gunther, C.W., Verbeek, H.: XES standard definition, March 2014
10. Janowicz, K., Haller, A., Cox, S.J.D., Le Phuoc, D., Lefrançois, M.: SOSA: a lightweight ontology for sensors, observations, samples, and actuators. J. Web Semant. **56**, 1–10 (2019)
11. Koschmider, A., Janssen, D., Mannhardt, F.: Framework for process discovery from sensor data, p. 8 (2020)
12. Koschmider, A., Mannhardt, F., Heuser, T.: On the contextualization of event-activity mappings. In: Daniel, F., Sheng, Q.Z., Motahari, H. (eds.) BPM 2018. LNBIP, vol. 342, pp. 445–457. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11641-5_35
13. Leotta, F., Mecella, M., Mendling, J.: Applying process mining to smart spaces: perspectives and research challenges. In: Persson, A., Stirna, J. (eds.) CAiSE 2015. LNBIP, vol. 215, pp. 298–304. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19243-7_28
14. Mannhardt, F.: Sepsis cases - event log, December 2016
15. Noy, N.F., McGuinness, D.L.: Ontology development 101: a guide to creating your first ontology, p. 28 (2001)
16. Rosemann, M., Recker, J., Flender, C.: Contextualization of business processes. IJBPIM **3**(1), 47 (2008)
17. Serpanos, D., Wolf, M.: Internet-of-Things (IoT) Systems. Springer, Cham (2018). http://link.springer.com/10.1007/978-3-319-69715-4
18. Soffer, P., et al.: From event streams to process models and back: challenges and opportunities. Inf. Syst. **81**, 181–200 (2019)
19. Sztyler, T., Carmona, J., Völker, J., Stuckenschmidt, H.: Self-tracking reloaded: applying process mining to personalized health care from labeled sensor data. In: Koutny, M., Desel, J., Kleijn, J. (eds.) Transactions on Petri Nets and Other Models of Concurrency XI. LNCS, vol. 9930, pp. 160–180. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53401-4_8

20. Tax, N., Sidorova, N., Haakma, R., van der Aalst, W.M.P.: Event abstraction for process mining using supervised learning techniques. arXiv:1606.07283 [cs] 15, pp. 251–269 (2018)
21. Trzcionkowska, A., Brzychczy, E.: Practical aspects of event logs creation for industrial process modelling. Multidisc. Aspects Prod. Eng. **1**(1), 77–83 (2018)
22. Wasserkrug, S., Gal, A., Etzion, O., Turchin, Y.: Complex event processing over uncertain data. In: DEBS, p. 253. ACM Press (2008)
23. van der Werf, J.M.E.M., Verbeek, H.M.W., van der Aalst, W.M.P.: Context-aware compliance checking. In: Barros, A., Gal, A., Kindler, E. (eds.) BPM 2012. LNCS, vol. 7481, pp. 98–113. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32885-5_7

ML4PM 2021: 2nd International Workshop in Leveraging Machine Learning for Process Mining

2nd International Workshop in Leveraging Machine Learning for Process Mining (ML4PM 2021)

The interest in combining Machine Learning (ML) and Process Mining (PM) has seen increasing growth in the last few years. Nowadays, the application of ML to PM is considered as the emerging technology that is fostering a new paradigm for improving business process management by enabling process task automation, simplification, and monitoring. The intent of the 2nd International Workshop in Leveraging Machine Learning for Process Mining has been to provide a venue to discuss the recent research developments at the intersection of ML and PM by bringing together practitioners and researchers from both communities. The open call for contributions has solicited submissions in the areas of automated process mining and updating, conformance checking, predictive and prescriptive process mining, multi-perspective and multi-dimensional process mining, applications of deep learning techniques, and transfer learning, IoT business services, and block-chains.

The workshop has attracted nineteen submissions by confirming the liveliness of the field. From the received nineteen submissions, seven submissions have been passed to the review process and accepted for presentation at the workshop, while three papers have been presented in the poster session of the workshop. Each paper has been reviewed by three or four members of the program committee. Papers presented at the workshop have been also selected for inclusion into the post-proceedings. These articles are briefly summarized below.

The paper by Chiorrini et al. investigates how to use instance graphs derived from traces for next activity prediction, in order to improve predictive performances by exploiting information about parallelism among activities.

The paper by a Fani Sani et al. describes an instance selection procedure to speed up the training stage of the next activity prediction methods by maintaining reliable levels of prediction accuracy.

The paper by Peeperkorn et al. proposes an evaluation scheme tailored towards measuring the capacity of deep learning models to learn process model structures.

The paper of Post et al. tackles the problem of detecting exceptions by encoding traces, assigning an anomaly score to each trace, and using the domain knowledge of auditors to update the anomaly scores assigned through active anomaly detection.

The paper by Pourbafrani et al. illustrates an approach that increases the interpretability of Remaining Time Prediction models by accounting for extracted features for multiple performance patterns caused by inter-caste dynamics.

The paper of Shoush and Dumas presents a prescriptive monitoring technique that combines predictive modeling and causal inference, in order to identify traces that are likely to lead to a negative outcome and estimate the effect of an intervention on a trace's outcome.

The paper of Stevens et al. introduces a definition of explainability that allows comparing different outcome-oriented predictive models based on model-agnostic quantitative measures.

In addition to these seven papers and the four posters, the program of the workshop has included the panel on "Machine Learning and Process Mining - Marriage or Cohabitation?" that involved by Ernesto Damiani, Chiara Di Francescomarino, Marlon Dumas, Wil van der Aalst, as panelists.

We would like to thank all the authors who have submitted papers for publication in this book. We are also grateful to the members of the Program Committee and external referees for their excellent work in reviewing submitted and revised contributions with expertise and patience.

October 2021

Paolo Ceravolo
Sylvio Barbon Jr.
Annalisa Appice

Organization

Workshop Chairs

Paolo Ceravolo	Università degli Studi di Milano, Italy (https://orcid.org/0000-0002-4519-0173)
Sylvio Barbon Jr.	State University of Londrina, Brazil (https://orcid.org/0000-0002-4988-0702)
Annalisa Appice	Università degli Studi di Bari, Italy (https://orcid.org/0000-0001-9840-844X)

Program Committee

Michelangelo Ceci	University of Bari Aldo Moro
Shridhar Devamane	APS College of Engineering, Bangalore
Chiara Di Francescomarino	Fondazione Bruno Kessler
Matthias Ehrendorfer	University of Vienna
Luís Paulo Faina Garcia	University of Brasilia
Antonella Guzzo	Università della Calabria
María Teresa Gómez López	University of Seville
Mariangela Lazoi	University of Salento
Fabrizio Maria Maggi	Free University of Bozen-Bolzano
Gabriel Marques Tavares	Università degli Studi di Milano
Emerson Cabrera Paraiso	Pontifícia Universidade Católica do Paraná
Sarajane Marques Peres	University of São Paulo
Domenico Potena	Università Politechnica delle Marche
Natalia Sidorova	Eindhoven University of Technology
Niek Tax	Booking.com
Irene Teinemaa	Booking.com
Bruno Bogaz Zarpelão	State University of Londrina
Wil van der Aalst	RWTH Aachen University

Additional Reviewers

Andrea Chiorrini
Antonio Pellicani
Aurora Rimini
Icaro Miranda
Vincenzo Pasquadibisceglie
Pedro Pio

Exploiting Instance Graphs and Graph Neural Networks for Next Activity Prediction

Andrea Chiorrini(✉), Claudia Diamantini, Alex Mircoli, and Domenico Potena

Department of Information Engineering,
Polytechnic University of Marche, Ancona, Italy
a.chiorrini@pm.univpm.it, {c.diamantini,a.mircoli,d.potena}@univpm.it

Abstract. Nowadays, a lot of data regarding business process executions are maintained in event logs. The next activity prediction task exploits such event logs to predict how process executions will unfold up until their completion. The present paper proposes a new approach to address this task: instead of using traces to perform predictions, we propose to use the instance graphs derived from traces. To make the most out of such representation we train a message passing neural network, specifically a Deep Graph Convolutional Neural Network to predict the next activity that will be performed in the process execution. The experiments performed show promising performance hinting that exploiting information about parallelism among activities in a process can induce a performance improvement in highly parallel process.

Keywords: Deep Learning · next activity prediction · Predictive Process Monitoring · Graph Neural Networks · Process Mining

1 Introduction

Nowadays, many business processes maintain a significant amount of data regarding their executions in the form of events logs. The predictive process monitoring field is concerned with the exploitation of such event logs to predict how such executions will unfold up until their completion. Predictive process monitoring includes various tasks: one of them is the next activity prediction, that is concerned with the prediction of what will next happen in an execution.

In the last years predictive process monitoring, and particularly next activity prediction, is receiving an ever-increasing attention, and researchers are tackling the problem using various deep learning approaches [3,4,13–15,19,21]. It is also known that using graphs is an extremely convenient way of representing process executions [6,20]. Recently, a new type of neural network architecture is gaining ground in the deep learning community: the graph neural network [16,24]. Still, there is almost no work in the literature that evaluates the possibility of exploiting such family of networks for the predictive process monitoring tasks. Driven

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 115–126, 2022.
https://doi.org/10.1007/978-3-030-98581-3_9

by this lack of studies, we introduce a methodology to properly exploit the graph representation of processes executions [6] for the next activity prediction task.

In particular, our proposal requires a business process event log and a model of such process, which are used to build for each trace a proper corresponding instance graph. From this graph representation, the dataset is then created and the Deep Convolutional Graph Neural Network [25] is trained to perform the prediction for next activity. Experimental results show that information contained in instance graphs, in particular concerning the parallelism among activities, can improve the prediction accuracy.

The rest of the paper is organized as follows: in Sect. 2 a review of the task-relevant literature is performed. Section 3 describes the proposed methodology. In Sect. 4 a comparison with the results from the relevant literature is performed and commented. Finally, in Sect. 5 we draw some conclusions and list further directions of research.

2 Related Work

In the scientific community there is an ever-increasing interest in the application of deep learning techniques to predictive process monitoring tasks. In [18] the authors proposed to adopt an LSTM architecture for the one step ahead event prediction and the suffix prediction, using the one-hot encoding of the associated activity and three temporal features related to the event's timestamp. The LSTM architecture has also been used by [3,8,13]. The approach in [8] predicts only the next activity type, while [13] predicts the next activity and all the associated categorical attribute. The authors of [3] combine both [8] and [18] approaches to extend [8] to the next completion time prediction, using an abstract notion of class resource, i.e. group of resources that usually perform similar activities. Recently, in [14] a CNN architecture has been proposed: the authors convert the sequential temporal data in the log into a spatial representation to then treat data as images. In [15] the approach has been further extended by the authors.

The of use Generative Adversarial Nets (GANs) is proposed in [19] to address the next event prediction task, tackling in this way the lack of sufficient training data that often impact performances. In the GAN approach the authors used LSTM networks for both the generator and discriminator. For this reason they trained various networks, each one over sub-sequences of processes of specific length, thus producing more than one prediction model for each process, the same limit of the other LSTM based approaches.

In 2008 Scarselli et al. introduced "The Graph Neural Network Model" (GNN) [16], a neural network model capable of processing data in graph domains. Since their definition, graph neural networks have been increasingly used in several fields. In a recent survey [24] it has been proposed *a new taxonomy to divide the state-of-the-art GNNs into four categories, namely, recurrent GNNs, convolutional GNNs, graph autoencoders, and spatial-temporal GNNs*. In the same work the authors also outline the various application fields.

It is relevant to note that very recently a proposal of usage of Graph Convolutional Neural Network in the next activity and timestamp predictions has

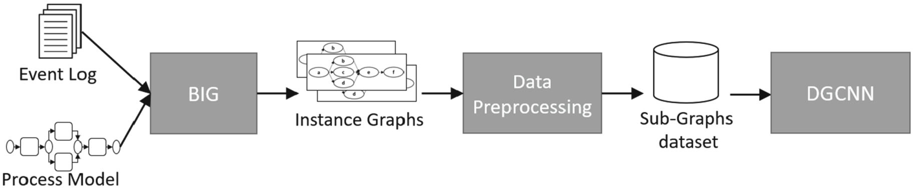


Fig. 1. The BIG-DGCNN methodology pipeline

been formulated by [21]. Their approach differs from the one proposed in the present paper as they propose to use a Directly Follows Graph representation to model the traces while our approach adopts an Instance Graph representation. Furthermore, the network architecture used in this work and in [21] are different. In [21] a single graph convolutional layer followed by two fully connected layers are used, while in the present paper a more elaborated architecture is adopted as described in Sect. 3. Also lately, in [23] the author explored the opportunity of exploiting gated graph neural network for the next activity prediction. In the work different graph representations are tested but are only compared with self implemented baselines, focusing the work exclusively on network architecture.

3 Methodology

In this Section we discuss the proposed methodology, first explaining the rationale of the whole pipeline shown in Fig. 1, then delving into its relevant aspects.

The goal of the present work is to define a robust approach that makes use of information about the parallelism among activities in the next activity prediction task. To this end, we propose to represent each trace with its corresponding Instance Graph (IG), and to process IGs by graph neural networks, that are designed to natively manage graph structures.

It is known that replaying a trace on a Petri net it is possible to produce an instance graph representing the process execution [6]. The starting point of our methodology is thus an event log and a model of the process expressed as Petri net. The model can be given by a domain expert, opening to the possibility to provide the desired perspective over the process, or a Petri net can be derived by some process discovery algorithm. In both cases, a problem arises with highly variable processes whose event log includes many non-conforming traces. This situation often occurs when an a-priori model has been defined for a process that rapidly evolves thus making the model obsolete; or in decision-intensive processes where only a high-level model can be defined and executions vary from case to case (e.g. care process in a hospital). When process discovery is exploited to synthesize a process, lossy algorithms may be adopted (e.g. Heuristic Miner, infrequent Inductive Miner) that discard uncommon behaviours to ensure simpler and more meaningful models. In the presence of non-conforming traces a simple replay will lead to not valid IGs, e.g. disconnected graphs, or graphs with more than one terminal node. To deal with this issue, our approach also provides a repairing of non-conforming traces by the adoption of the BIG algorithm [5]. This prevents the loss of data and grants to our system the possibility to predict also uncommon events. From BIG-generated IGs the dataset is then created and the graph neural network trained to perform the prediction of the next activity. Among the different architectures existing in the literature, the Deep Graph Convolutional Neural Network (DGNN) [25] has been chosen. In the following we will refer to our methodology as BIG-DGCNN.

3.1 Building Instance Graphs

An Instance Graph (IG) is a graph that describes a specific execution of some process model. In an IG each node represents an activity, and an edge between activity A and activity B denotes the existence of a *causal relation* between A and B, namely the fact that B cannot be executed until A is terminated; in other words, the execution of B depends on the execution of A. For a more formal definition of instance graphs and causal relations see [6]. In an instance graph parallelisms are shown besides causal relations, while choices are not represented. This is obvious, since in each single execution choices have already been made. Activities that can be done in parallel within one instance can appear in any order in the trace without changing the resulting instance graph. Hence, an execution is represented by an unique IG, which in turn represent different traces.

Instance graphs can be built from a set of traces in an event log and the corresponding process model. In this paper we refer to the Building Instance Graph (BIG) algorithm proposed in [5]. Unlike other approaches in the literature, BIG enables the representation of parallel behaviors and is able to handle traces that do not conform to the model. BIG is a two-steps algorithm: first an IG is extracted for each trace, then IGs from non-conforming traces are repaired. In the first step, a node is created in the IG for each event of the trace and the node is labeled with both the activity associated with the given event and the position of the event in the trace. For each pair of events (e_i, e_j) with $i < j$ such that causal relation exists between the corresponding activities in the model, an edge is inserted between the corresponding nodes in the IG if and only if between the pair of events there is neither another causal successor of e_i, nor a causal predecessor of e_j.

It is worth noting that, if the trace does not fit the model, the corresponding IG is affected by several issues, e.g. being a disconnected graph and having multiple terminal nodes. From a semantic point-of-view these IGs represent many more behaviours than the model actually allows for, namely they over-generalize. Hence, the representation of the process behavior provided by these IGs is very poor. In order to mitigate these issues, in [5] a procedure for the repair of IGs is proposed.

First, non-synchronous movements in the trace, which lead to issues in the IG, are identified by using aligned-based conformance checking technique [2], which returns the kind of non-synchronous movements and their position in the trace. This information is used to repair each move-in-the-model and each move-in-the-log occurring in the alignment. The former leads to disconnected graphs, which are repaired by identifying nodes corresponding to activities that are in a causal relation with the move-in-the-model and properly connecting them. For each move-in-the-log, the repair procedure changes the edges connecting the nodes corresponding to the events before and the events after the non-synchronous event; the causal relations among the activity corresponding to move-in-the-log and activities related to its predecessors and successors in the trace are used to build new edges.

3.2 Data Preprocessing

This Section describes the processing of the log to obtain the dataset for training and testing. First of all, since a log not necessarily has single specific ending and starting activities, we introduce such artificial events in the logs. This is done to guarantee that a trace always has a termination activity and to ensure that parallelism at the beginning and end of a process execution are properly represented in the IG. Second, BIG is applied to obtain an IG for each trace.

The approach of this paper learns a function that, given a *graph prefix* τ_g of dimension g, returns a label a that can be interpreted as the next performed activity in the process execution. Hence, the third step of the proposed methodology is to build the pairs (τ_g, a) from an IG for any $g \in \{2, \ldots, N-1\}$ where N is the number of nodes in the IG (i.e. the trace length). Therefore from one IG, we produce $N-2$ pairs. The procedure is repeated for every IGs in the dataset.

Each node in an IG has a progressive index associated with it that represents its position in the trace. This index determines an order of the nodes, which we use to progressively build the graph prefixes from the full IG. We denote by a_i, $i = 1, \ldots, N$ the activity associated with the node of index i.

Given an IG, the graph prefix τ_2 is obtained by selecting the first two nodes and the edges between them. This prefix is labelled with the activity a_3 of the next node (Fig. 2). The next prefix is simply derived by τ_2, extending it with the node of index 3 and the edges connecting it to τ_2. The associated label is a_4 (Fig. 3). Iteratively, the procedure re-build the whole instance graph until the last node is selected as label.

We adopt the one-hot encoding representation for the set of activities in the log.

3.3 Deep Graph Convolutional Neural Network

In this paper, we use a variant of the Deep Graph Convolutional Neural Network (DGCNN) proposed in [25]. The DGCNN is composed of three sequential stages. In Fig. 4, we show a representation of the overall neural network architecture. First it has several graph convolution layers which extract the features from the nodes local substructure and define a consistent vertex ordering. Second it has a SortPoolingLayer which sorts the vertex features according to the order defined in the previous stage, selecting the top nodes. In this way the dimension of the input is unified. At last, a 1-D convolution layer and a dense layer take the obtained representation to perform predictions.

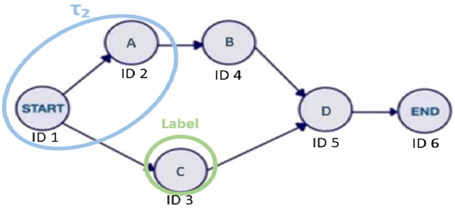


Fig. 2. First labelling step

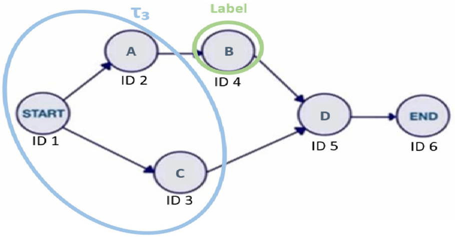


Fig. 3. Second labelling step

The graph convolution layer adopted by DGCNN is represented by the following formula:

$$Z = f(\tilde{D}^{-1}\tilde{A}XW) \tag{1}$$

where $\tilde{A} = A + I$ is the adjacency matrix (A) of the graph with added self-loops(I), $\tilde{D}$ is its diagonal degree matrix with $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$, $X \in \mathbb{R}^{n\times c}$ is the graph nodes information matrix (in our case the one-hot encoding of the activity labels associated to the nodes), $W \in \mathbb{R}^{c\times c'}$ is the matrix of trainable weight parameters, f is a nonlinear activation function, and $Z \in \mathbb{R}^{n\times c'}$ is the output activation matrix. In the formulas, n is the number of nodes of the input graph (in our case the graph prefix), c is the number of features associated to a node, and c' is the number of features in the next layer tensor representation of the node.

In a graph, the convolution operation aggregates node information in local neighborhoods so to extract local structural information. In order to extract multi-scale structural features, multiple graph convolution layers (Eq. 1) are stacked as follows:

$$Z^{k+1} = f(\tilde{D}^{-1}\tilde{A}Z^kW^k) \tag{2}$$

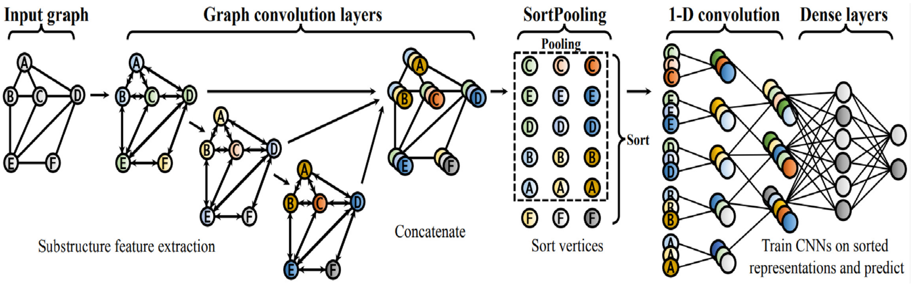
where $Z^0 = X$, $Z^k \in \mathbb{R}^{n\times c_k}$ is the output of the k^{th} convolution layer, c_k is the number of features of layer k, and $W^k \in \mathbb{R}^{c_k\times c_{k+1}}$ maps c_k features to c_{k+1} features.

The graph convolution outputs $Z^k, k = 1, ..., h$ are then concatenated in a tensor $Z^{1:h} := [Z^1, ..., Z^h] \in \mathbb{R}^{n\times \sum_1^h c_k}$ which is then passed to the SortPoolingLayer. It first sorts the input $Z^{1:h}$ row-wise according to Z^h, and then returns as output the top m nodes representations, where m is a user-defined parameter. This way, it is possible to train the next layers on the resulting fixed-in-size graph representation.

In the original proposal the DGCNN includes a 1-D convolution layer, followed by several MaxPooling layers, one further 1-D convolution layer followed by a dense layer and a softmax layer.

In the present paper we simplify the architecture leaving only one 1-D convolution layer with dropout [17] followed by a dense and a softmax layer. This is because the process mining domain tend to present smaller graphs in comparison with those of typical application domains of graph neural networks [24].

For further information on the architecture we refer the interested reader to [25].



credit to: Zhang, M., Cui, Z., Neumann, M., Chen, Y.: An end-to-end deep learning architecture for graph classification (2018)

Fig. 4. DGCNN architecture. Taken from [25].

4 Experiments

4.1 Experimental Setup

In this Section, we characterize the benchmark datasets used, then we discuss the setting of parameters for the algorithms used in the experiments.

Datasets. In order to be comparable with the literature [3,14,18,21], we tested our approach on 2 commonly used benchmark datasets, namely *Helpdesk* and *BPI12W*.

The *Helpdesk* dataset [22] contains traces from a ticketing management process of the help desk of an Italian software company. In this process all execution instances start with the insertion of a new ticket into the ticketing management system.

Table 1. Overview of benchmark dataset. $|\sigma|$ is used to represent the trace length.

Dataset	N.traces	Tot.events	N.act.types	Min $\|\sigma\|$	Max $\|\sigma\|$	Avg $\|\sigma\|$
Helpdesk	3804	13710	9	1	14	3.60
BPI12W	9658	72413	6	1	74	20.03

The *BPI12* dataset [7] is taken from a Dutch Financial Institute. The process represents a personal loan or overdraft applications within a global financing organization. The event log is a merge of three intertwined sub processes, related to the evolution of the status of the loan/draft application (*BPI12A*), of the work items belonging to the application (*BPI12W*), and of the offer belonging to the application (*BPI12O*) respectively. We considered the *BPI12W* sub-process. Furthermore, as usually done, we retained only the completed events in the log. Table 1, shows the characteristics of both datasets.

We hasten to note that *BPI12W* execution instances show no parallel activities, while some level of parallelism exists in *Helpdesk* Thus the selection of these datasets allows us to better appreciate the impact of parallelisms on performance.

Finally, in order to be as much comparable as possible with the literature, we split each log keeping the first (chronologically ordered) 67% of the traces for training and the remaining part for testing.

Parameter Settings. In the BIG-DGCNN methodology there are two main algorithms that require the setting of parameters: the infrequent Inductive Miner (iIM) [12] used to derive the Petri net representing the process models requested for the execution of the BIG algorithm, and the DGCNN. Regarding iIM, the noise threshold must be set. This threshold is used to determine how much infrequent behaviour will be filtered out when building the model. In choosing such parameter we tested the fitness of the resulting model (i.e. the extent to which the discovered model can accurately reproduce the cases recorded in the log). We changed the noise threshold with 10% step from 0% to 100% and selected the smallest noise threshold that granted at least a 90% fitness. In this way, we obtain processes that are capable of modeling a vast majority of traces while still maintaining a good degree of generalization, thus emulating a model provided by an expert, and putting us in a close-to-real setting.

Regarding the parameters of the DGCNN, as stated before we set the number of 1-D convolution layers to one, followed by a dense layer, both with 32 neurons. We used ADAM [10] as optimization algorithm and trained the network for 100 epochs with an early stopping. We used as loss function the categorical cross entropy. For both datasets we varied the following parameters:

- the number of nodes selected by the SortPooling layer (m), in $\{3, 5, 30\}$
- the number of stacked graph convolution layer (h), in $\{2, 3, 5, 7\}$
- the batch size (bs), in $\{16, 32, 64, 128\}$

- the initial learning rate (*lr*), in $\{10^{-2}, 10^{-3}, 10^{-4}\}$
- the dropout percentage (*do*), in $\{0.1, 0.2, 0.5\}$

The configurations that provide the best accuracy are shown in Table 2.

Table 2. Best network parameters for the dataset.

Dataset	m	h	bs	lr	do
Helpdesk	30	5	32	10^{-3}	0.1
BPI12W	5	5	32	10^{-3}	0.2

Tools and Hardware. All the experiments have been performed using pytorch geometric [9] with torch version 1.8.1, on a Tesla T4 GPU, a Intel(R) Xeon(R) CPU@2.20 GHz, and a 12 GB RAM.

4.2 Results

Table 3 shows the accuracy achieved on the selected datasets, by our approach and other approaches exploiting different neural network architectures.

We can see that on Helpdesk BIG-DGCNN achieves the best performance. It is worth noting that in all the competitor approaches instances are described by a richer set of features, like time information, while our input graphs encode only activity labels and flow information.

On the other hand, on the BPI12W dataset there is a relevant performance degradation w.r.t. the other methods. As noted before, BPI12W shows no parallel activities. This seems to confirm our hypothesis that properly taking into account information on parallelisms can be beneficial to the next activity prediction, whilst graph neural networks do not develop their full potential on sequential processes. The hypothesis also gains confidence when the BIG-DGCNN performances are compared against the results presented in [19] on the whole BPI12

Table 3. Literature comparison, measured accuracy

	Dataset	
Approach	Helpdesk	BPI12w
BIG-DGCNN	**82.58%**	62.43%
GCNN [21]	79.54%	65.69%
MLP [21]	82.01%	65.59%
CNN [14]	73.93 %	78.17%
CNN [15]	-	**82.20%**
LSTM [18]	71.23%	76.00%
LSTM [3]	78.90%	77.80%

dataset. There, plain CNN [14], and LSTM [3,18] approaches has been compared to a LSTM trained in a GAN framework. BIG-DGCNN is able to achieve an accuracy of 76.45% which is higher than the accuracy presented for all the plain methods, although it is lower than that achieved by LSTM+GAN. We remark that GAN is a novel learning framework, where any neural network architecture can be adopted. It is possible that the adoption of graph neural networks like those presented in this work could further improve performance, solving also a limit of the LSTM network used in [19] which is constrained to fixed size prefixes, hence forcing the training of a specialized network for every desired prefix length. We plan to experiment the GAN framework with graph neural networks in future work.

Since [21] also adopts a graph neural network architecture, one may want to analyse the reasons under the different accuracy. Various factors can be responsible for this: first, [21] adopts directly follows graphs, which are derived at a process level, while we build and properly repair the instance graph for each trace. Second, the representation of the graph is performed in a different way. The approach described in [21] only takes into account the last occurred event of a sequence of events with the same activity label when building the input graph. On the contrary, our approach considers all passed events of the trace, including the events relative to repeated actions. Third, the network used in this work is endowed with several graph convolution layers, while that used in [21] is a single layer graph convolutional neural network [11] variant.

5 Conclusions and Future Works

The main contribution of this work is the definition of BIG-DGCNN, a methodology to address the task of next activity prediction exploiting information about parallelism among activities in a process. The methodology adopts BIG [5] to repair non-conforming traces in order to always build a fully representative instance graph. Then it uses a rather new kind of neural network architecture, the Deep Graph Convolutional Neural Network [25], that is capable of effectively using the structural information of a graph in its functioning. The comparison with the literature highlights that BIG-DGCNN show promising performance in datasets relative to process with a consistent presence of parallelism, while performing less effectively in sequential datasets like BPI12W. Although it is well known that parallelism is a characterizing feature of business processes [1], variants of the approach that can better deal with sequential dataset can be worth of investigation. Other interesting future research directions include:

- extending the BIG-DGCNN inputs to all the available features of every benchmark dataset, aspect neglected in this work so to isolate the structural contribution of the process workflow,
- extending the experimentation to other datasets,
- testing the GAN training method using BIG-DGCNN thus *"eliminating the need for a large training data"* [19] and avoiding the fixed prefix restriction of GAN+LSTM.

References

1. van der Aalst, W., et al.: Process mining manifesto. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM 2011. LNBIP, vol. 99, pp. 169–194. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28108-2_19
2. Adriansyah, A., van Dongen, B.F., van der Aalst, W.M.: Conformance checking using cost-based fitness analysis. In: 2011 IEEE 15th International Enterprise Distributed Object Computing Conference, pp. 55–64. IEEE (2011)
3. Camargo, M., Dumas, M., González-Rojas, O.: Learning accurate LSTM models of business processes. In: Hildebrandt, T., van Dongen, B.F., Röglinger, M., Mendling, J. (eds.) BPM 2019. LNCS, vol. 11675, pp. 286–302. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26619-6_19
4. Chiorrini, A., Diamantini, C., Mircoli, A., Potena, D.: A preliminary study on the application of reinforcement learning for predictive process monitoring. In: Leemans, S., Leopold, H. (eds.) ICPM 2020. LNBIP, vol. 406, pp. 124–135. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-72693-5_10
5. Diamantini, C., Genga, L., Potena, D., van der Aalst, W.: Building instance graphs for highly variable processes. Expert Syst. Appl. **59**, 101–118 (2016)
6. van Dongen, B.F., van der Aalst, W.M.P.: Multi-phase process mining: building instance graphs. In: Atzeni, P., Chu, W., Lu, H., Zhou, S., Ling, T.-W. (eds.) ER 2004. LNCS, vol. 3288, pp. 362–376. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30464-7_29
7. van Dongen, B.: BPI challenge 2012, April 2012
8. Evermann, J., Rehse, J.R., Fettke, P.: Predicting process behaviour using deep learning. Decision Support Syst. **100**, 129–140 (2017). Smart Business Process Management
9. Fey, M., Lenssen, J.E.: Fast graph representation learning with PyTorch Geometric. In: ICLR Workshop on Representation Learning on Graphs and Manifolds (2019)
10. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2017)
11. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: Proceedings of the 5th International Conference on Learning Representations, ICLR 2017 (2017)
12. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from incomplete event logs. In: Ciardo, G., Kindler, E. (eds.) PETRI NETS 2014. LNCS, vol. 8489, pp. 91–110. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07734-5_6
13. Lin, L., Wen, L., Wang, J.: MM-Pred: a deep predictive model for multi-attribute event sequence. In: Proceedings, Society for Industrial and Applied Mathematics, pp. 118–126 (2019)
14. Pasquadibisceglie, V., Appice, A., Castellano, G., Malerba, D.: Using convolutional neural networks for predictive process analytics. In: 2019 International Conference on Process Mining (ICPM 2019), pp. 129–136 (2019)
15. Pasquadibisceglie, V., Appice, A., Castellano, G., Malerba, D.: Predictive process mining meets computer vision. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) BPM 2020. LNBIP, vol. 392, pp. 176–192. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58638-6_11
16. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Trans. Neural Networks **20**(1), 61–80 (2009)
17. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)

18. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive business process monitoring with LSTM neural networks. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 477–492. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_30
19. Taymouri, F., Rosa, M.L., Erfani, S., Bozorgi, Z.D., Verenich, I.: Predictive business process monitoring via generative adversarial nets: the case of next event prediction. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) BPM 2020. LNCS, vol. 12168, pp. 237–256. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58666-9_14
20. van der Aalst, W., van Dongen, B., Herbst, J., Maruster, L., Schimm, G., Weijters, A.: Workflow mining: a survey of issues and approaches. Data Knowl. Eng. **47**(2), 237–267 (2003)
21. Venugopal, I., Tollich, J., Fairbank, M., Scherp, A.: A comparison of deep learning methods for analysing and predicting business processes. In: Proceedings of International Joint Conference on Neural Networks, IJCNN. IEEE Press, July 2021
22. Verenich, I.: Helpdesk (2016). https://doi.org/10.17632/39bp3vv62t.1. https://data.mendeley.com/datasets/39bp3vv62t/1
23. Weinzierl, S.: Exploring gated graph sequence neural networks for predicting next process activities. In: Marrella, A., Weber, B. (eds.) BPM 2021. LNBIP, vol. 436, pp. 30–42. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-94343-1_3
24. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Yu, P.S.: A comprehensive survey on graph neural networks. IEEE Trans. Neural Netw. Learn. Syst. **32**(1), 4–24 (2021)
25. Zhang, M., Cui, Z., Neumann, M., Chen, Y.: An end-to-end deep learning architecture for graph classification. In: 32nd AAAI Conference on Artificial Intelligence (2018)

Can Deep Neural Networks Learn Process Model Structure? An Assessment Framework and Analysis

Jari Peeperkorn[1](✉), Seppe vanden Broucke[1,2], and Jochen De Weerdt[1]

[1] Research Center for Information Systems Engineering (LIRIS), KU Leuven, Leuven, Belgium
{jari.peeperkorn,jochen.deweerdt,seppe.vandenbroucke}@kuleuven.be

[2] Department of Business Informatics and Operations Management, Ghent University, Ghent, Belgium

Abstract. Predictive process monitoring concerns itself with the prediction of ongoing cases in (business) processes. Prediction tasks typically focus on remaining time, outcome, next event or full case suffix prediction. Various methods using machine and deep learning have been proposed for these tasks in recent years. Especially recurrent neural networks (RNNs) such as long short-term memory nets (LSTMs) have gained in popularity. However, no research focuses on whether such neural network-based models can truly learn the structure of underlying process models. For instance, can such neural networks effectively learn parallel behaviour or loops? Therefore, in this work, we propose an evaluation scheme complemented with new fitness, precision, and generalisation metrics, specifically tailored towards measuring the capacity of deep learning models to learn process model structure. We apply this framework to several process models with simple control-flow behaviour, on the task of next-event prediction. Our results show that, even for such simplistic models, careful tuning of overfitting countermeasures is required to allow these models to learn process model structure.

Keywords: Predictive Process Monitoring · Next Event Prediction · Recurrent Neural Network · Generalisation

1 Introduction

In the field of process mining, a clear trend can be discerned in terms of a shift from post factum analysis to predictive and even prescriptive modelling. This is for instance clearly reflected in the surge in papers presenting deep learning-based modelling techniques to address analysis tasks including remaining time, outcome, and next event prediction. One potentially problematic issue regarding the application of such deep learning models is the fact that, to the best of our knowledge, no research has focused on investigating whether popular modelling architectures such as LSTM neural networks, can actually "learn" process behaviour from a possibly incomplete set of example traces in an event log.

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 127–139, 2022.
https://doi.org/10.1007/978-3-030-98581-3_10

Accordingly, the main contribution of this work is to propose a framework to assess the capability of deep learning-based next event prediction techniques to truly learn process model structure. The framework consists of a variant-based resampling procedure combined with novel metrics to assess fitness, precision and generalisation of the learned neural network models. In our experimental evaluation, we rely on six relatively (and purposefully) trivial process models that reflect essential control-flow constructs in business processes. By doing so, we can investigate the relation between types of control-flow behaviour (e.g. AND, XOR, and OR split/join, loops, long distance dependencies etc.) and the capacity of deep learning models to truly learn these structural patterns. Our findings indicate that even for such trivial models, and in particular for models with parallel behaviour, rigorous application of overfitting countermeasures is required to have any chance of steering the neural network model towards the goal of truly learning process model structure, more so when compared with other domains in which such networks have been applied. These findings have important consequences, given that real-life models and event logs are usually orders of magnitude more complex than the models used here. As such, we believe that this paper opens up an important agenda for further investigation.

Our paper is organised as follows. First, Sect. 2 discusses some relevant related work. Next, our proposed framework is explained in Sect. 3. In Sect. 4, the carefully selected artificial process models are proposed, before discussing the hyperparameter grid search and presenting the experimental results. What follows is a brief discussion of the results and their implication (Sect. 5). The paper is concluded in Sect. 6, which also provides an outlook towards future research. The synthetic data, results and trained models used and presented in this paper are available online[1].

2 Related Work

In recent years, within the field of Predictive Process Monitoring (PPM), a lot of attention has been attracted by deep learning-based solutions, most frequently Recurrent Neural Networks (RNN) [3,4,6,9,15,16]. Given the scope of this paper, we limit this section to a selection of PPM works addressing the next event prediction problem, i.e. given a prefix of activities, produce a probability vector corresponding with the likelihood of each respective activity occurring as the next one. In their pioneering work, Tax et al. [15] propose to use a Long Short-Term Memory network (LSTM) to predict next events and corresponding timestamps, thereby relying on one-hot encoding of the activity labels as input for the LSTM, together with their timestamps. Moreover, Evermann et al. [6] also propose the use of LSTM networks, specifically to predict full case suffixes rather than the next event only. However, they reduce the input dimension of the event labels by using vector embeddings, and include attributes such

[1] https://github.com/jaripeeperkorn/GeneralizationPPM.

as resources. Camargo et al. [4] use separately trained embeddings of categorical variables and timestamps in order to predict both next events as well as future timestamps. Moreover, Lin et al. [9] use an LSTM encoder-decoder and a modulator structure to predict next events and suffixes, using both control-flow information as well as other event attributes. Recently Taymouri et al. [16] proposed a Generative Adversarial Networks approach to the problem of next event, suffix and timestamp prediction showing promising results. In Bukhsh [3] a transformer network approach is proposed to the problem of next event prediction. Transformer networks have recently been used to beat several benchmarks in other fields like Natural Language Processing [18].

Despite the drastic increase in research attention, no studies have investigated whether RNN-based architectures can actually learn process model structure. As such, it is unclear whether the generalisation that is expected from plain process discovery techniques is realised by neural network models. It has been shown that RNNs are universal approximators [12]. And while Siegelmann and Sontag [13] showed in 1995 already that RNNs are Turing complete, i.e. for any given computable function there exists a finite RNN to compute it, there is still much work on understanding what makes functions difficult to learn, let alone under the constraint of data incompleteness as is the case with business process data sets [11]. That is, whilst it might be clear that RNNs are capable to fit the given training data in the form of process cases, the central question we investigate here is whether such models can be constructed so that they are also able to generalise towards making good predictions for new unseen control-flow behaviour, which is highly likely to occur once a process starts to exhibit even a limited amount of complexity. In other words: do these models memorise the training data or truly learn the process structure?

3 A Framework for Assessing the Generalisation Capacity of RNNs

With the goal of this paper in mind, we set out on developing a framework that is capable to assess to what extent RNN-based architectures are capable to learn process model structure. For an introduction on how RNNs work and how they can be used in predictive process monitoring, the interested reader is referred to [6]. This framework relies on a specific resampling procedure combined with a set of new metrics to quantify recall, precision and generalisation. Recall that we restrict ourselves to next event prediction models, however, an extension to suffix prediction is trivial. Moreover, given that many remaining time or outcome prediction models also rely on incorporating control-flow information, it can be expected that these models too would benefit from proper generalisation, if at all possible.

3.1 The Resampling Procedure

A schematic overview of the assessment framework is shown in Fig. 1. We start from a (simple) process model and play-out the model to obtain a corresponding event log. We assume that we work in a setting where the number of variants (i.e. distinct traces in terms of the events and their order), is bounded. Therefore, in case of loops, we assume a maximum number of times a certain marking can be visited. Once the event log is obtained, we determine all of its unique control-flow variants. Next, a resampling procedure at the level of variants is performed to construct training and test sets. For instance, one can decide to simply retain all cases pertaining to one single variant in the test log, resulting in a “leave-one-variant-out cross-validation” (LOVOCV).

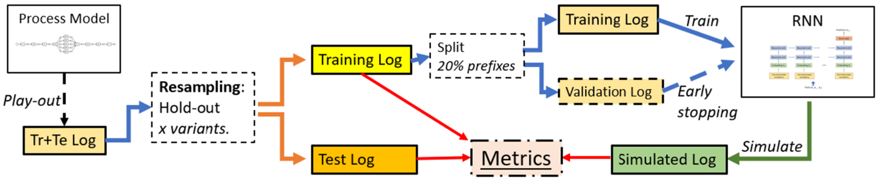


Fig. 1. Overview of the setup.

Thus, starting from the complete event log, which is referred to as the *Train+Test log (Tr+Te)*, we single out all cases pertaining to one or more variants to form the *Test log (Te)*. The remaining variants form the *Training log (Tr)*, which is split into all possible prefixes to train the model. From this set of prefixes, a *Validation log (Val)* is created, mainly to allow the training procedure of the LSTMs to use early stopping. For now, we simply perform a random selection of 20% of the *Training log*'s prefixes. The *Training* and *Validation log*'s prefixes are then used to train a model for next event prediction using the observed next events of every prefix as the target. Accordingly, the model is trained to predict for every prefix what the subsequent event's activity label will be. As such, the model outputs a probability for each of the different activities in the activity vocabulary. Once trained, we use the RNN model to simulate a *Simulated log (Sim)* as follows. We start by presenting the RNN with a prefix only containing the beginning of sequence token (BOS). As output, the model returns a probability for each of the possible activity labels to be the next event. Using these probabilities, we sample a possible next event, to be appended to the existing prefix. Subsequently the prefix is used to sample a next event in the same way. We do this until we reach the end of sequence token (EOS) or a certain predetermined maximum size is reached. LSTMs have been used as generative models similarly in [4]. The idea is that, by simulating an event log from the RNN, we should ideally obtain an event log that is behaviourally highly similar to the event log that we started from. That is, we expect that, even when leaving out one single variant, the RNN should be able to (1) generalise this variant from the observed variants in the Training log, (2) avoid creating variants that were not observed in the original event log (Train+Test), and (3) contain all the variants present in the Training log.

3.2 Metrics

Accordingly, we define novel recall, precision and generalisation metrics that can quantify these three criteria. Based on the *Training* (including *Validation*), *Test* and *Simulated* logs, we define the following metrics:

$$Fitness = \sum_{v \in Var(Tr)} \frac{Min\,(Occ(v, Sim), Occ(v, Tr))}{|Tr|} \tag{1}$$

$$Precision = \sum_{v \in Var(Sim)} \frac{Min\,(Occ(v, Sim), Occ(v, Tr + Te))}{|Sim|} \tag{2}$$

$$Generalisation = \sum_{v \in Var(Te)} \frac{Min\,(\mathrm{Occ}(v, Sim), Occ(v, Te))}{|Te|} \tag{3}$$

with $|L|$ denoting the number of traces in an event log L, *Var(L)* denoting the set of variants of an event log L and *Occ(v,L)* a function denoting the frequency or multiplicity of a variant v in an event log L.

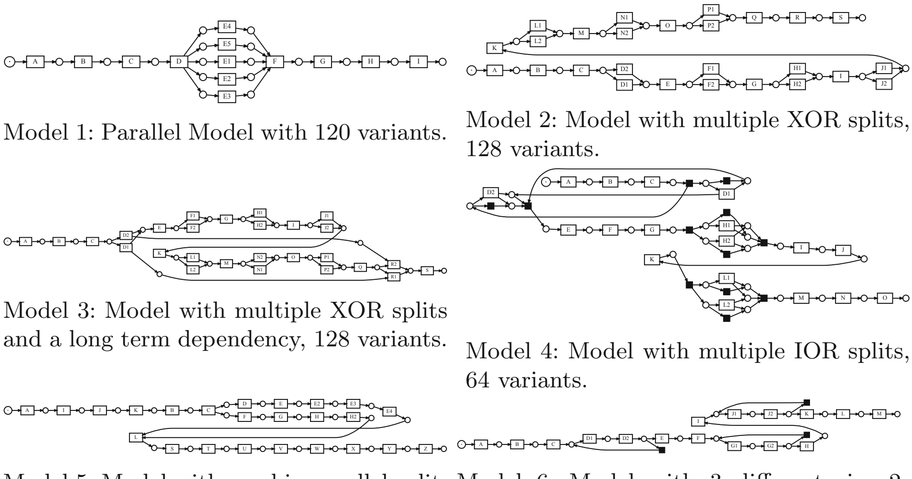
Each of these metrics outputs a value between 0 and 1. Beware that these metrics make use of nominal counts, so they only make sense when the original *Train+Test log* and the *Simulated log* contain the same amount of traces. If not, the metrics will have to be corrected. First of all, the fitness metric measures to what extent all of the variants present in the *Training log* are also present in the *Simulated log.* This is because we want the RNN to learn and replicate all of the behaviour found in the *Training log.* Moreover, we expect that the frequency of each variant in *Simulated log* is, more or less, equal to the frequency of observation of that variant in the *Training log.* Therefore, the fitness measure will punish if a certain variant is under-represented in the *Simulated log.* Secondly, the precision metric measures whether the RNN allows for too much behaviour, i.e. traces that have not been seen in *Train+Test log.* Moreover if certain correct variants are over-represented in the *Simulated log* the precision will also decrease. Finally, the generalisation metric quantifies to which extent the RNN is able to generalise, i.e. whether it is able to learn and reproduce correct but unseen behaviour. Therefore, the metric measures whether the frequency of occurrence of the unseen variant(s) in the *Test log* is actually reproduced to the same level in the *Simulated log.*

4 Experimental Evaluation

With the introduction of our assessment framework, consisting of a variant-level resampling procedure combined with these three metrics, we can now devise an experimental setup to evaluate the generalisation capacity of RNNs. While it is theoretically possible to perform such a assessment using complex artificial and even real-life logs and models, we opt to focus on simple models, as these provide a *sine qua non* condition in terms of investigating whether such models can deal with the aforementioned control-flow patterns at all.

4.1 Process Models

Hence, we generated artificial process models that represent main modelling constructs. In order to obtain the full event log (i.e. the *Train+Test log*), we use the play-out functionality of the Python Process Mining library PM4PY [2]. The models are depicted as Petri nets in Fig. 2. Model 1 is a simple linear model with a parallel gateway consisting of five parallel branches containing each one single activity. This process has 120 (equally likely) control-flow variants. In Model 2, a process model with 128 (equally likely) control-flow variants is created by sequencing seven exclusive OR (XOR) splits. Similarly, Model 3 consists out of eight XOR splits, but with a long-term dependency added in. Model 4 consists of three inclusive OR (IOR) splits, where at least one, but possibly both of the two activities have to occur. This leads to in total 64 control-flow variants. Model 5 shows a process which has two parallel paths, consisting of five activities each, leading to 125 control-flow variants with varying likelihood. Finally, Model 6 shows a process with three different possible loops (containing two activities each). The amount of possible control-flow variants is technically unlimited, so that we restrict each marking to be visited a maximum of three times, we keep 27 different variants.



Model 1: Parallel Model with 120 variants.

Model 2: Model with multiple XOR splits, 128 variants.

Model 3: Model with multiple XOR splits and a long term dependency, 128 variants.

Model 4: Model with multiple IOR splits, 64 variants.

Model 5: Model with one big parallel split, 126 variants.

Model 6: Model with 3 different size 2 loops.

Fig. 2. Process models used in the experimental evaluation.

4.2 Hyperparameter Search

The generalisation capacity of RNNs strongly depends on tuning its hyperparameters. This is an essential part of the training procedure. An overview of the investigated hyperparameters can be found in Table 1. A maximum prefix length of size 10 was used (with longer prefixes left-truncated), unless explicitly mentioned otherwise. The model's weights are optimised using the Adam [8] optimiser with a mini-batch size of 128 prefixes, using a starting learning rate of 0.005. The learning rate is decreased when the accuracy on the validation set has not decreased for over 10 epochs and the training is stopped (early stopping) when the accuracy has not increased for over 30 epochs (or when a maximum of 600 epochs is reached). The loss function used is categorical crossentropy. For clarity, the RNN is trained optimising accuracy in a "classical" sense, i.e. whether the activity predicted by the model to be the most probable activity is actually correct. As mentioned earlier, the usage of an embedding layer is the first binary hyperparameter. Where applicable, the dimension of the embedding was set to $\lceil \sqrt[4]{\text{Act. Voc. Size}} \rceil$, as was done in [4] (note that the embeddings were pretrained independently from the RNN in that work, though we used this value as a starting point in this investigation). The number of stacked LSTM layers is varied between one and two, with the layers' hidden dimension size set to 16, 32 or 64 units. Furthermore, we experiment with different values for overfitting countermeasures like regularisation and dropout. We try five different values for *L1* (*Lasso*) and *L2* (*Ridge*) regularisation [20], which adds a small penalty to the model based on the absolute or squared value of the weights respectively. In order to limit the scope of our grid we change $L1$ and $L2$ together. Lastly we also added the dropout hyperparameter [14], in which, per epoch, a fraction of the nodes is selected to be ignored during training, reducing the likelihood of overfitting to the training data. In this paper we experiment with multiple dropout values (including no dropout). There are multiple ways to introduce dropout when using RNNs related to the internal structure of such units; in this work we chose to add dropout to the output of the LSTM layer (note that dropout on the inputs would lead to removing certain steps of a prefix which is undesirable in our setting). We omitted the use of batch normalisation due to its limited effectiveness when applied to RNNs [5]. In future work, however, we could explore the effectiveness of *Recurrent Batch Normalisation* [5] and *Layer Normalisation* [1]. The neural network implementation was created using the Python library Keras[2].

Table 1. The hyperparameter values used in the grid search.

Hyperparameter	Values
Embedding Layer	Yes, No
Number of LSTM Layers	1, 2
LSTM Layer Size	16, 32, 64
L1 and L2	0.0, 0.00001, 0.0001, 0.001, 0.01
Dropout	0.0, 0.2, 0.4

[2] https://keras.io.

This led to 180 different hyperparameter configurations to iterate over. Based on some preliminary exploration, it was noticed that an RNN, without explicit overfitting countermeasures like regularisation and dropout, struggled to generalise for the process in Model 1. In order to contain computational time, and because the goal is to obtain a RNN that is able to generalise different types of behaviour at the same time, we opted to only perform one grid search on the event log obtained from this process model. The best hyperparameter settings for Model 1 were subsequently used and applied to all other models. For obtaining optimal hyperparamaters using Model 1, we conducted a tailored leave-one-variant-out cross-validation (LOVOCV) procedure as introduced above. More specifically, we generated a *Train+Test log* consisting of 12.000 traces for each model. In each LOVOCV iteration, we singled out all cases pertaining to one single variant into the *Test log*. For every hyperparameter combination, we performed the tailored LOVOCV eight times, each time with a different variant in the test set. In each iteration, we obtained a *Simulated log* of equal size of the original *Train+Test log*. Based on these eight LOVOCV iterations, we calculated the three different metrics defined above, and, for each setting, took the average over the eight iterations. Note that we use the *Test log* for this hyperparameter tuning, rather than the *(cross) validation log* as is usual. Since we are not trying to compare the predictive quality of different approaches as such, this is justified. We choose to continue working with the setting showing the highest average score over all three metrics, i.e., no embedding layer, one LSTM layer of hidden size 32, an *L1* and *L2* of 0.001 and a *dropout* of 0.4. Because this model was only trained on the data of one simple process model, and the differences between certain settings were slim, we however do not want to claim this setting is ideal for each predictive process monitoring problem. However in the context of this investigation rather than optimisation experiment, we continue with this setting in the rest of the paper. Not shown here, but apparent from the hyperparameter search was: (i) for all three metrics a regularisation value of 0.01 resulted in low scores, (ii) no or limited application of overfitting countermeasures leads to high fitness and precision (as expected) but weak generalisation scores. We like to stress that it is therefore of the utmost importance to tune the hyperparameters correctly when training RNNs.

4.3 Results

Subsequently, we then repeat the experiment using the settled hyperparameter configuration for all models. For each of these, we applied a LOVOCV setup again, working with *Test logs* containing a single variant. As for some models the frequency of the variants is not evenly distributed and to obtain more robust results, we now conducted an exhaustive LOVOCV, i.e. the procedure is repeated as many times as the number of variants in the event log, so that every variant is used once to form the *Test log*. In the case of Model 6 (loops), we restricted the analysis to variants for which the loop is taken a maximum of three times (27 variants). This was however not restricted neither in the play-out itself, nor in the simulation with the trained LSTM, leading to more different variants in these logs. In the experiment for Model 3 (including the long-term dependency) the prefix length of the input of the RNN was set to the maximum trace length minus one instead of the default of 10 because it needed to be long enough in order to have a chance of dealing with the long-term dependency. The average values over all variants for each of the three metrics can be found on the left side of Table 2. The error intervals are calculated by taking the standard deviation over all the metric values. Various interesting observations can be made from the results. First, it can be noticed that models with parallel behaviour (Model 1, 4 and 5) are problematic for the LSTMs. One can observe some level of generalisation, but given the extremely lenient LOVOCV-setup, it is remarkable that the generalisation scores go well below 0.80. On the other hand, the LSTMs show to be much more robust when dealing with XOR-splits (Model 2 and 3) and loops (Model 6). Also the long-term dependency in Model 3 seems to be handled well by the LSTM models. The standard deviations are significantly higher for the generalisation scores as this metric seems to be more prone to fluctuations. This is most likely due to the changing *Test log*, though should be further investigated.

Up until this point we have used the most trivial setting, i.e. the leave-one-variant-out *Test log*. However, one might expect that LSMTs should be able to cope with larger fractions of unseen behaviour. Therefore, this final evaluation part addresses the use of larger test sets, in particular, leaving out 20% of the control-flow variants from the *Training log*. We repeated this three times, with each time 20% randomly selected variants. The average of the metric values for each of these experiments can be found on the right side of Table 2. The error intervals are calculated by taking the standard deviation over the metric values for the three different experiments. When comparing this with the results from the LOVOCV experiment we can see that for some models, precision seems the decrease a bit. This might suggest that because the model has less behaviour to learn the correct process model structure from, it fares worse, allowing for some extra incorrect behaviour. Fitness seems not to be affected. Generalisation also results in lower values when having a more diverse *Test log*. This is especially apparent in the process models which already yielded low generalisation scores above, like Model 4 and 5. Again the standard deviation is higher when calculating the generalisation, because it is highly dependent on which variants exactly were in the *Test log*, and of the apparently more fluctuation-prone behaviour of this metric.

5 Discussion

From the hyperparameter tuning, it appears that overfitting countermeasures like regularisation and dropout are important parameters for constructing predictive RNNs. However, precise tuning is crucial, and we therefore urge other researchers to perform a good hyperparameter experiment when training RNNs on predictive process monitoring tasks. This may require the inclusion of resampling methods as introduced here, as opposed to only holding out randomly

Table 2. The results on the different process models, averaged over all leave-one-variant-out experiments with every different control flow variant. And the results on the different process models when taking a *Test log* consisting of 20% of the control flow variants. Average over three different randomly selected *Test logs*.

	LOVOCV			Leave 20% out		
Model	**Prec.**	**Fit.**	**Gen.**	**Prec.**	**Fit.**	**Gen.**
Model 1	0.94 ± 0.00	0.94 ± 0.00	0.79 ± 0.12	0.89 ± 0.01	0.93 ± 0.00	0.72 ± 0.04
Model 2	0.94 ± 0.00	0.94 ± 0.00	0.92 ± 0.09	0.92 ± 0.01	0.93 ± 0.01	0.89 ± 0.04
Model 3	0.94 ± 0.00	0.94 ± 0.00	0.91 ± 0.10	0.92 ± 0.02	0.93 ± 0.01	0.85 ± 0.05
Model 4	0.95 ± 0.01	0.95 ± 0.00	0.75 ± 0.13	0.87 ± 0.03	0.94 ± 0.01	0.61 ± 0.14
Model 5	0.92 ± 0.01	0.92 ± 0.01	0.68 ± 0.21	0.84 ± 0.01	0.94 ± 0.01	0.47 ± 0.05
Model 6	0.93 ± 0.01	0.93 ± 0.01	0.92 ± 0.11	0.92 ± 0.01	0.93 ± 0.00	0.83 ± 0.12

selected prefixes. Please observe that the "best" hyperparameters were selected on the test observations for Model 1. It can be expected that standard tuning on a random validation set and thereby only optimising accuracy of next events (and not the metrics presented here), is unlikely to result in the same outcome.

LSTMs seem to be less suited for generalising process models with parallel behaviour. When the degree of incompleteness is increased, i.e. the amount of variants not seen by the RNN during training is expanded, LSTMs seem to struggle more. The generalisation, as well as the precision, decreased when increasing the amount of variants in the test set, when compared with the LOVOCV experiments.

It could be interesting to investigate to what extent the generalisation and overfitting problems could affect predictions of real-life processes, considerably more complex than the artificially created data discussed in this work. Extra overfitting measures may need to be included as well in the future. One option could be found in using a similar resampling method as used here in order to construct the *Test log*, or a hybrid approach, to create the *Validation log*. Another course of action would be to alter the loss function used to train the RNN. Important to check further is whether low generalisation scores of merely memorising and overfitting models also lead to less accurate next event predictions. We still assume proper generalisation would be beneficial in predictive task, especially with increasing complexity, and theoretically the power of deep learning models, actually lies in their generalisation capability [7]. Moreover if overfitting would not be an issue, more explainable statistical models could easily be found as well, as opposed to the black-box LSTMS, with comparable accuracy, and preference should be given to these more explainable models.

6 Conclusion and Future Work

This paper addressed the so far largely uninvestigated problem of neural networks' capability to learn the behaviour of the underlying process behind an event log. By introducing a new framework that combines a variant-level resampling scheme with three novel metrics, we were able to investigate to what extent

LSTMs trained to predict the next event of a process execution are general, fit and precise. By applying this framework on several simple process models, it was shown that LSTMs can only generalise parallel behaviour to a certain extent. Even in the most lenient setting, the LOVOCV, the generalisation metric does not return values close to optimal. When increasing the amount of variants in the *Test log*, unseen by the LSTM during training, generalisation decreases further, as well as precision. This paper opens up the door for future work in predictive process monitoring to include explicit generalisation checks as well, in model selection, hyperparameter tuning and testing.

The experiments in this work were limited to only use control-flow behavior. However, since the generalization behavior investigated in this paper is only control-flow like, this choice is justified. Nevertheless, it might be interesting to investigate the effects of adding more dimensions (like timestamps and resource) to the predictive models. In future work, more synthetic logs could be investigated deepening the relation between process model behaviour and RNN generalisation. A more rigorous theoretical elaboration, similar to [17], might provide some interesting insights as well. Furthermore, it should be investigated whether new recommendations can be proposed regarding how to optimally sample training and validation sets. Also, it would be interesting to apply these metrics on models trained on real-life event logs, as opposed to only synthetic data. In this paper it was opted to first work with synthetic data since it would allow us to test different types of behaviour independently. However real-life logs are in general more complex, and in this way might present other difficulties. When doing this, it might be useful to expand the hyperparameter grid and include extra parameters such as Layer Normalization [1] and Recurrent Batch Normalization [5], and to ameliorate the overfitting measures applied in this work. Other future work can address a comparison of results presented here with the findings in recently proposed work by [19], and similarly test multiple encoding techniques like hash encoding. Moreover, an additional investigation of the attention mechanism seems worthwhile [3]. In addition, a similar experiment could be applied to alternative architectures including Convolutional Neural Networks, Transformer Networks, and GAN-style approaches. This work was also limited to only include next-event prediction models, thus it might be useful to expand the metric definitions to also be able to evaluate full suffix prediction (or even remaining time prediction). Finally, the application of neural networks to formal languages has been investigated in other domains, e.g. in [10], which could lead to more fundamental research on grammar learning capabilities of RNNs.

References

1. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization (2016)
2. Berti, A., van Zelst, S.J., van der Aalst, W.: Process mining for python (PM4Py): bridging the gap between process-and data science. In: Proceedings of the ICPM Demo Track 2019, co-located with 1st International Conference on Process Mining (ICPM 2019), Aachen, Germany, June 24–26, 2019. pp. 13–16 (2019)

3. Bukhsh, Z.A., Saeed, A., Dikman, R.M.: Processtransformer: predictive business process monitoring with transformer network. CoRR abs/2104.00721 (2021). https://arxiv.org/abs/2104.00721
4. Camargo, M., Dumas, M., González-Rojas, O.: Learning accurate LSTM models of business processes. In: Hildebrandt, T., van Dongen, B.F., Röglinger, M., Mendling, J. (eds.) BPM 2019. LNCS, vol. 11675, pp. 286–302. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26619-6_19
5. Cooijmans, T., Ballas, N., Laurent, C., Courville, A.C.: Recurrent batch normalization. CoRR abs/1603.09025 (2016). http://arxiv.org/abs/1603.09025
6. Evermann, J., Rehse, J.R., Fettke, P.: Predicting process behaviour using deep learning. Decis. Support Syst. **100**, 129–140 (2017)
7. Kawaguchi, K., Kaelbling, L.P., Bengio, Y.: Generalization in deep learning (2020)
8. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings (2015). http://arxiv.org/abs/1412.6980
9. Lin, L., Wen, L., Wang, J.: MM-Pred: A Deep Predictive Model for Multi-attribute Event Sequence, pp. 118–126 (2019)
10. Michalenko, J.J., Shah, A., Verma, A., Baraniuk, R.G., Chaudhuri, S., Patel, A.B.: Representing formal languages: a comparison between finite automata and recurrent neural networks. CoRR abs/1902.10297 (2019). http://arxiv.org/abs/1902.10297
11. Moreira, C., Haven, E., Sozzo, S., Wichert, A.: Process mining with real world financial loan applications: improving inference on incomplete event logs. PLOS ONE **13**, e0207806 (2018)
12. Schäfer, A.M., Zimmermann, H.G.: Recurrent neural networks are universal approximators. In: Kollias, S.D., Stafylopatis, A., Duch, W., Oja, E. (eds.) ICANN 2006. LNCS, vol. 4131, pp. 632–640. Springer, Heidelberg (2006). https://doi.org/10.1007/11840817_66
13. Siegelmann, H.T., Sontag, E.D.: On the computational power of neural nets. J. Comput. Syst. Sci. **50**(1), 132–150 (1995)
14. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**, 1929–1958 (2014)
15. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive Business process monitoring with LSTM neural networks. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 477–492. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_30
16. Taymouri, F., Rosa, M.L., Erfani, S., Bozorgi, Z.D., Verenich, I.: Predictive business process monitoring via generative adversarial nets: the case of next event prediction. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) BPM 2020. LNCS, vol. 12168, pp. 237–256. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58666-9_14
17. Tu, Z., He, F., Tao, D.: Understanding generalization in recurrent neural networks. In: International Conference on Learning Representations (2020). https://openreview.net/forum?id=rkgg6xBYDH
18. Vaswani, A., et al.: Attention is all you need (2017)

19. Weinzierl, S., et al.: An empirical comparison of deep-neural-network architectures for next activity prediction using context-enriched process event logs. CoRR abs/2005.01194 (2020). https://arxiv.org/abs/2005.01194

20. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society. Series B (Statistical Methodology) **67**(2), 301–320 (2005). http://www.jstor.org/stable/3647580

Remaining Time Prediction for Processes with Inter-case Dynamics

Mahsa Pourbafrani[1](✉), Shreya Kar[1], Sebastian Kaiser[2], and Wil M. P. van der Aalst[1]

[1] Chair of Process and Data Science, RWTH Aachen University, Aachen, Germany
{mahsa.bafrani,wvdaalst}@pads.rwth-aachen.de, shreya.kar@rwth-aachen.de
[2] LMU Munich, Munich, Germany
sebastian.kaiser@stat.uni-muenchen.de

Abstract. Process mining techniques use event data to describe business processes, where the provided insights are used for predicting processes' future states (*Predictive Process Monitoring*). *Remaining Time Prediction* of process instances is an important task in the field of Predictive Process Monitoring (PPM). Existing approaches have two key limitations in developing *Remaining Time Prediction Models* (RTM): (1) The features used for predictions lack process context, and the created models are black-boxes. (2) The process instances are considered to be in isolation, despite the fact that process states, e.g., the number of running instances, influence the remaining time of a single process instance. Recent approaches improve the quality of RTMs by utilizing process context related to *batching-at-end* inter-case dynamics in the process, e.g., using the time to batching as a feature. We propose an approach that decreases the previous approaches' reliance on user knowledge for discovering fine-grained process behavior. Furthermore, we enrich our RTMs with the extracted features for multiple performance patterns (caused by inter-case dynamics), which increases the interpretability of models. We assess our proposed remaining time prediction method using two real-world event logs. Incorporating the created inter-case features into RTMs results in more accurate and interpretable predictions.

Keywords: process mining · predictive process monitoring · remaining time prediction · inter-case dynamics behavior

1 Introduction

Remaining time prediction approaches learn from historical process executions and build prediction models for running process instances, i.e., cases, based on

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2023 Internet of Production- Project ID: 390621612. We also thank the Alexander von Humboldt (AvH) Stiftung for supporting our research.

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 140–153, 2022.
https://doi.org/10.1007/978-3-030-98581-3_11

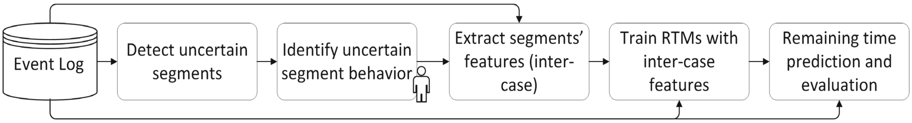

Event Log
Detect uncertain segments
Identify uncertain segment behavior
Extract segments' features (inter-case)
Train RTMs with inter-case features
Remaining time prediction and evaluation

Fig. 1. Our proposed framework for inter-case-aware RTMs. Patterns are discovered after detecting uncertain segments, i.e., segments causing high prediction errors due to inter-case dynamics. RTMs are trained using the extracted features from the patterns within uncertain segments.

the extracted features from the event data. Many approaches have been suggested to solve the remaining time prediction problem [17]. However, most proposed approaches have considerably high prediction errors. Based on [17], the best performing model using an LSTM neural network [10] showed a prediction error of 178.4 days on average for the Road Traffic Management (RF) event log [9]. These approaches also only consider control-flow-related aspects of processes and individual case properties, i.e., intra-case properties, while making predictions [12]. A process also has other dimensions associated with it [13]. For instance, specific rules determining scheduling and assignment of limited resources, queuing mechanism, and decision logic in the process create inter-case dependencies within the performance of process instances. Moreover, most of the effort put into this research area has focused on applying new predictive modeling techniques, which create black-box prediction models. Considering inter-case along with intra-case process features in RTMs increases the explainability, interpretability, and accuracy of the prediction [8]. Therefore, we aim to improve the quality of RTMs and introduce more interpretability in the predictions. The accuracy of a RTM which is unaware of inter-case behavior is substantially impacted if cases in a process segment, i.e., a pair of related activities, are processed in a batch, First-In-First-Out (FIFO), or other patterns. The prediction accuracy decreases as a case passes through such segments indicating that RTM is uncertain about the underlying process behavior in such segments. We call these process segments uncertain segments. Therefore, recognizing all uncertain segments and translating their various inter-case patterns of process execution into features for training RTMs increases prediction quality.

In this paper, we present a three-step approach for developing inter-case dynamics aware RTMs: (1) Identifying process segments that cause high prediction errors due to inter-case dynamics, i.e., uncertain segments. (2) Discovering insights about the underlying patterns, e.g., *batching*, that leads to inter-case dependencies within the detected segments. (3) Transforming derived insights into features and incorporating them in RTMs to improve the quality of predictions. For instance, the waiting time for the batching in a segment is transformed into a feature and introduced into the RTM. We evaluate the prediction errors of RTMs without incorporating inter-case dependencies, such as batching behavior in a process segment, as shown in Fig. 1, and identify uncertain segments that involve inter-case dynamics. We continue by extracting the features associated with the observed patterns in the uncertain segments.

We introduce preliminaries and the related work in Sect. 2. In Sect. 3, we present our main approach. We evaluate the approach in Sect. 4 using real event logs, and Sect. 5 concludes this work.

2 Preliminaries and Related Work

In this section, we introduce the necessary concepts and related work required to understand the approach presented in this paper.

2.1 Related Work

RTM approaches can be classified into three broad categories [17]. *Process aware approaches* make predictions using explicit process model representations such as transition system [1]. *Process agnostic approaches* typically use machine learning (ML) methods [14] to make predictions. Recent process agnostic approaches predominantly make use of sophisticated neura!l network architectures like LSTM [16] and explainable AI methods [5] to develop RTMs. *Hybrid approaches* like [11] combine capabilities of both categories by exploiting transition systems that are annotated using a machine learning algorithm. However, most approaches across all three categories only consider the intra-case perspective for predictions.

RTM approaches based on queuing models [15] and supervised learning [14] utilized the inter-case dimension in predictions. They create features on the basis of queuing theory like case priority and open cases of similar type. However, these approaches assume FIFO queuing behavior throughout the entire process. Two recent PPM approaches [3,8] use performance spectra [2] to learn inter-case dynamics present in the process without any prior assumption. Denisov et al. [3] presented a novel approach to predict the aggregated performance of non-isolated cases that utilize performance-related features. Klijn et al. [8] presented a novel RTM approach that is aware of *batching-at-end* dynamics. In this paper, we extend the process agnostic RTM approach presented in [8] by considering inter-case dynamics caused by *non-batching*, *batching-at-start* patterns too. We use and improve the fine-grained error analysis technique proposed in [8] to identify inter-case dynamics by limiting manual intervention.

2.2 RTM Background

RTM approaches predict the remaining time to completion of an ongoing process instance, i.e., *case*, based on process execution data of completed cases. Process execution of a completed case is recorded as a non-empty sequence of events (e), i.e., $\sigma = \langle e_1, .., e_n \rangle$ or *trace*. An event log L is a set of completed traces. Let $\mathcal{A}, \mathcal{T}, \mathcal{E}$ be the universe of activities (event classifiers), timestamps and events. Each event $e \in \mathcal{E}$ consists of mandatory and additional attributes. Let AN be the set of attribute names. For $an {\in} AN$, we define $\#_{an}(e)$ as the value of attribute an for event e. An event e has mandatory attributes timestamp $\#_t(e) {\in} \mathcal{T}$ at which e occurs and activity $\#_{act}(e) {\in} \mathcal{A}$ that occurs during e.

We first need to understand the general steps to develop a RTM described in [17]. In the *offline* or training phase, the first step is to prepare the input data, i.e., event log. Since a RTM makes prediction for incomplete traces, it trains on *prefixes* extracted from traces in L. A prefix is extracted by taking the first $k \in \mathbb{N}$ events from a completed trace ($\sigma = \langle e_1, .., e_n \rangle$) using function $hd^k(\sigma) = \langle e_1, .., e_k \rangle, k \leq n$. The resulting prefixes are collectively known as a *prefix log* L^* of L. Therefore, data preparation includes cleaning the data, creating a prefix log and feature engineering. Features like *weekday* or *sojourn time* are extracted from event data and categorical features are encoded.

A RTM can be instantiated based on three main parameters, methods for grouping similar prefixes into buckets, prefix encoding methods, and used prediction techniques. For instance, $RTM = (p, a, x)$ represents that the model's prefix bucketing method is based on similar prefix lengths (p), the encoding method is aggregating data of all prefix events (a), and ML algorithm is XGBoost (x). After training, the models are tuned using techniques like hyperparameter optimization. Finally, the optimal model's prediction accuracy is evaluated using aggregated metrics, e.g., Mean Absolute Error (MAE).

2.3 Performance Spectrum with Error Progression

To identify process segments subject to high prediction errors due to inter-case dynamics, Klijn et al. [8] introduced a visual analysis technique, *Performance Spectrum with Error Progression (PSwEP)*. It uses the performance spectrum (PS) [2], which maps the performance of each case passing through a segment over time. A process segment $(a, b) \in \mathcal{A} \times \mathcal{A}$ can be defined as any two successive steps in the process, e.g., a step from activity a to activity b. For traces of form $\langle ..., e_i, e_{i+1}, ... \rangle$, where $\#_{act}(e_i) = a$, $\#_t(e_i) = t_a$, $\#_{act}(e_{i+1}) = b$, and $\#_t(e_{i+1}) = t_b$, we observe an occurrence of a segment (a, b) from time t_a to t_b. Each occurrence of segment (a, b) representing a case is plotted in a PS as a line from (t_a, a) to (t_b, b). In PSwEP, segment occurrences within a PS are classified based on the error progression of the case while passing through the segment. Let $\mathcal{P}$ be the set of predictions made on test data using RTM. Each prediction $pr_k \in \mathcal{P}$ corresponds to a prediction made for prefix $hd^k(\sigma) = \langle e_1, .., e_k \rangle$ at point of prediction $\#_{act}(e_k) = a_k$ and $t_{pr_k} = \#_t(e_k)$, i.e., the time moment of prediction.

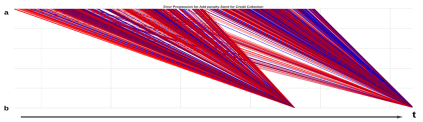


Fig. 2. PswEP for *(Add Penalty (AP), Send for Credit Collection (SC))* in RF: error decrease (red), error increase (blue).(Color figure online)

y_{pr_k} and $\overline{y_{pr_k}}$ denote the actual and predicted outcomes of pr_k. To measure the error progression of segment occurrence (a_k, a_{k+1}) linked to σ, the prediction errors at a_k and a_{k+1} are compared. The difference in relative absolute errors $DRAE(rae_k, rae_{k+1}) = rae_k - rae_{k+1}$ with $rae_k = |\overline{y_{pr_k}} - y_{pr_k}| / y_{pr_k}$ is measured. If the prediction error decreases for a segment occurrence, i.e., $DRAE > 0$ this plotted line is colored

red in the PSwEP. If the prediction error increases, i.e., $DRAE < 0$ the line is colored blue. Figure 2 shows PSwEP of segment *(Apply Penalty (AP), Send for Credit Collection (SC))* in the RF event log.

3 Approach

In this section, we will discuss the main approach proposed to develop an inter-case-dynamics-aware RTM. In Sect. 3.1, we discuss the proposed techniques to automatically identify uncertain segments. In Sect. 3.2, we discuss the process of identifying and deriving insights about inter-case dynamics. Finally, in Sect. 3.3, we propose ways to create inter-case features by utilizing derived insights.

3.1 Detecting Uncertain Segments

Measuring Uncertainty of a Process Segment. To identify uncertain segments, we need to measure the uncertainty of each process segment. To do so, we first measure the $DRAE$ (Sect. 2.3) of individual segment occurrences linked to predictions made using RTM on test data. Table 1 shows an example of how individual predictions are aligned with segment occurrences and the error progression of each occurrence is classified. A *decrease* in error, i.e., $DRAE > 0$ for a case passing through segment (a, b) implies that after the occurrence of activity b the remaining time prediction improves. This decrease could indicate some uncertainty between activity a and b, which gets resolved after activity b completes. An *increase* in error implies that after the occurrence of activity b, the prediction model becomes more unsure about how the partial trace will proceed. If prediction error remains the same, i.e., $DRAE = 0$, there is no clear indication of uncertainty within the process segment. We can either ignore such rare cases or include them as error *decrease*, where we consider the latter.

Based on above insights, we use three aggregated metrics to quantify uncertainty of segments. For each segment (S) linked to $\mathcal{P}$, we measure (1) *observations* or total occurrences linked to S in $\mathcal{P}$, (2) *decrease cases* or total occurrences linked to S with $DRAE \geq 0$, and (3) *increase cases* or total occurrences linked to S with $DRAE < 0$. Table 2 is the result of applying the above aggregations to occurrences of segments found in Table 1.

Table 1. Error progression for the occurrence of segments linked to predictions.

Case ID	Prefix	t_{pr_k}	y_{pr_k}	$\overline{y}_{pr_k}$	rae	Segment	$DRAE$	Error Progression
c1	$\langle a\rangle$	1	6	10	0.667			
c1	$\langle a, b\rangle$	2	5	2	0.600	(a, b)	0.007	decrease
c1	$\langle a, b, c\rangle$	4	3	2	0.333	(b, c)	0.267	decrease
c1	$\langle a, b, c, d\rangle$	4	3	2	0.333	(c, d)	0	same
c1	$\langle a, b, c, d, e\rangle$	7	0	0	∞	(d, e)	$-\infty$	increase
c2	$\langle a\rangle$	3	11	14	0.272			
c2	$\langle a, b\rangle$	5	9	14	0.555	(a, b)	-0.283	increase
c2	$\langle a, b, c\rangle$	14	2	3	0.500	(b, c)	0.055	decrease

Table 2. Measuring uncertainty of each segment by aggregating its occurrences to calculate *observations*, *decrease cases*, and *increase cases*.

Segment	Observations	Decrease Cases	Increase Cases
(a, b)	2	1	1
(b, c)	2	2	0
(c, d)	1	1	0
(d, e)	1	0	1

Selecting the Most Uncertain Segments. We define a mapping function $u_S : \mathbb{N} \times \mathbb{R} \longrightarrow [0, 1]$ to select a subset of process segments for which inter-case features could be created (Eq. 1). The inputs are the number of observations (o) and the ratio $r = d/max(1, i)$ of decrease cases (d) to increase cases (i) for segment S (as shown in Table 2). Output 1 indicates the segment is highly uncertain. Note that ideal candidates for uncertain segments are those where decrease cases are almost the same or more than increase cases, i.e., their ratio should be greater than some threshold t_r. The threshold for the number of observations (t_{obs}) indicates the occurrences of the segments. These thresholds can be set for each process individually.

$$u_S(o, r) = \begin{cases} 1 & \text{if}\, o \geq t_{obs} \; and \; round(r) \geq t_r \\ 0 & otherwise \end{cases} \tag{1}$$

Let SG be the set of all segments in a process and SG_{start} be the set of starting segments. Therefore, we apply u_S to $S \in SG \setminus SG_{start}$ based on some t_r and t_{obs} and select set of segments U for which $u_S(o, r) = 1$. Removing starting activities in traces is due to the fact that the RTM has too little information, and the prediction error is likely to decrease when the second activity occurs. We use the RF event log [9] as the running example. First, predictions are made on the last 20% (temporally split) of the event log using a RTM, here $RTM = (p, a, x)$. Then, these predictions are used to measure the uncertainty of each process segment and u_S is applied to all non-starting segments. We set $t_r = 1$ and $t_{obs} > \mu$, e.g., $t_{obs} = 2 * std$ where μ, std are the mean and standard deviation of segment occurrences. The selected uncertain segments are *(Send Fine (SF), Insert Fine Notification (IF)), (Insert Fine Notification (IF), Add Penalty (AP))* and *(Add Penalty (AP), Send for Credit Collection (SC))*. The details of selecting the most uncertain segments presented here[1].

3.2 Identifying Inter-case Dynamics in Uncertain Segments

In order to diagnose causes for uncertainty within segments, first, we visualize the performance of cases within the process segment using PSwEP (Sect. 2.3). After that, the observed patterns in the performance spectrum are compared to a taxonomy [2] to identify underlying process behavior that causes inter-case patterns within the process segment. We explain the process of deriving insights for the uncertain segments identified in the running example.

In the shown PSwEP of (SF, IF) in Fig. 3 (left), two patterns, *batching-at-start* and *non-batching* FIFO behavior are identified. These are elementary patterns related to the order of case arrival. We notice uncertainty (as shown by the red lines) for non-batched cases. Therefore, RTM is currently not aware that non-batched cases are processed much faster than batch ones. Batched cases within the segment (Fig. 3) are also classified using red. The uncertainty concerning these cases is caused by the prediction model's lack of awareness

[1] https://www.pads.rwth-aachen.de/go/id/qcekn/lidx/1.

about *batching-at-start* dynamics. The order of lines in PSwEP of $(AP,\ SC)$ presented before in Fig. 2 clearly shows that the inter-case pattern is caused by *batching-at-end*. The prediction model is currently unaware of this inter-case dynamic within the process segment. In PSwEP of $(IF,\ AP)$ in Fig. 3 (right), we observe a FIFO with a constant time pattern in the order of case arrival. The performance of a case is strongly correlated to the previous case that passed through the segment. We also know that there are two possible activities, *Add Penalty (AP)* or *Insert Date Appeal to Prefecture (ID)*, that can occur after *Insert Fine Notification (IF)* and the time that cases wait within the segments is significantly different. Therefore, incorrectly assuming the path of a case arrives at IF impacts the remaining time prediction. We are able to predict the path by observing the recent performance of cases in $(IF,\ AP)$ and $(IF,\ ID)$ w.r.t. inter-case dependencies. Lastly, across three segments, we observe changing the density of lines indicating varying workloads.

Based on the above derived insights, we define the abbreviated inter-case pattern(s) identified for segments $(SF,\ IF), (IF,\ AP)$ and $(AP,\ SC)$ as $R_1 = non-batching,\ batch(s)$, $R_2 = non-batching$ and $R_3 = batch(e)$ respectively.

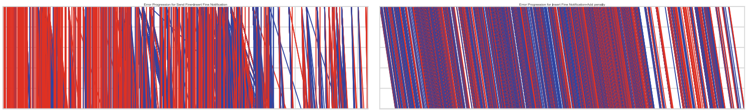


Fig. 3. PSwEP for segments *(Send Fine (SF), Insert Fine Notification (IF))* (left), and *(Insert Fine Notification (IF), Add Penalty (AP))* (right) in the RF event log.

Table 3. The created inter-case features for segment predictions ($\mathcal{C} = \{C_S, C_{S_1}, C_{S_2}, C_{S_3}\}$) and waiting time ($w$) within uncertain segments for the RF event log.

Case ID	Activity	Timestamp	...	C_S	C_{S_1}	C_{S_2}	C_{S_3}	w	y
N71924	SF	09-17 08:00	...	1	1	0	0	1154258.7	39229200.0
S120874	AP	05-09 08:00	...	1	0	1	0	2808000.3	28080000.0
S86803	SF	11-03 09:00	...	1	1	0	0	1212661.0	36115200.0
S57422	SC	01-10 09:00	...	0	0	0	0	0.0	0.0
S70222	CF	09-29 08:00	...	0	0	0	0	0.0	40438800.0

3.3 Inter-case Feature Creation

As the running example shows, ignoring inter-case dynamics results in high prediction errors for prefixes expected to pass through segment $S \in U$. Therefore, we need to provide the RTM information about a prefix being subject to inter-case pattern R detected in uncertain segment S prior to the occurrence of the segment. We use these insights to develop inter-case features.

Consider the running example with three uncertain segments S_1, S_2, and S_3 with inter-case pattern(s) R_1, R_2 and R_3, respectively, we define the following inter-case features: (1) $C_S \in \{0, 1\}$, to indicate if a prefix passes through an uncertain segment $S \in U$, (2) $C_{S_1} \in \{0, 1\}$, to indicate that the prefix passes through S_1 with inter-case pattern(s) R_1, (3) $C_{S_2} \in \{0, 1\}$, to indicate that prefix passes through S_2 with inter-case pattern(s) R_2, (4) $C_{S_3} \in \{0, 1\}$, to indicate that prefix passes through S_3 with inter-case pattern(s) R_3, and (5) w, to indicate the waiting time of the prefix in $S \in U$, as a result of inter-case pattern(s) R. As a result of the feature creation step for the running example, Table 3 is generated showing inter-case features. These features are used to train an inter-case-dynamics-aware RTM. Feature y is the target feature, i.e., remaining time to completion.

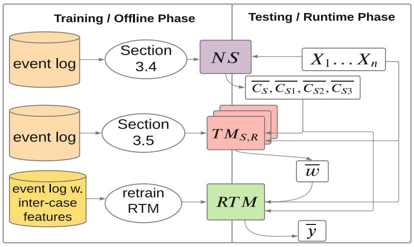


Fig. 4. The overview of feature creation process for RF event log with uncertain segments S_1, S_2 and S_3.

Creating inter-case features for an ongoing case at run-time requires its own prediction models. We need a model (NS) to predict inter-case features related to segment prediction and waiting time prediction model ($TM_{S,R}$) for each uncertain segment $S \in U$ with inter-case pattern(s) R. Figure 4 gives an overview of the steps involved in creating the models (offline) and utilization of these models to create inter-case features (at run-time). This process is the extended version of the presented feature-creation in [8].

3.4 Predicting the Next Segment

Classifier NS should determine if a prefix passes through segment $S \in U$ at the point of prediction. To build NS, we build a classifier for the next activity prediction using [18] and modify the outcome to predict the value of segment prediction inter-case features. Let $hd^k(\sigma)$ be the input prefix with last activity a for NS. If the next activity predicted is b, we say that the prefix passes through segment (a, b) at the point of prediction. If $(a, b) \in U$, then $\overline{C_S} = 1$, else we set it to 0. If $\overline{C_S} = 1$, we set the value of the boolean variable representing the prefix passing through segment (a, b) as 1. Therefore, if predicted $(a, b) = S_1$, then $\overline{C_{S_1}} = 1$, $\overline{C_{S_2}} = 0$, and $\overline{C_{S_3}} = 0$. The collective set of predicted features using NS is called $\overline{\mathcal{C}}$.

3.5 Predicting Waiting Time

In this section, we present general steps to create a waiting time prediction model ($TM_{S,R}$) that predicts how long a case stays in a segment S with inter-case patterns R. Consider a case c_1 arriving at segment $S=(a,b)$ at time t_a (Fig. 5). Because of inter-case dynamics, the waiting time w of c_1 will depend on the performance of other cases in relevant segments in some recent time interval, i.e., historic spectrum (S_h) [3] and relevant individual properties (intra-case features). The intra-case feature of c_1 and performance seen within S_h can be encoded as feature vector $X_1..X_n$ using insights gained about R within S. This allows us to formulate the waiting time prediction problem as a supervised learning problem: $w=f(X_1..X_n)+\varepsilon$, where function f predicts w from $X_1..X_n$. To learn f, we create training samples using the sliding window method and apply a ML method like LightGBM [6] that tries to minimize prediction error ε. Table 4 shows sample data used to train a $TM_{S,R}$ for *(IF, AP)*.

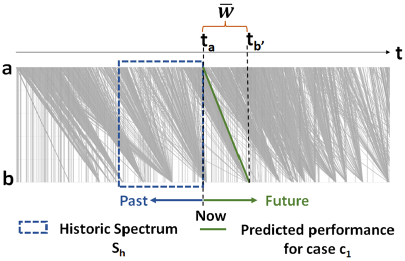


Fig. 5. Illustration of a single instance for $TM_{S,R}$ to learn waiting time for case c_1 using performance-related features extracted from S_h and relevant individual properties of c_1.

Waiting Time Prediction for Non-batching Dynamics. In Sect. 3.1, we learned that $\overline{w}$ of a case in (IF, AP) is influenced by $R=non-batching$ and varying workload in segments (IF, AP) and (IF, AD). To derive workload related context, we define h in S_h as the period between arrival of c_1 and the last case before it and derive: (1) *starting cases* or the number of cases that started (arrived at the segment) in period h, (2) *ending cases* or the number of cases that completed (exited the segment) in period h and (3) *pending cases* or the number of cases that have started within period h and will complete in the future. Since, performance of a case in (IF, AP) strongly depends on the previous case, we also extract last waiting time (w_l), e.g., Table 4.

Table 4. Sample data for training waiting time prediction model ($TM_{S,R}$) for uncertain segment (IF, AP) with pattern $R=non{-}batching$.

starting cases	ending cases	pending cases	w_l	w
60	37	60	5183000.0	5184000.0
14	10	14	5184000.0	5184000.0
19	17	18	5187000.0	5187000.0

Waiting Time Prediction for Batching-at-Start Dynamics. $\overline{w}$ of a case $c1$ arriving at *(SF, IF)* will depend on $R=batch(s), non-batching$ and varying workload within the segment. Therefore, S_h contains only segment *(SF, IF)*. To learn performance related to $R=non-batching$ and the workload, we include features presented in Sect. 3.5. To include features related to $R=batch(s)$, we extract features related to the previous batch [7] with batching moment BM_l: (1) least (w_{min}) and longest waiting time (w_{max}) in previous batch, (2) previous *batch size* and *batch size percentile*, (3) mean and standard deviation of $IBCT$ or *inter batch case completion time*, which is the time difference between the completion times of two successive cases in the batch and (4) *batch type*, which distinguishes batches with less than 2 observations that behave like simultaneous batches, and (5) CIA or *case inter-arrival time* which the time between arrival of c_1 and the case before it. We also include relevant intra-case features *resource, expense, points* and *weekday, month, hour* of previous batch. *Duration* or the waiting time of the case in the previous segment is also included to distinguish batched and non-batched cases. However, learning case-specific w is difficult because *batching-at-start* cases proceed randomly, i.e., not in the order they arrived at the batch. To avoid learning this random behavior, we propose building a $TM_{S,R}$ that predicts the average of expected waiting times for all cases that arrive along with c_1. Hence, the training data will be prepared by extracting the above-mentioned features and then aggregating (calculating mean) feature values for instances that correspond to cases arriving simultaneously in the segment.

Waiting Time Prediction for Batching-at-End Dynamics. *(AP,SC)* contains inter-case dynamics caused by $R = batch(e)$ and varying workload. To consider the varying workload across the segment, we include the features presented in Sect. 3.5. To learn batching related performance, we extract features w_{min}, w_{max} and CIA described in previous section. Additionally, we include: (1) t_{lb}: or the time elapsed since the occurrence of the last batch, (2): the mean and standard deviation of $IBIA$ *(inter-case arrival rate)* which is the difference between the arrival times of two successive cases in the batch. We also include intra-case features *month* and *weekday.*

4 Evaluation

4.1 Experimental Setup

We evaluate the proposed approach on two real-life event logs: the RF event log [9] and BPIC'20 event log [4]. We implemented inter-case feature creation and PSwEP in Python, which is publicly available[2]. To train and test RTMs, we use the benchmark implementation for RTM approaches[3] [17]. First, we make predictions with $RTM = (p, a, x)$ for both event logs to identify uncertain segments and their patterns. The uncertain segments identified from RF event log are *(SF, IF)*, *(IF, AP)* and *(AP, SC)* with inter-case pattern(s) $R_1 = non{-}batching$, $batch(s)$, $R_2 = non{-}batching$ and $R_3 = batch(e)$, respectively. The two identified uncertain segments from BPIC'20 event log are *(Declaration Final Approved by Administration (DF), Request Payment (RP))* and *(Request Payment (RP), Payment*

[2] https://github.com/karshreya98/Inter_case_aware_RTM.

[3] https://github.com/verenich/time-prediction-benchmark.

Table 5. Weighted average MAE (in days) of different RTM models with different bucketing, encoding and ML methods, e.g., (p, a, x), while using no inter-case features $I(\emptyset)$ and with the created inter-case features using segment predictions $I(\overline{\mathcal{C}}, \overline{w})$.

		(p,a,x)	(p,l,x)	(c,a,x)	(c,l,x)	(p,l,r)	(c,a,r)	(c,l,r)	(s,l,x)
RF	$I(\emptyset)$	212.60	209.69	210.32	208.59	221.39	221.05	221.53	203.29
	$I(\overline{\mathcal{C}}, \overline{w})$	**187.65**	**179.78**	**201.17**	**179.34**	**191.06**	**205.87**	**190.63**	**179.78**
BPIC'20	$I(\emptyset)$	3.68	3.66	3.87	3.62	3.85	3.90	3.72	3.66
	$I(\overline{\mathcal{C}}, \overline{w})$	**3.58**	**3.57**	**3.81**	**3.53**	**3.69**	**3.70**	**3.65**	**3.48**

Handled (PH)). The inter-case pattern(s) identified for segments *(DF, RP)* and *(RP, PH)* are $R_1 = non - batching$, $batch(s)$, and $R_2 = batch(e)$ respectively. To create inter-case features, we implement NS using [18] and follow steps described in Sect. 3.5 to create $TM_{S,R}$ models using LightGBM [6]. Predictions are made with different bucketing prefixes, encoding prefix events, and ML methods. We consider *prefix bucketing methods* to be grouping by prefix lengths (p), using a clustering algorithm (c) or grouping all prefixes in a single bucket (s). Common *prefix encoding methods* include data of only last prefix event (l) or aggregating data of all prefix events (a), and apply ML models, XGBoost (x) or random forest (r) to the input encoded feature vectors. The following input configurations are used: (1) $I(\emptyset)$: event log with no inter-case features, (2) $I(\mathcal{C}, \overline{w})$: event log with inter-case features created using actual segment prediction $\mathcal{C}$, and (3) $I(\overline{\mathcal{C}}, \overline{w})$: event log with inter-case features created using segment prediction made using NS. We use 80% and 20% (by temporally splitting) of the event logs for training and testing the RTMs. To measure overall prediction accuracy, we measure the weighted average MAE [17] of all predictions $\mathcal{P}$ made on test data.

4.2 Results

Table 5 shows that using inter-case features leads to an increase in performance for all 8 combinations of bucketing prefixes, encoding prefix events, and ML methods in RTMs against baseline $I(\emptyset)$. For the RF event log, we see that prediction error decreases by a maximum of 14.26% and a minimum of 4.27% for methods (p, l, x) and (c, a, x), respectively, with $I(\overline{\mathcal{C}}, \overline{w})$. For the BPIC'20 event log, we observe a maximum decrease of 5.12% and a minimum decrease of 1.55% in weighted average MAE for methods (c, a, r) and (c, a, x), respectively. Since BPIC'20 is a smaller event log with fewer cases subject to the identified inter-case patterns, the overall reduction in prediction error is smaller. The most accurate predictions for the

Table 6. MAE (in days) for different configurations (I) with the similar lengths bucketing (p), aggregating events data for encoding prefix events (a), and XGBoost (x) as the ML method, $RTM = (p, a, x)$. $\mathcal{P}_k$ is the set of all predictions for prefixes of length k.

RF	$I(\emptyset)$	$I(\mathcal{C}, \overline{w})$	$I(\overline{\mathcal{C}}, \overline{w})$	BPIC'20	$I(\emptyset)$	$I(\mathcal{C}, \overline{w})$	$I(\overline{\mathcal{C}}, \overline{w})$
$\mathcal{P}_{k=2}$	176.37	107.85	**106.74**	$\mathcal{P}_{k=3}$	4.03	**3.84**	4.06
$\mathcal{P}_{k=3}$	227.38	**189.22**	200.02	$\mathcal{P}_{k=4}$	2.64	**2.22**	2.23
$\mathcal{P}_{k=4}$	202.92	**123.19**	171.11	$\mathcal{P}_{k=5}$	1.07	0.98	**0.97**

RF event log obtained using $I(\overline{\mathcal{C}}, \overline{w})$ with (c, l, x), has a MAE 0.6 days less than the benchmark result [17]. However, our approach's privilege is that these predictions can be interpreted more easily because of the inter-case features.

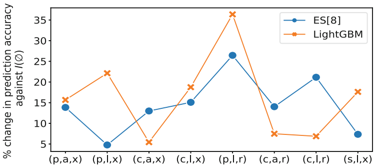


Fig. 6. Comparing prediction results for RF

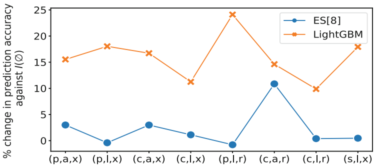


Fig. 7. Comparing prediction results for BPIC'20

In our approach, inter-case features are primarily included for prefixes passing through uncertain segments which occur at some step k of the process. Therefore, we look at MAE of predictions made for all prefixes of relevant length k, i.e., $\mathcal{P}_k \subseteq \mathcal{P}$. Segments $(SF,\ IF)$, $(IF,\ AP)$ and $(AP,\ SC)$ of the RF event log occur predominantly at step $k = 2$, $k = 3$ and $k = 4$ of the process respectively. Segments $(DF,\ RP)$ and $(RP,\ PH)$ of the BPIC'20 log occur predominantly at steps $k = 3$ and $k = 4,\ 5$ respectively. Table 6 shows us the results for predictions made using $RTM = (p, a, x)$. For the RF event log, the prediction error decreases by 39%, 12% and 15% for $\mathcal{P}_2$, $\mathcal{P}_3$ and $\mathcal{P}_4$, respectively using $I(\overline{\mathcal{C}}, \overline{w})$ over baseline. For BPIC'20, error decreases up to 15% and 9% for $\mathcal{P}_4$ and $\mathcal{P}_5$, respectively, when using $I(\overline{\mathcal{C}}, \overline{w})$. However, the MAE of $\mathcal{P}_3$ is slightly higher for configuration $I(\overline{\mathcal{C}}, \overline{w})$ compared to $I(\emptyset)$. This is because of incorrect segment predictions for *(DF, RP)* made by NS which is proven by the results of $I(\mathcal{C}, \overline{w})$. Figures 6 and 7 compare the *batching-at-end* aware predictions made using inter-case features created in our approach that uses LightGBM [6]) and previous approach [8] that uses exponential smoothing (ES). We measure the increase/decrease in performance of $\mathcal{P}_4$ made using different combination of RTMs over their respective baselines. We compare only predictions at $k = 4$ for both logs where uncertain segments with *batching-at-end* dynamics occur. Figure 6 shows that, our approach performs better than previous approach in 5 of the 8 input configuration (I) for batched cases in RF event log. Figure 7 shows that for the batched cases in BPIC'20 log, our method performs better for all the configurations.

5 Conclusion

We presented an approach to systematically discover a subset of uncertain process segments with inter-case dynamics that cause high prediction errors. Contrary to previous approaches, our designed function for detecting the subset of uncertain segments, limited the manual intervention to the identification of inter-case patterns within these segments. Using visual analysis, we identified and gained insights about inter-case pattern(s) within uncertain segments. In particular, we gained insights into non-batching (FIFO and unordered), *batching-at-start*, and *batching-at-end* inter-case patterns. Subsequently, we included these insights in remaining time predictions by transforming them into the inter-case features. For instance, there is a maximum increase in overall prediction performance by 14.2% for RF event-log. Since there is no standardized process to create a ML model for inter-case feature creation, our proposed approach is also sensitive to user interpretation. Yet, it provides more interpretability to RTMs. Note that despite an overall decrease in prediction error, some prefixes were heavily over-predicted or under-predicted. Therefore, the next step is to improve the prediction models and leverage routing probability derived from stochastic process models. It improves the inter-case feature creation for segment prediction. Another possible path is to make RTM aware of non-case-related aspects, e.g., resources dependencies.

References

1. van der Aalst, W.M.P., Schonenberg, M., Song, M.: Time prediction based on process mining. Inf. Syst. **36**(2), 450–475 (2011)
2. Denisov, V., Fahland, D., van der Aalst, W.M.P.: Unbiased, fine-grained description of processes performance from event data. In: Business Process Management (2018)
3. Denisov, V., Fahland, D., van der Aalst, W.M.P.: Predictive performance monitoring of material handling systems using the performance spectrum. In: International Conference on Process Mining (ICPM), pp. 137–144. IEEE (2019)
4. van Dongen, B.F.: BPI Challenge 2020: domestic declarations dataset (2020)
5. Galanti, R., Coma-Puig, B., de Leoni, M., Carmona, J., Navarin, N.: Explainable predictive process monitoring. In: (ICPM), pp. 1–8. IEEE (2020)
6. Ke, G., et al.: LightGBM: a highly efficient gradient boosting decision tree, In: NIPS, pp. 3149–3157 (2017)
7. Klijn, E., Fahland, D.: Performance mining for batch processing using the performance spectrum. In: Business Process Management Workshops (2019)
8. Klijn, E.L., Fahland, D.: Identifying and reducing errors in remaining time prediction due to inter-case dynamics. In: (ICPM), pp. 25–32. IEEE (2020)
9. de Leoni, M.M., Mannhardt, F.: Road traffic fine management process (2015)
10. Navarin, N., Vincenzi, B., Polato, M., Sperduti, A.: LSTM networks for data-aware remaining time prediction of business process instances. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1–7 (2017)
11. Polato, M., Sperduti, A., Burattin, A., Leoni, M.: Time and activity sequence prediction of business process instances. Computing **100**(9), 1005–1031 (2018). https://doi.org/10.1007/s00607-018-0593-x
12. Pourbafrani, M., van der Aalst, W.M.P.: Extracting process features from event logs to learn coarse-grained simulation models. In: CAiSE 2021, pp. 125–140 (2021). https://doi.org/10.1007/978-3-030-79382-1_8
13. Pourbafrani, M., Jiao, S., van der Aalst, W.M.P.: SIMPT: process improvement using interactive simulation of time-aware process trees. In: Cherfi, S., Perini, A., Nurcan, S. (eds.) RCIS 2021. LNBIP, vol. 415, pp. 588–594. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-75018-3_40

14. Senderovich, A., Di Francescomarino, C., Ghidini, C., Jorbina, K., Maggi, F.M.: Intra and inter-case features in predictive process monitoring: a tale of two dimensions. In: Carmona, J., Engels, G., Kumar, A. (eds.) BPM 2017. LNCS, vol. 10445, pp. 306–323. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-65000-5_18
15. Senderovich, A., Weidlich, M., Gal, A., Mandelbaum, A.: Queue mining for delay prediction in multi-class service processes. Inf. Syst. **53**, 278–295 (2015)
16. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive business process monitoring with LSTM. Neural Netw. **10253**, 477–492 (2017)
17. Verenich, I., Dumas, M., Rosa, M.L., Maggi, F.M., Teinemaa, I.: Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring. ACM **10**(4), 1–34 (2019)
18. Wang, J., Yu, D., Liu, C., Sun, X.: Outcome-oriented predictive process monitoring with attention-based bidirectional LSTM neural networks, In: ICWS. pp. 360–367 (2019)

Event Log Sampling for Predictive Monitoring

Mohammadreza Fani Sani[1(✉)], Mozhgan Vazifehdoostirani[2], Gyunam Park[1],
Marco Pegoraro[1], Sebastiaan J. van Zelst[1,3], and Wil M. P. van der Aalst[1,3]

[1] Process and Data Science Chair, RWTH Aachen University, Aachen, Germany
{fanisani,gnpark,pegoraro,s.j.v.zelst,
wvdaalst}@pads.rwth-aachen.de

[2] Industrial Engineering and Innovation Science, Eindhoven University of Technology,
Eindhoven, The Netherlands
m.vazifehdoostirani@tue.nl

[3] Fraunhofer FIT, Birlinghoven Castle, Sankt Augustin, Germany

Abstract. Predictive process monitoring is a subfield of process mining that aims to estimate case or event features for running process instances. Such predictions are of significant interest to the process stakeholders. However, state-of-the-art methods for predictive monitoring require the training of complex machine learning models, which is often inefficient. This paper proposes an instance selection procedure that allows sampling training process instances for prediction models. We show that our sampling method allows for a significant increase of training speed for next activity prediction methods while maintaining reliable levels of prediction accuracy.

Keywords: Process Mining · Predictive Monitoring · Sampling · Machine Learning · Deep Learning · Instance Selection

1 Introduction

As the environment surrounding business processes becomes more dynamic and competitive, it becomes imperative to predict process behaviors and take proactive actions [1]. Predictive business process monitoring aims at predicting the behavior of business processes, to mitigate the risk resulting from undesired behaviors in the process. For instance, by predicting the next activities in the process, one can foresee the undesired execution of activities, thus preventing possible risks resulting from it [12]. Moreover, by predicting an expected high service time for an activity, one may bypass or add more resources for the activity [15]. Recent breakthroughs in machine learning have enabled the development of effective techniques for predictive business process monitoring. Specifically, techniques based on deep neural networks, e.g., Long-Short Term Memory (LSTM) networks, have shown high performance in different tasks [8]. Additionally, the emergence of ensemble learning methods leads to improvement in accuracy in different areas [4]. Particularly, for predictive process monitoring, eXtreme Gradient Boosting (XGBoost) [6] has shown promising results, often outperforming other ensemble methods such as Random Forest or using a single regression tree [25,28].

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 154–166, 2022.
https://doi.org/10.1007/978-3-030-98581-3_12

Indeed, machine learning algorithms suffer from the expensive computational costs in their training process [34]. In particular, machine learning algorithms based on neural networks and ensemble learning might require tuning their hyperparameters to be able to provide acceptable accuracy. Such long training time limits the application of the techniques considering the limitations in time and hardware [21]. This is particularly relevant for predictive business process monitoring techniques. Business analysts need to test the efficiency and reliability of their conclusions via repeated training of different prediction models with different parameters [15]. Moreover, the dynamic nature of business processes requires new models adapting to new situations in short intervals.

Instance selection aims at reducing original datasets to a manageable volume to perform machine learning tasks, while the quality of the results (e.g., accuracy) is maintained as if the original dataset was used [11]. Instance selection techniques are categorized into two classes based on the way they select instances. First, some techniques select the instances at the boundaries of classes. For instance, Decremental Reduction Optimization Procedure (DROP) [32] selects instances using *k*-Nearest Neighbors by incrementally discarding an instance if its neighbors are correctly classified without the instance. The other techniques preserve the instances residing inside classes, e.g., Edited Nearest Neighbor (ENN) [33] preserves instances by repeatedly discarding an instance if it does not belong to the class of the majority of its neighbors.

Such techniques assume independence among instances [32]. However, in predictive business process monitoring training, instances may be highly correlated [2], impeding the application of techniques for instance selection. Such instances are computed from event data that are recorded by the information system supporting business processes [14]. The event data are correlated by the notion of *case*, e.g., patients in a hospital or products in a factory. In this regard, we need new techniques for instance selection applicable to event data.

In this work, we suggest an instance selection approach for predicting the next activity, one of the main applications of predictive business process monitoring. By considering the characteristics of the event data, the proposed approach samples event data such that the training speed is improved while the accuracy of the resulting prediction model is maintained. We have evaluated the proposed methods using two real-life datasets and state-of-the-art techniques for predictive business process monitoring, including LSTM [13] and XGBoost [6].

The remainder is organized as follows. We discuss the related work in Sect. 2. Next, we present the preliminaries in Sect. 3 and proposed methods in Sect. 4. Afterward, Sect. 5 evaluates the proposed methods using real-life event data and Sect. 6 provides discussions. Finally, Sect. 7 concludes the paper.

2 Related Work

Predictive process monitoring is an exceedingly active field of research. At its core, the fundamental component of predictive monitoring is the abstraction technique it uses to obtain a fixed-length representation of the process component subject to the prediction (often, but not always, process traces). In the earlier approaches, the need for such abstraction was overcome through model-aware techniques, employing process models and replay techniques on partial traces to abstract a flat representation of event sequences. Such process models are mostly automatically discovered from a set of available complete traces, and require perfect fitness on training instances (and, seldomly, also on unseen test instances). For instance, van der Aalst et al. [1] proposed a time prediction framework based on replaying partial traces on a transition system, effectively clustering training instances by control-flow information. This framework has later been the basis for a prediction method by Polato et al. [20], where the transition system is annotated with an ensemble of SVR and Naïve Bayes classifiers, to perform a more accurate time estimation. A related approach, albeit more linked to the simulation domain and based on a Monte Carlo method, is the one proposed by Rogge-Solti and Weske [24], which maps partial process instances in an enriched Petri net.

Recently, predictive process monitoring started to use a plethora of machine learning approaches, achieving varying degrees of success. For instance, Teinemaa et al. [27] provided a framework to combine text mining methods with Random Forest and Logistic Regression. Senderovich et al. [25] studied the effect of using intra-case and inter-case features in predictive process monitoring and showed a promising result for XGBoost compared to other ensemble and linear methods. A comprehensive benchmark on using classical machine learning approaches for outcome-oriented predictive process monitoring tasks [28] has shown that the XGBoost is the best-performing classifier among different machine learning approaches such as SVM, Decision Tree, Random Forest, and logistic regression.

More recent methods are model-unaware and perform based on a single and more complex machine learning model instead of an ensemble. The LSTM network model has proven to be particularly effective for predictive monitoring [8,26], since the recurrent architecture can natively support sequences of data of arbitrary length. It allows performing trace prediction while employing a fixed-length event abstraction, which can be based on control-flow alone [8,26], data-aware [16], time-aware [17], text-aware [19], or model-aware [18].

A concept similar to the idea proposed in this paper, and of current interest in the field of machine learning, is *dataset distillation*: utilizing a dataset to obtain a smaller set of training instances that contain the same information (with respect to training a machine learning model) [31]. While this is not considered sampling, since some instances of the distilled dataset are created ex-novo, it is an approach very similar to the one we illustrate in our paper. Moreover, recently some instance selection algorithms have been proposed to help process mining algorithms. For example, [9,10] proposed to use instance selection techniques to improve the performance of process discovery and conformance checking procedures.

In this paper, we examine the underexplored topic of event data sampling and selection for predictive process monitoring, with the objective of assessing if and to which extent prediction quality can be retained when we utilize subsets of the training data.

3 Preliminaries

In this section, some process mining concepts such as event log and sampling are discussed. In process mining, we use events to provide insights into the execution of business processes. Each *event* is related to specific activities of the underlying process. Furthermore, we refer to a collection of events related to a specific process instance as a *case*. Both cases and events may have different attributes. An event log that is a collection of events and cases is defined as follows.

Definition 1 (Event Log). *Let $\mathcal{E}$ be the universe of events, $\mathcal{C}$ be the universe of cases, $\mathcal{AT}$ be the universe of attributes, and $\mathcal{U}$ be the universe of attribute values. Moreover, let $C\subseteq\mathcal{C}$ be a non-empty set of cases, let $E\subseteq\mathcal{E}$ be a non-empty set of events, and let $AT\subseteq\mathcal{AT}$ be a set of attributes. We define (C, E, π_C, π_E) as an event log, where $\pi_C{:}C\times\mathcal{AT}\nrightarrow\mathcal{U}$ and $\pi_E{:}E\times\mathcal{AT}\nrightarrow\mathcal{U}$. Any event in the event log has a case, therefore, $\nexists_{e\in E}(\pi_E(e, case)\notin C)$ and $\bigcup\limits_{e\in E}(\pi_E(e, case))=C$.*

Furthermore, let $\mathcal{A}\subseteq\mathcal{U}$ be the universe of activities and let $\mathcal{V}\subseteq\mathcal{A}^*$ be the universe of sequences of activities. For any $e\in E$, function $\pi_E(e, activity)\in\mathcal{A}$, which means that any event in the event log has an activity. Moreover, for any $c\in C$ function $\pi_C(c, variant)\in\mathcal{A}^*\backslash\{\langle\rangle\}$ that means any case in the event log has a variant.

Therefore, there are some mandatory attributes that are *case* and *activity* for events and *variants* for cases. In some process mining applications, e.g., process discovery and conformance checking, just variant information is considered. Therefore, event logs are considered as a multiset of sequences of activities. In the following, a simple event log is defined.

Definition 2 (Simple event log). *Let $\mathcal{A}$ be the universe of activities and let the universe of multisets over a set X be denoted by $\mathcal{B}(X)$. A simple event log is $L\in\mathcal{B}(\mathcal{A}^*)$. Moreover, let $\mathcal{EL}$ be the universe of event logs and $EL=(C, E, \pi_C, \pi_E)\in\mathcal{EL}$ be an event log. We define function $sl{:}\mathcal{EL}\rightarrow\mathcal{B}(\{\pi_E(e, activity)|e\in E\}^*)$ returns the simple event log of an event log. The set of unique variants in the event log is denoted by $\overline{sl(EL)}$.*

Therefore, sl returns the multiset of variants in the event logs. Note that the size of a simple event log equals the number of cases in the event logs, i.e., $sl(EL)=|C|$

In this paper, we use sampling techniques to reduce the size of event logs. An event log sampling method is defined as follows.

Definition 3 (Event log sampling). *Let $\mathcal{EL}$ be the universe of event logs and $\mathcal{A}$ be the universe of activities. Moreover, let $EL=(C, E, \pi_C, \pi_E)\in\mathcal{EL}$ be an event log, we define function $\delta{:}\mathcal{EL}\rightarrow\mathcal{EL}$ that returns the sampled event log where if $(C', E', \pi'_C, \pi'_E)=\delta(EL)$, then $C'\subseteq C$, $E'\subseteq E$, $\pi'_e\subseteq\pi_E$, $\pi'_C\subseteq\pi_C$, and consequently, $\overline{sl(\delta(EL))}\subseteq\overline{sl(EL)}$. We define that δ is a variant-preserving sampling if $\overline{sl(\delta(EL))}=\overline{sl(EL)}$.*

In other words, a sampling method is variant-preserving if and only if all the variants of the original event log are presented in the sampled event log.

To use machine learning methods for prediction, we usually need to transfer each case to one or more features. The feature is defined as follows.

Definition 4 (Feature). *Let $\mathcal{AT}$ be the universe of attributes, $\mathcal{U}$ be the universe of attribute values, and $\mathcal{C}$ be the universe of cases. Moreover, let $AT\subseteq\mathcal{AT}$ be a set of attributes. A feature is a relation between a sequence of attributes' values for AT and the target attribute value, i.e., $f\in(\mathcal{U}^{|AT|}\times\mathcal{U})$. We define $fe{:}\mathcal{C}\times\mathcal{EL}\rightarrow\mathcal{B}(\mathcal{U}^{|AT|}\times\mathcal{U})$ is a function that receives a case and an event log, and returns a multiset of features.*

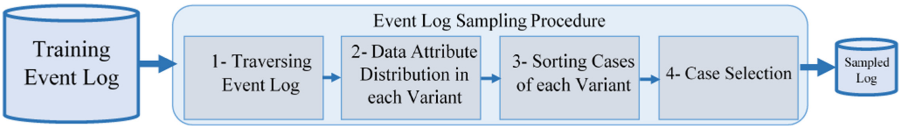

Training Event Log
Event Log Sampling Procedure
1- Traversing Event Log
2- Data Attribute Distribution in each Variant
3- Sorting Cases of each Variant
4- Case Selection
Sampled Log

Fig. 1. A schematic view of the proposed sampling procedure

For the next activity prediction, i.e., our prediction goal, the target attribute value should be an activity. Moreover, a case in the event log may have different features. For example, suppose that we only consider the activities. For the case $\langle a, b, c, d\rangle$, we may have $(\langle a\rangle, b)$, $(\langle a, b\rangle, c)$, and $(\langle a, b, c\rangle, d)$ as features. Furthermore, $\sum_{c\in C} fe(c, EL)$ are the corresponding features of event log $EL = (C, E, \pi_C, \pi_E)$ that could be given to different machine learning algorithms. For more details on how to extract features from event logs please refer to [23].

4 Proposed Sampling Methods

In this section, we propose an event log preprocessing procedure that helps prediction algorithms to perform faster while maintaining reasonable accuracy. The schematic view of the proposed sampling approach is presented in Fig. 1. We first need to traverse the event log and find the variants and corresponding traces of each variant in the event log. Moreover, different distributions of data attributes in each variant will be computed. Afterward, using different sorting and instance selection strategies, we are able to select some of the cases and return the sample event log. In the following, each of these steps is explained in more detail.

1. *Traversing the event log*: In this step, the unique variants of the event log and the corresponding traces of each variant are determined. In other words, consider event log EL that $\overline{sl(EL)} = \{\sigma_1, ..., \sigma_n\}$ where $n = |\overline{sl(EL)}|$, we aim to split EL to $EL_1, .., EL_n$ where EL_i only contains all the cases that $C_i = \{c\in C | \pi_C(c, variant) = \sigma_i\}$ and $E_i = \{e\in E | \pi_E(e, case)\in C_i\}$. Obviously, $\bigcup_{1\leq i\leq n} (C_i) = C$ and $\bigcap_{1\leq i\leq n} (C_i) = \varnothing$.
2. *Distribution Computation*: In this step, for each variant of the event log, we compute the distribution of different data attributes $a\in AT$. It would be more practical if the interesting attributes are chosen by an expert. Both event and case attributes can be considered. A simple approach is to compute the frequency of categorical data values. For numerical data attributes, it is possible to consider the average or the median of values for all cases of each variant.
3. *Sorting the cases of each variant*: In this step, we aim to sort the traces of each variant. We need to sort the traces to give a higher priority to those traces that can represent the variant better. One way is to sort the traces based on the frequency of the existence of the most occurred data values of the variant. For example, we can give a higher priority to the traces that have more frequent resources of each variant. It is also possible to sort the traces based on their arrival time or randomly.

4. *Returning sample event logs*: Finally, depending on the setting of the sampling function, we return some of the traces with the highest priority for all variants. The most important point about this step is to know how many traces of each variant should be selected. In the following, some possibilities will be introduced.
   - *Unique selection*: In this approach, we select only one trace with the highest priority. In other words, suppose that $L' = sl(\delta(EL)), \forall_{\sigma \in L'} L'(\sigma) = 1$. Therefore, using this approach we will have $|sl(\delta(EL))| = |\overline{sl(EL)}|$. It is expected that using this approach, the distribution of frequency of variants will be changed and consequently the resulted prediction model will be less accurate.
   - *Logarithmic distribution*: In this approach, we reduce the number of traces in each variant in a logarithmic way. If $L = sl(EL)$ and $L' = sl(\delta(EL))$, $\forall_{\sigma \in L'} L'(\sigma) = [Log_k(L(\sigma))]$. Using this approach, the infrequent variants will not have any trace in the sampled event log. By using a higher k, the size of the sampled event log is reduced more.
   - *Division*: This approach performs similar to the previous one, however, instead of using logarithmic scale, we apply the division operator. In this approach, $\forall_{\sigma \in L'} L'(\sigma) = \lceil \frac{(\sigma)}{k} \rceil$. A higher k results in fewer cases in the sample event log. Note that using this approach all the variants have at least one trace in the sampled event log.

   There is also a possibility to consider other selection methods. For example, we can select the traces completely randomly from the original event log.

By choosing different data attributes in Step 2 and different sorting algorithms in Step 3, we are able to lead the sampling of the method on *which* cases should be chosen. Moreover, by choosing the type of distribution in Step 4, we determine *how many* cases should be chosen. To compute how sampling method δ reduces the size of the given event log EL, we use the following equation:

$$R_S = \frac{|sl(EL)|}{|sl(\delta(EL))|} \tag{1}$$

The higher R_S value means, the sampling method reduces more the size of the training log. By choosing different distribution methods and different *k-values*, we are able to control the size of the sampled event log. It should be noted that the proposed method will apply just to the training event log. In other words, we do not sample event logs for development and test datasets.

5 Evaluation

In this section, we aim at designing some experiments to answer our research question, i.e., "Can we improve the computational performance of prediction methods by using the sampled event logs, while maintaining a similar accuracy?". It should be noted that the focus of the experiments is not on prediction model tuning to have higher accuracy. Conversely, we aim to analyze the effect of using sampled event logs (instead of the whole datasets) on the required time and the accuracy of prediction models. In the following, we first explain the event logs that are used in the experiments. Afterward, we provide some information about the implementation of sampling methods. Moreover, the experimental setting is discussed and, finally, we show the experimental results.

Table 1. Overview of the event logs that are used in the experiments. The accuracy and the required times (in seconds) of different prediction methods for these event logs are also presented.

Event Log	Cases	Activities	Variants	Attributes	FE Time	*LSTM* Train Time	*LSTM Acc*	*XG* Train Time	*XG Acc*
RTFM	150370	11	231	1	73649	3021	0.791	11372	0.814
BPIC-2012-W	9658	6	2643	2	1212	3344	0.68	2011	0.685

5.1 Event Logs

To evaluate the proposed sampling procedure for prediction, we have used two event logs widely used in the literature. Some information about these event logs is presented in Table 1. In the *RTFM* event log, which corresponds to a road traffic management system, we have some high frequent variants and several infrequent variants. Moreover, the number of activities in this event log is high. Some of these activities are infrequent, which makes this event log imbalanced. In the *BPIC-2012-W* event log, relating to a process of an insurance company, the average of variant frequencies is lower.

5.2 Implementation

We have developed the sampling methods as a plug-in in the ProM framework [30], accessible via https://svn.win.tue.nl/repos/prom/Packages/LogFiltering. This plug-in takes an event log and returns k different train and test event logs in the CSV format. Moreover, to train the prediction method, we have used XGBoost [6] and LSTM [13] methods as they are widely used in the literature and outperformed their counterparts. Our LSTM network consisted of an input layer, two LSTM layers with *dropout* rates of 10%, and a dense output layer with the *SoftMax* activation function. We used "categorical cross-entropy" to calculate the loss and adopted *ADAM* as an optimizer. We used *gbtree* with a max depth of 6 as a booster in our XGBoost model. Uniform distribution is used as the sampling method inside our XGBoost model. To avoid overfitting in both models, the training set is further divided into 90% training set and 10% validation set to stop training once the model performance on the validation set stops improving. We used the same setting of both models for original event logs and sampled event logs. To access our implementations of these methods and the feature generation please refer to https://github.com/gyunamister/pm-prediction/. For details of the feature generation and feature encoding steps, please refer to [18].

5.3 Evaluation Setting

To sample the event logs, we use three distributions that are log *distribution*, *division*, and *unique variants*. For the log distribution method, we have used $2, 3$, and 10 (i.e., log_2, log_3, and log_{10}). For the division method, we have used $2, 5$, and 10 (i.e., $d2, d5$, and $d10$). For each event log and for each sampling method, we have used a 5-$fold$ cross-validation. Moreover, as the results of the experiments are non-deterministic, all the experiments have been repeated 5 times and the average values are represented.

Note that, for both training and evaluation phases, we have used the same settings for extracting features and training prediction models. We used one-hot encoding to encode the sequence of activities for both LSTM and XGBoost models. We ran the experiment on a server with Intel Xeon CPU E7-4850 2.30 GHz, and 512 GB of RAM. In all the steps, one CPU thread has been used. We employed the *Weighted Accuracy* metric [22] to compute how a prediction method performs for test data. To compare the accuracy of the prediction methods, we use the *relative accuracy* that is defined as follows.

$$R_{Acc} = \frac{\text{Accuracy using the sampled training log}}{\text{Accuracy using the whole training log}} \tag{2}$$

If R_{Acc} is close to 1, it means that using the sampling event logs, the prediction methods behave almost similar to the case that the whole data is used for the training. Moreover, values higher than 1 indicate the accuracy of prediction methods has improved.

To compute the improvement in the performance of training time, we will use the following equations.

$$R_t = \frac{\text{Training time using whole data}}{\text{Training time using the sampled data}} \tag{3}$$

$$R_{FE} = \frac{\text{Feature extraction time using whole data}}{\text{Feature extraction time using the sampled data}} \tag{4}$$

For both equations, the resulting values indicate how many times the sampled log is faster than using all data.

5.4 Experimental Results

Table 2 presents the reduction rate and the improvement in the feature extraction phase using different sampling methods. As it is expected, the highest reduction rate is for log_{10} (as it removes infrequent variants and keeps few traces of frequent variants), and respectively it has the biggest improvement in R_{FE}. Moreover, the lowest reduction is for *d2*, especially if there are lots of unique variants in the event log (i.e., for the *RTFM* event log). We expected smaller event logs to require less feature extraction time. However, results indicate that the relationship is not linear, and by having more reduction in the size of the sampled event log there will be a much higher reduction in the feature extraction time.

In Table 3 and Table 4, the results of improvement in R_t and R_{Acc} are shown for LSTM and XG prediction methods. As expected, by using fewer cases in the training, the performance of training time improvement will be higher. Comparing the results in these two tables and the results in Table 2, it is interesting to see that in some cases, even by having a high reduction rate, the accuracy of the trained prediction model is close to the case in which whole training log is used. For example, using $d10$ for the *RTFM* event log, we will have high accuracy for both prediction methods. In other words, we are able to improve the performance of the prediction procedure while the accuracy is still reasonable.

When using the LSTM prediction method for the RTFM event log, there are some cases where we have accuracy improvement. For example, using $d3$, there is a 0.4% improvement in the accuracy of the trained model. It is mainly because of the existence of high frequent variants. These variants lead to having unbiased training logs and consequently, the accuracy of the trained model will be lower for infrequent behaviors.

Table 2. The reduction in the size of training logs (i.e., R_S) and the improvement in the performance of feature extraction part (i.e., R_{FE}) using different sampling methods.

Sampling Methods	d2		d3		d10		log_2		log_3		log_{10}		unique	
Event Log	R_S	R_{FE}	R_S	R_{FE}	R_S	R_{FE}	R_S	R_{FE}	R_S	R_{FE}	R_S	R_{FE}	R_S	R_{FE}
RTFM [7]	1.99	4.8	3.0	11.1	9.8	106.9	153.5	12527.6	236.3	23699.2	**572.3**	**74912.8**	285.1	24841.8
BPIC-2012-W [29]	1.22	1.37	1.41	1.80	1.66	2.51	6.06	22.41	9.05	37.67	28.50	208.32	1.73	2.36

Table 3. The accuracy and the improvement in the performance of prediction using different sampling methods for LSTM.

Sampling Methods	d2		d3		d10		log_2		log_3		log_{10}		unique	
Event Log	R_{Acc}	R_t	R_{Acc}	R_t	R_{Acc}	R_t	R_{Acc}	R_t	R_{Acc}	R_t	R_{Acc}	R_t	R_{Acc}	R_t
RTFM	1.001	2.0	**1.004**	2.9	0.990	9.0	0.716	26.7	0.724	33.0	0.767	41.8	0.631	29.1
BPIC-2012-W	1.000	1.4	0.985	1.3	0.938	1.3	0.977	4.7	0.970	5.8	0.876	11.9	0.996	1.6

Table 4. The accuracy and the improvement in the performance of prediction using different sampling methods for XGBoost.

Sampling Methods	d2		d3		d10		log_2		log_3		log_{10}		unique	
Event Log	R_{Acc}	R_t	R_{Acc}	R_t	R_{Acc}	R_t	R_{Acc}	R_t	R_{Acc}	R_t	R_{Acc}	R_t	R_{Acc}	R_t
RTFM	**1.000**	2.4	**1.000**	1.4	**1.000**	**84.1**	0.686	126.4	0.706	191.8	0.772	355.0	0.582	297.7
BPIC-2012-W	0.999	2.3	0.998	2.4	0.997	3.4	0.923	10.7	0.970	16.7	0.883	64.8	0.997	2.8

6 Discussion

The results indicate that we do not always have a typical trade-off between the accuracy of the trained model and the performance of the prediction procedure. In other words, there are some cases where the training process is much faster than the normal procedure, even though the trained model provides an almost similar accuracy. We did not provide the results for other metrics; however, there are similar patterns for weighted recall, precision, and f1-score. Thus, the proposed sampling methods can be used when we aim to apply hyperparameter optimization [3]. In this way, more settings can be analyzed in a limited time. Moreover, it is reasonable to use the proposed method when we aim to train an online prediction method or on naive hardware such as cell phones.

Another important outcome of the results is that for different event logs, we should use different sampling methods to achieve the highest performance. For example, for the *RTFM* event log—as there are some highly frequent variants—the division distribution may be more useful. In other words, independently of the used prediction method, if we change the distribution of variants (e.g., using $unique$ distribution), it is expected that the accuracy will sharply decrease. However, for event logs with a more uniform distribution, we can use logarithmic and unique distributions to sample event logs. The results indicate that the effect of the chosen distribution (i.e., $unique$, $division$, and $logarithmic$) is more important than the used *k-value*. Therefore, it would be valuable to investigate more on the characteristics of the given event log and suitable sampling parameters for such distribution. For example, if most variants of a given event log are unique, the $division$ and $unique$ methods are not able to have remarkable R_S and consequently, R_{FE} and R_t will be close to 1.

Moreover, results have shown that by oversampling the event logs, although we will have a very big improvement in the performance of the prediction procedure, the accuracy of the trained model is significantly lower than the accuracy of the model that is trained by the whole event log. Therefore, we suggest gradually increasing (or decreasing) the size of the sampled event log in the hyper-parameter optimization scenarios.

By analysis of the results using common prediction methods, we have found that the infrequent activities can be ignored using some hyper-parameter settings. This is mainly because the event logs are unbalanced for these infrequent activities. Using the sampling methods that modify the distribution of the event logs such as the $unique$ method can help the prediction methods to also consider these activities.

Finally, in real scenarios, the process can change because of different reasons [5]. This phenomenon is usually called *concept drift*. By considering the whole event log for training the prediction model, it is most probable that these changes are not considered in the prediction. Using the proposed sampling procedure, and giving higher priorities to newer traces, we are able to adapt to the changes faster, which may be critical for specific applications.

7 Conclusion

In this paper, we proposed to use the subset of event logs to train prediction models. We proposed different sampling methods for next activity prediction. These methods are implemented in the ProM framework. To evaluate the proposed methods, we have applied them on two real event logs and have used two state-of-the-art prediction methods: LSTM and XGBoost. The experimental results have shown that, using the proposed method, we are able to improve the performance of the next activity prediction procedure while retaining an acceptable accuracy (in some experiments, the accuracy increased). However, there is a relation between event logs characteristics and suitable parameters that can be used to sample these event logs. The proposed methods can be helpful in situations where we aim to train the model fastly or in hyper-parameter optimization scenarios. Moreover, in cases where the process can change over time, we are able to adapt to the modified process more quickly using sampling methods.

To continue this research, we aim to extend the experiments to study the relationship between the event log characteristics and the sampling parameters. Additionally, we plan to provide some sampling methods that help prediction methods to predict infrequent activities, which could be more critical in the process. Finally, it is interesting to investigate more on using sampling methods for other prediction method applications such as last activity and remaining time prediction.

Acknowledgment. The authors would like to thank the Alexander von Humboldt (AvH) Stiftung for funding this research.

References

1. van der Aalst, W.M.P., Schonenberg, M., Song, M.: Time prediction based on process mining. Inf. Syst. **36**(2), 450–475 (2011). https://doi.org/10.1016/j.is.2010.09.001
2. van der Aalst, W.M.P.: Process Mining - Data Science in Action, 2nd edn. Springer, Verlag (2016)
3. Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. In: Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, Proceedings of a Meeting Held 12–14 December 2011, Granada, Spain. pp. 2546–2554 (2011)
4. Breiman, L.: Bagging predictors. Mach. Learn. **24**(2), 123–140 (1996)
5. Carmona, J., Gavaldà, R.: Online techniques for dealing with concept drift in process mining. In: Hollmén, J., Klawonn, F., Tucker, A. (eds.) IDA 2012. LNCS, vol. 7619, pp. 90–102. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34156-4_10
6. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13–17, 2016, pp. 785–794. ACM (2016)
7. De Leoni, M., Mannhardt, F.: Road traffic fine management process. Eindhoven University of Technology, Dataset (2015)
8. Evermann, J., Rehse, J., Fettke, P.: Predicting process behaviour using deep learning. Decis. Support Syst. **100**, 129–140 (2017)
9. Fani Sani, M., van Zelst, S.J., van der Aalst, W.M.P.: Conformance checking approximation using subset selection and edit distance. In: Dustdar, S., Yu, E., Salinesi, C., Rieu, D., Pant, V. (eds.) CAiSE 2020. LNCS, vol. 12127, pp. 234–251. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49435-3_15
10. Fani Sani, M., van Zelst, S.J., van der Aalst, W.M.P.: The impact of biased sampling of event logs on the performance of process discovery. Computing **103**(6), 1085–1104 (2021). https://doi.org/10.1007/s00607-021-00910-4
11. Garca, S., Luengo, J., Herrera, F.: Data Preprocessing in Data Mining. Springer, Cham (2014)
12. Breuker, D., Matzner, M., Delfmann, P., Becker, J.: Comprehensible predictive models for business processes. Mis Q. **40**(4), 1009–1034. https://doi.org/10.25300/MISQ/2016/40.4.10
13. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. CoRR arXiv:1508.01991 (2015)
14. de Leoni, M., van der Aalst, W.M.P., Dees, M.: A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs. Inf. Syst. **56**, 235–257 (2016). https://doi.org/10.1016/j.is.2015.07.003
15. Marquez-Chamorro, A.E., Resinas, M., Ruiz-Cortes, A.: Predictive monitoring of business processes: a survey. IEEE Trans. Services Comput. **11**(6), 962–977 (2017). https://doi.org/10.1109/TSC.2017.2772256
16. Navarin, N., Vincenzi, B., Polato, M., Sperduti, A.: LSTM networks for data-aware remaining time prediction of business process instances. In: 2017 IEEE Symposium Series on Computational Intelligence, SSCI 2017, Honolulu, HI, USA, November 27 - December 1, 2017, pp. 1–7. IEEE (2017)
17. Nguyen, A., Chatterjee, S., Weinzierl, S., Schwinn, L., Matzner, M., Eskofier, B.: Time matters: time-aware LSTMs for predictive business process monitoring. In: Leemans, S., Leopold, H. (eds.) ICPM 2020. LNBIP, vol. 406, pp. 112–123. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-72693-5_9
18. Park, G., Song, M.: Predicting performances in business processes using deep neural networks. Decis. Support Syst. **129**, 113191 (2020)

19. Pegoraro, M., Uysal, M.S., Georgi, D.B., van der Aalst, W.M.P.: Text-aware predictive monitoring of business processes. In: Abramowicz, W., Auer, S., Lewanska, E. (eds.) 24th International Conference on Business Information Systems, BIS 2021, Hannover, Germany, June 15–17, 2021. pp. 221–232 (2021)
20. Polato, M., Sperduti, A., Burattin, A., Leoni, M.: Time and activity sequence prediction of business process instances. Computing **100**(9), 1005–1031 (2018). https://doi.org/10.1007/s00607-018-0593-x
21. Pourghassemi, B., Zhang, C., Lee, J.H., Chandramowlishwaran, A.: On the limits of parallelizing convolutional neural networks on GPUS. In: SPAA 2020: 32nd ACM Symposium on Parallelism in Algorithms and Architectures, Virtual Event, USA, July 15–17, 2020. pp. 567–569. ACM (2020)
22. Powers, D.M.W.: Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. CoRR arXiv:2010.16061 (2020)
23. Qafari, M.S., van der Aalst, W.: Root cause analysis in process mining using structural equation models. In: Del Río Ortega, A., Leopold, H., Santoro, F.M. (eds.) BPM 2020. LNBIP, vol. 397, pp. 155–167. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-66498-5_12
24. Rogge-Solti, A., Weske, M.: Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays. In: Basu, S., Pautasso, C., Zhang, L., Fu, X. (eds.) ICSOC 2013. LNCS, vol. 8274, pp. 389–403. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-45005-1_27
25. Senderovich, A., Di Francescomarino, C., Ghidini, C., Jorbina, K., Maggi, F.M.: Intra and inter-case features in predictive process monitoring: a tale of two dimensions. In: Carmona, J., Engels, G., Kumar, A. (eds.) BPM 2017. LNCS, vol. 10445, pp. 306–323. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-65000-5_18
26. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive business process monitoring with LSTM neural networks. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 477–492. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_30
27. Teinemaa, I., Dumas, M., Maggi, F.M., Di Francescomarino, C.: Predictive business process monitoring with structured and unstructured data. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM 2016. LNCS, vol. 9850, pp. 401–417. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45348-4_23
28. Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M.: Outcome-oriented predictive process monitoring: Review and benchmark. ACM Trans. Knowl. Discovery Data (TKDD) **13**(2), 1–57 (2019)
29. Van Dongen, B.F. (Boudewijn): BPI Challenge 2012 (2012). https://doi.org/10.4121/UUID:3926DB30-F712-4394-AEBC-75976070E91F
30. Verbeek, E., Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: Prom 6: the process mining toolkit. In: Proceedings of the Business Process Management 2010 Demonstration Track, Hoboken, NJ, USA, September 14–16, 2010. vol. 615. CEUR-WS.org (2010)
31. Wang, T., Zhu, J.Y., Torralba, A., Efros, A.A.: Dataset distillation. arXiv preprint arXiv:1811.10959 (2020)
32. Wilson, D.R., Martinez, T.R.: Reduction techniques for instance-basedlearning algorithms. Mach. Learn. **38**(3), 257–286 (2000). https://doi.org/10.1023/A:1007626913721
33. Wilson, D.L.: Asymptotic properties of nearest neighbor rules using edited data. Syst., Man Cyber., IEEE Trans. **2**(3), 408–421 (1972). https://doi.org/10.1109/TSMC.1972.4309137
34. Zhou, L., Pan, S., Wang, J., Vasilakos, A.V.: Machine learning on big data: opportunities and challenges. Neurocomputing **237**, 350–361 (2017). https://doi.org/10.1016/j.neucom.2017.01.026

Active Anomaly Detection for Key Item Selection in Process Auditing

Ruben Post[1(✉)], Iris Beerepoot[1], Xixi Lu[1], Stijn Kas[1], Sebastiaan Wiewel[1], Angelique Koopman[2], and Hajo Reijers[1]

[1] Information and Computer Sciences, Utrecht University, Utrecht, The Netherlands
r.m.post@students.uu.nl

[2] Department of Accountancy, Tilburg University, Tilburg, The Netherlands

Abstract. Process mining allows auditors to retrieve crucial information about transactions by analysing the process data of a client. We propose an approach that supports the identification of unusual or unexpected transactions, also referred to as exceptions. These exceptions can be selected by auditors as "key items", meaning the auditors wants to look further into the underlying documentation of the transaction. The approach encodes the traces, assigns an anomaly score to each trace, and uses the domain knowledge of auditors to update the assigned anomaly scores through active anomaly detection. The approach is evaluated with three groups of auditors over three cycles. The results of the evaluation indicate that the approach has the potential to support the decision-making process of auditors. Although auditors still need to make a manual selection of key items, they are able to better substantiate this selection. As such, our research can be seen as a step forward with respect to the usage of anomaly detection and data analysis in process auditing.

Keywords: Process Mining · Domain Knowledge · Anomaly Detection · Auditing

1 Introduction

In the past years, it has become clear that data captured by information systems are relevant for auditors [1,2]. Process mining allows auditors to elicit behaviour from process data in the form of event logs derived from information systems of their clients [1, p. 32]. An event log is a collection of cases, where a case is a sequence of events performed in the context of a single process [1, p. 128]. As an event log collects behaviour captured by the information systems involved, it can be considered as an unbiased perspective on the client's processes [3]. Take, for example, an event log that contains loan offers made by a bank. The bank receives a customer request for a loan, asks for additional information until it has sufficient information, and finally decides to grant the loan or not. Without process data, the auditor does not know the steps taken before the loan was granted (i.e. the behaviour), while this behaviour could be instrumental in the auditor's decision to further investigate a particular loan offer.

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 167–179, 2022.
https://doi.org/10.1007/978-3-030-98581-3_13

Choosing which loan offer, or any other transaction of the client, to investigate further is also referred to as key item selection. Key items are specific transactions that auditors want to look further into by, for example, requesting additional documentation on the transactions because they might have a higher likelihood of containing a material misstatement (i.e. transactions that violate accounting and auditing standards [4, p. 374]). Currently, key items are selected based on the size of the transactions (i.e. the transactions with the highest monetary value), professional judgement, or by drawing a sample [5, Ch. 6]. Without detailed information about transactions, unusual or unexpected aspects such as the number of times a transaction has been declined and resubmitted, the total number of activities performed in the transaction, or the throughput time of the transaction, are largely ignored while selecting key items.

Anomaly detection algorithms can be used to detect exceptions, which can then be selected as key items. However, both supervised and unsupervised approaches may be unsuitable for practice because they either require a large amount of labeled training data or lack explainability [6–11]. More specifically, selecting the key items based on the underlying process data still requires domain knowledge [11]. Hence, active engagement of domain experts (i.e. auditors) is needed to detect exceptions. This leads to the following research question: *How can key item selection be supported using active anomaly detection on process data?* The research question is answered by structuring the identification of exceptions in process data in a three-step approach. In doing so, we contribute to the research field of process mining and auditing. By embedding domain knowledge in the identification of exceptions, the approach shows that the involvement of domain experts can be beneficial for both the domain experts' insight into the process of the client and the results of the approach itself. Additionally, because the approach provides a more detailed account of the transactions, the selection is better substantiated.

2 Related Work

2.1 Anomaly Detection

Anomaly detection approaches can be used to identify unusual or unexpected transactions in process data, also referred to as *exceptions*. Several anomaly detection approaches suitable for process data have been proposed [6–11]. Current approaches mostly use trained models to detect anomalies, like Ko et al. [10] and Pauwels et al. [9]. Using these approaches in practice is difficult because there is no labeled data available during audits, meaning these models cannot be trained. An alternative to training data could be a temporal holdout set where the data of the prior audit is used to train the model (i.e. the prior-year data is used to train the model to identify anomalies in the current-year data). However, if a temporal holdout set is used, concepts such as concept drift should be taken into account because the model might not know the difference between an anomaly and the introduction of a new process. An example of this is the recent COVID-19 pandemic, which coerced the digitisation of processes, introducing concept drift to the process data.

Other approaches such as those of Nolle et al. [8] and Böhmer et al. [7] use neural network-based autoencoders and Basic Likelihood Graphs to identify anomalies. This introduces complexity through the techniques they use, leading to both an increase in required processing power and unexplainability or incomprehensibility of the approach. This is problematic because it prohibits the domain expert to adequately substantiate their selection of key items.

In contrast to the other approaches, Schumann et al. [11] use low-complexity models that do not require training data to identify anomalies. Because they determine certain non-compliant patterns in the data beforehand and inject the data with these patterns, no training data is required. While the other approaches are evaluated through various performance metrics, only Schumann et al. [11] evaluate their technique with domain experts. The benefit of this type of evaluation is that is allows for domain experts to differentiate between real anomalies and cases that are considered compliant in practice. However, the rationale used by the domain experts is not explained, which brings the applicability and replicability of the approach in practice in question.

2.2 Active Anomaly Detection

Traditional anomaly detection approaches do not actively engage domain experts when identifying exceptions while the performance of anomaly detection approaches can potentially be improved by incorporating domain expert feedback. An example of a framework that allows using domain expert feedback is the Active Anomaly Detection (AAD) framework by Das et al. [12]. The AAD framework takes an ensemble model and assigns an initial weight to each individual model. The weight of a model influences how much it contributes to the anomaly score of a data point. A higher weight gives a model more influence on the anomaly score. After assigning an anomaly score to each case, a query budget B is defined and the instances with the top-B anomaly scores are labeled by domain experts. After each instance is labeled, the weights of the models are updated. The technical details of how the weights are updated are left out due to size limitations but can be found in [12].

To the best of our knowledge, anomaly detection approaches that actively engage domain experts are not currently used in practice. Furthermore, as mentioned above, selecting key items on the underlying process data still requires domain knowledge. AAD could provide domain experts the opportunity to embed their knowledge in the anomaly detection algorithm. However, because in this research process data is used, some additional steps need to be taken before the data is suitable the AAD framework. The reason the AAD framework is chosen is because it is written in a programming language compatible with current process mining techniques (i.e. Python).

2.3 Trace Visualisation

By using the AAD framework, domain knowledge is embedded in the assigning of the anomaly scores. It could however be that domain experts have different opinions on the label of a case. Hence, the information presented to domain experts should make clear **why** the presented case has a high anomaly score in an understandable and interpretable way.

Within process mining literature, different types of visualisations have been proposed, each of them serving different purposes [13]. According to Klinkmüller et al. [14], when assessing the conformance of a case in the BPIC2012 event log, around 41.3% of the information needs can be fulfilled with tables presenting case and/or event attributes and 34.8% by a process model. The remainder is often fulfilled with a line or bar chart, fulfilling 20.6% of the information needs. By taking into account the information needs of domain experts when visualising the trace, they are supported in their decision-making process and can better substantiate their key item selection.

3 Active Selection Approach

Taking into account the literature discussed above, we propose the Active Selection Approach. The goal of the approach is to structure the selection of key items using process data available during an audit. Figure 1 gives an overview of the steps that make up the approach. The remainder of this section describes each step in more detail.

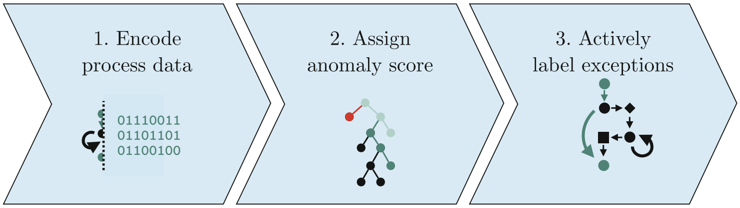


Fig. 1. Active Selection Approach

3.1 Step One: Encode Process Data

Before anomalies can be detected using traditional methods, the event log needs to be transformed into a tabular data structure where data is structured into rows, each of which contains information about a case, also known as *trace encoding* [15,16]. The way the traces are encoded should be tailored towards the process of the client and the type of information that should be retained. For example, if the domain experts are only interested in the resources and monetary value of the transactions, there is no need to consider the temporal aspect of the process data during encoding. Should some of the resulting features consistently have the same value as another feature or holds a constant value throughout the event log, they can be removed. Preferably, no further feature selection should be done, in order to retain as much information about the event log as possible.

3.2 Step Two: Assign Anomaly Score

After the process data is encoded, an anomaly detection algorithm is used to assign an anomaly score to each case. The only constraint to the choice of algorithm is that it has to be an ensemble method so that the weight of the individual models can be updated based on domain expert feedback.

3.3 Step Three: Actively Label Exceptions

With each case having an anomaly score, the cases with the highest anomaly score can be visualised and shown to a domain expert. Based on the visualisations, the domain expert has to label the case as either a key item or not. Based on the label, the weights of the algorithm are updated. This step has two benefits: 1) the domain experts gain insight into anomalous traces within the process of the client and 2) the weights of the algorithm are updated, potentially improving the results. After the domain expert has labeled a set number of cases, the algorithm assigns an updated anomaly score to each case. The result of the approach is an enriched event log containing updated anomaly scores. Based on these, the domain expert can decide to select certain cases with a high anomaly score as key items, thereby supporting their decision-making process.

4 Evaluation

The approach was implemented in Python (available on Github[1]) and evaluated over three cycles with several domain experts: senior auditors from an audit firm, experienced students from a post-master accountancy program (around 2–4 years of practical experience), and attendees of a symposium on statistical auditing. Each cycle had two objectives: (1) evaluate the performance of the approach and (2) measure the saturation of the information needs of the domain experts with regards to the trace visualisation. During each cycle, domain experts completed a survey[2] that showed them six cases. Three of those cases were considered an exception (i.e. had a high anomaly score) and the other three were not (i.e. had a low anomaly score). The label given by the domain experts was viewed as the true label to later compute performance metrics. Table 1 provides an overview of who participated in each cycle and which sub-process they were shown.

4.1 Step One: Encode Process Data

The process data used during the evaluation is the publicly available Business Process Intelligence Challenge 2012 (BPIC2012) event log [17]. The event log contains 13.087 cases with 262.200 events. It describes an application process for a personal loan within a bank. The event log is chosen because the loan applications contain financial information and could therefore realistically be

[1] https://github.com/rubenpost/Model_agnostic_AAD/blob/main/main.py.

[2] https://survey.uu.nl/jfe/form/SV_5jUGtcjPq1muHgG.

Table 1. Evaluation cycles

Cycle	Participants	Sub-process	# cases labeled
One	3 senior auditors and 15 students	The offer (O)	108
Two	3 senior auditors and 53 students	The offer (O)	336
Three	3 senior auditors, 18 students, and 108 symposium attendees	The offer (O) and the work items (W)	648

part of an audit. There are 24 unique activities in the event log, representing three sub-processes. The event log contains three sub-processes, the application (A), the offer (O), and the work items (W) belonging to the application. To reduce the learning curve for domain experts when interpreting the information about the process, only one sub-process was used during each evaluation. This also reduced the number of activities the domain experts had to review. During the first and second cycle, the offer (O) was shown. In the third cycle, the offer (O) was shown to the auditors and students, while the work items (W) were shown to the professionals at the Limperg Symposium Statistical Auditing. Only accepted loan applications were included, as we assume that only these cases would have a financial impact on the client. The final event log contained 2.243 cases and 15.701 events.

The event log contained several attributes. In Table 2, the trace encoding used during this evaluation is described. All attributes were encoded as either aggregates or static. This means the order of the activities is lost, but the frequency is still kept as part of the feature. After trace encoding, the data had a shape of 2243×71, meaning there are 2.243 cases each represented by 71 features. Because of the limited moments available with the domain experts, the encoding type per attribute was not optimised based on the evaluation results.

Table 2. Feature encoding on BPIC2012 event log

Attribute	Category	Type	Encoding
CaseID	Case	Static	Not included
Resource	Event	Dynamic	Frequency
Activity	Event	Dynamic	Frequency
Timestamp	Event	Dynamic	Frequency
Registration	Event	Dynamic	Frequency
Status	Event	Dynamic	Frequency
Amount	Case	Static	As-is
Activity count	Case	Static	As-is
Case length	Case	Static	As-is

4.2 Step Two: Assign Anomaly Score

During the evaluation, the Isolation Forest algorithm was used to assign an anomaly score to each case [18]. The Isolation Forest algorithm was used because it can cope with high-dimensional data sets, is generally fast, and is an ensemble method. The default parameters for the Isolation Forest as described in Das et al. [12] are used. Because of the way the approach is evaluated, the algorithm is not instantiated within the Active Anomaly Detection framework until the third cycle. For the first and second cycle, each case was assigned an anomaly score by Isolation Trees without individual weights. After assigning the anomaly score, the top and bottom 100 anomaly scores (viewed as exceptions and no exceptions, respectively) were used in the survey during cycle one and two. The labels received during the first and second cycle were used to update the weights of the algorithm for the third cycle.

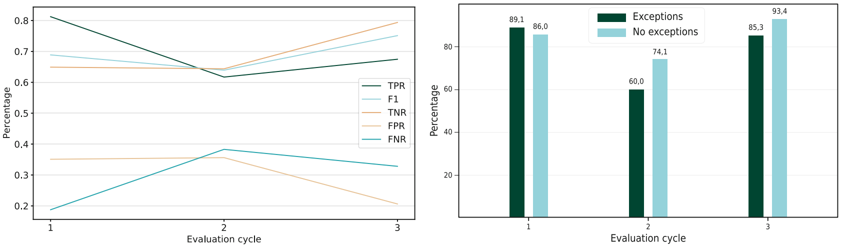
4.3 Step Three: Actively Label Exceptions

The trace visualisation for the evaluation consisted of four different visuals. The first visual is a directly-follows-graph process model generated with PM4Py, a Python-based process mining package [19]. This type of graph is solely based on which activity directly follows which activity (i.e. a directly-follows dependency $(a > b)$) [20]. This means that concurrency and parallelism are ignored. This type of process model was chosen due to its ability to show the many loops a process can take [21]. In addition to the activities, the process model also includes the time between activities and time spent on the activity. One table visualises the directly-follows dependency between all activities in the case. The reason the directly-follows dependency was included in a separate table is that the frequency of the dependency is shown in the table, but not in the process model. Another table shows which resource performed which activities and how many times. Lastly, all numeric features of the case were plotted in a histogram. The bin in which the value of the case resides is highlighted.

4.4 Performance Results

With the labels collected during the survey, the performance metrics were computed to evaluate the performance progression after each evaluation cycle. In addition to the performance metrics, the label confidence per label is computed, which shows how often domain experts agreed on the label of a case. The metrics are visualised in Fig. 2 (a). In this figure, the progression of the metric after each evaluation in a cycle is shown. The last result of the evaluation is the label confidence per evaluation cycle. In Fig. 2 (b), the label confidence is shown both for cases identified as exceptions and as not an exception by the approach.

After labeling the cases, the domain experts were asked two questions: *"What additional information would help in making your decision?"* and *"Did you have enough information to make your decision about each case?"*. These open questions relate to the second objective of the evaluation (i.e. are the information



(a) Performance metrics per cycle

(b) Label confidence per cycle

Fig. 2. Survey results

needs met?). The first question was an open question that aimed to provide feedback and points of interest for the approach. In addition to this open question, comments made by the domain experts were also written down and used as input for the next cycle. The answers to the second question also indicate the saturation of the domain experts' information needs and has three levels: *"yes"*, *"somewhat"*, and *"no"*.

The results of the first open question are shown in Table 3. The table describes which information needs were identified during each evaluation and how these were implemented in the next cycle (i.e. their impact) and Fig. 3 shows the indication of saturation per cycle.

Table 3. Suggested information needs and impact on approach

Cycle	**Suggested information needs**	**Impact**
One	Relation between number of resources used in the case and the norm	Included the average number of cancellations a case in the event log contains and how often a case with the same or more cancellations is found
	Relation between number of times a case is cancelled and the norm	Included the average number of resources a case in the event log contains and how often a case is performed by the same or more resources
	The directly follows relationship table is hard to interpret at first	Included an explanation of the table in the figure title
Two	Internal procedures for cancelling a case	None, this information is not available
Three	The financial impact of certain activities on the organisation	None, this information is not available
	Background of the process and the resources that work on it	None, this information is not available

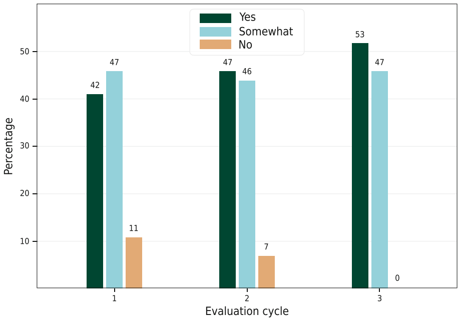


Fig. 3. Progression of information needs saturation per cycle

5 Discussion

In the next paragraphs, the results of the questions on information needs are discussed, the impact on the approach is described, and the performance metrics are interpreted per cycle.

5.1 Cycle One

During the first cycle, the domain experts seemed to agree that more context was needed on the case they were reviewing. Specifically, the attribute values of the case needed to be put in the context of the entire event log. Hence, the averages of several attributes deemed important by the domain expert are added to the trace visualisation. Besides this, they noted that the directly follows relationship table was hard to interpret. However, once they understood how to read the table, the information seemed very useful (mostly because it showed the order of the activities, something the model did not always do clearly).

The performance metrics show that the exceptions identified through the approach were quite often labeled as such by the domain experts. However, cases not identified as an exception were often labeled as an exception by domain experts as well. Hence, the FPR is slightly higher than the FNR. The F1 score was 68.9%, indicating that if the approach was used in a real audit, and the cases with the highest anomaly scores were selected as key items, it would select a key item most of the time. The label confidence was high in the first cycle (averaging 87.4%), meaning domain experts generally agreed on the label of a case. Hence, the robustness of the performance metrics of the first cycle is considered high. The performance metrics did not lead to further changes to the approach.

Based on these results, more context on the case was added to the trace visualisation. By adding the average over the entire event log and the rarity (i.e. how many other cases have the same values for that specific feature), the domain expert can compare the case to the ‘norm’.

5.2 Cycle Two

During cycle two, the information needs of the domain experts seemed to be more saturated. This can be seen both in the number of suggested information needs (only one) and the percentage of domain experts indicating they had enough information to review each case (93%). Except for the need for a more elaborate explanation of the histograms in the trace visualisation, the suggested information needs had no impact on the approach. This is because the information was not available during this research.

Despite the fact that the information saturation was higher in the second cycle, the performance metrics mostly decreased. The F1 score decreased to 63.4% (an 8% decline) and cases were more often predicted incorrectly. With the exception of the FNR, which increased with by 20%. This means domain experts were more likely to identify cases as an exception in general. The label confidence was also much lower, averaging 67.1%. The high information saturation and low label confidence indicate that professional judgement had a large influence on the performance metrics. This was not observed in the first cycle, but is further confirmed by domain experts indicating that working with process data is new and needs adjusting to, meaning that they have to rely more on their professional judgement than might be intended during an audit.

Based on these results, one minor change was made to the approach. An explanation was added to the histogram to describe what exactly the domain experts were looking at and the information the histogram gave them.

5.3 Cycle Three

During the last cycle, no further information needs were identified that had impact on the approach. Additionally, all of the domain experts indicated that they either had enough or somewhat enough information to review each case. None of the domain experts indicated they did not have enough information, indicating that the information needs were saturated the most in the third cycle.

The labels collected through the survey in the first and second cycle were used to update the weights of the algorithm. This led to an increase in performance metrics: the F1 score was the highest of all the cycles with 72.2%. A similar increase was also seen in the other performance metrics, meaning that domain experts were more likely to label a case the same way the approach did. The labeling confidence was also the highest out of all the cycles, averaging 89.4%. This cycle also evaluated two sub-processes. The increase in information saturation and performance metrics indicate that the approach generalised well to different sub-processes of the event log used during the evaluation. Because no further information needs were identified, no changes were made to the approach. This is in line with the measured information saturation.

6 Limitations

The study has potential limitations. The first limitation is the bias introduced by the domain experts when labeling the cases. This shows through the labeling confidence; the average labeling confidence is between 67.1% and 89.4%. This shows that labeling a case could involve a substantial amount of professional judgement. This could be related to the experience of the domain experts that participated in the survey, which was not always known. Besides the three auditors, the background and expertise of the students and professionals are unknown. This could lead to lower quality labels.

The approach requires domain experts to label cases before receiving a final list of identified exceptions. This might be cumbersome and could cause friction during the usage in audit. Regarding the results, it is unknown whether the results can be generalised to different event logs. Because only one event log, albeit divided into two sub-processes, is used during the evaluation, the results might not be reproducible with different event logs. The same is true for the encoding of the traces. Because different encoding types were not evaluated, it is unknown whether different trace encoding would improve the results.

7 Conclusion and Future Work

The evaluation showed that the approach has the potential to support the decision-making process of domain experts when selecting key items. Although auditors still need to make a manual selection of key items, they are able to better substantiate this selection. During the evaluation, multiple signs indicated that professional judgement had a large influence on the label domain experts gave a case (and therefore on the results). There were two reasons why professional judgement was still required. First of all, there was more uncertainty among the decision-makers with respect to the context of the process execution. During a normal audit, more contextual information is available. This could cause the domain experts to select more key items than they would normally do to reduce the risk of missing a misstatement. The second reason is that working with process data is new and needs adjusting to, meaning that domain experts relied more on their professional judgement then they normally would when selecting key items with less information about the behaviour of the transaction.

The subjectivity involved in labeling the cases should be further reduced. Currently, trace visualisation attempts to standardise the information on which the domain experts make their decision. By standardising this information, the decision of the domain experts becomes more structured and standardised, reducing the subjectivity involved during their decision-making. Future work should standardise the trace visualisation, further structuring the way professional judgement is used throughout the approach.

References

1. Van Der Aalst, W.M.P.: Process Mining: Data Science in Action, 2nd edn. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
2. Jans, M., Alles, M., Vasarhelyi, M.: The case for process mining in auditing: Sources of value added and areas of application. Int. J. Account. Inf. Syst. **14**(1), 1–20 (2013)

3. Jans, M., Alles, M., Vasarhelyi, M.: Process mining of event logs in auditing: Opportunities and challenges. SSRN Electron. J. 1–32 (2010). https://ssrn.com/abstract=1578912
4. International Standard on Auditing 450. Evaluation of misstatement identified during the audit (2009). https://www.ifac.org/system/files/downloads/a021-2010-iaasb-handbook-isa-450.pdf
5. Boynton, W.C., Kell, W.G., Johnson, R.N., Wheeler, S.W.: Modern Auditing, 8th edn. J. Wiley & Sons, Hoboken (2001)
6. Sureka, A.: Kernel based sequential data anomaly detection in business process event logs. arXiv preprint arXiv:1507.01168 (2015)
7. Böhmer, K., Rinderle-Ma, S.: Multi-perspective anomaly detection in business process execution events. In: Debruyne, C., et al. (eds.) OTM 2016. LNCS, vol. 10033, pp. 80–98. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48472-3_5
8. Nolle, T., Luettgen, S., Seeliger, A., Mühlhäuser, M.: Analyzing business process anomalies using autoencoders. Mach. Learn. **107**(11), 1875–1893 (2018). https://doi.org/10.1007/s10994-018-5702-8
9. Pauwels, S., Calders, T.: An anomaly detection technique for business processes based on extended dynamic bayesian networks. In: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, pp. 494–501 (2019)
10. Ko, J., Comuzzi, M.: Detecting anomalies in business process event logs using statistical leverage. Inf. Sci. **549**, 53–67 (2021)
11. Schumann, G., Kruse, F., Nonnenmacher, J.: A practice-oriented, control-flow-based anomaly detection approach for internal process audits. In: Kafeza, E., Benatallah, B., Martinelli, F., Hacid, H., Bouguettaya, A., Motahari, H. (eds.) ICSOC 2020. LNCS, vol. 12571, pp. 533–543. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-65310-1_39
12. Das, S., Wong, W.K., Dietterich, T., Fern, A., Emmott, A.: Incorporating expert feedback into active anomaly discovery. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), IEEE, pp. 853–858 (2016)
13. van der Aalst, W.M.P., de Leoni, M., ter Hofstede, A.H.: Process mining and visual analytics: Breathing life into business process models. BPM Center Report BPM-11-15, BPMcenter. org **17**, 699–730 (2011)
14. Klinkmüller, C., Müller, R., Weber, I.: Mining process mining practices: an exploratory characterization of information needs in process analytics. In: Hildebrandt, T., van Dongen, B.F., Röglinger, M., Mendling, J. (eds.) BPM 2019. LNCS, vol. 11675, pp. 322–337. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26619-6_21
15. Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M.: Outcome-oriented predictive process monitoring: Review and benchmark. ACM Trans. Knowl. Discovery Data (TKDD) **13**(2), 1–57 (2019)
16. De Leoni, M., van der Aalst, W.M.P., Dees, M.: A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs. Inf. Syst. **56**, 235–257 (2016)
17. van Dongen, B.F.: "BPI challenge 2012" (2012). https://data.4tu.nl/articles/dataset/BPI_Challenge_2012/12689204/1
18. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: 2008 eighth IEEE international conference on data mining. IEEE, pp. 413–422 (2008)
19. Berti, A., Van Zelst, S.J. and van der Aalst, W.: Process mining for python (pm4py): bridging the gap between process-and data science. International Conference on Process Mining (2019)

20. Augusto, A., Conforti, R., Dumas, M., La Rosa, M.: Split miner: discovering accurate and simple business process models from event logs. In: 2017 IEEE International Conference on Data Mining (ICDM), IEEE, pp. 1–10 (2017)
21. van der Aalst, W.M.P.: A practitioner's guide to process mining: limitations of the directly-follows graph (2019)

Prescriptive Process Monitoring Under Resource Constraints: A Causal Inference Approach

Mahmoud Shoush and Marlon Dumas(✉)

University of Tartu, Tartu, Estonia
{mahmoud.shoush,marlon.dumas}@ut.ee

Abstract. Prescriptive process monitoring is a family of techniques to optimize the performance of a business process by triggering interventions at runtime. Existing prescriptive process monitoring techniques assume that the number of interventions that may be triggered is unbounded. In practice, though, interventions consume resources with finite capacity. For example, in a loan origination process, an intervention may consist of preparing an alternative loan offer to increase the applicant's chances of taking a loan. This intervention requires time from a credit officer. Thus, it is not possible to trigger this intervention in all cases. This paper proposes a prescriptive monitoring technique that triggers interventions to optimize a cost function under fixed resource constraints. The technique relies on predictive modeling to identify cases that are likely to lead to a negative outcome, in combination with causal inference to estimate the effect of an intervention on a case's outcome. These estimates are used to allocate resources to interventions to maximize a cost function. A preliminary evaluation suggests that the approach produces a higher net gain than a purely predictive (non-causal) baseline.

1 Introduction

Prescriptive Process Monitoring (PrPM) [5,8] is a set of techniques to recommend or to trigger actions (herein called *interventions*) during the execution of a process in order to optimize its performance. PrPM techniques use business process execution logs (a.k.a. *event logs*) to predict negative outcomes that affect the performance of the process and use these predictions to determine if and when to trigger interventions to prevent or mitigate such negative outcomes.

Several PrPM techniques have been proposed [2,5,8]. These techniques, however, assume that it is possible to trigger any number of interventions at any point in time. In practice, each intervention requires some resources (e.g., time from an employee), and those resources have a limited capacity. For example, in a loan origination process, an intervention could be to provide an alternative

Supported by the European Research Council (PIX Project).

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 180–193, 2022.
https://doi.org/10.1007/978-3-030-98581-3_14

loan offer to increase the applicant's likelihood of taking a loan. This intervention can only be triggered if a loan officer is available to perform it.

In this setting, we address the question of whether or not to trigger an intervention during the execution of an instance of a business process (herein a *case*) to optimize a gain function that considers the cost of the case ending in a negative outcome and the cost of the intervention. We tackle this question in the context where each intervention requires locking a resource for given *treatment duration* and where the number of available resources is bounded.

To address this question, we use a predictive model to estimate the probability of a negative case outcome and a causal inference approach to estimate the effect of triggering an intervention on the probability of a negative case outcome. Based on these estimates, we estimate the gain of triggering an intervention for each case. We use this estimate to decide which cases should be treated given the available resources. We report on evaluating a real-life event log to compare the proposed approach with a baseline that relies only on predictive models.

The paper is structured as follows. Section 2 presents background concepts and related work. Section 3 explains the approach while Sect. 4 discusses the empirical evaluation. Finally, Sect. 5 summarizes this paper and future work directions.

2 Background and Related Work

2.1 Predictive Process Monitoring

PrPM techniques are closely related to techniques for estimating the probability of negative case outcomes, also known as outcome-oriented Predictive Process Monitoring (PPM) techniques [12]. The input of an outcome-oriented PPM technique is an *event log* representing the execution of a business process. An extract of a loan handling process is shown in Fig. 1. This log consists of two traces. Each trace consists of a sequence of *events*. An event describes the execution of one activity instance. An event contains three attributes: a case identifier (c_{id}), an activity label (*activity*), and a *timestamp*. Other event attributes may exist, like who does the activity (the *resource*). Additional attributes may be of one of two types: *case attributes* or *event attributes*. Case attributes are attributes whose values do not change within a case, while the value of an event attribute changes. For example, in Fig. 1, the log contains two case attributes (*age* and *gender*) and one event attribute (*resource*).

$$\begin{aligned} trace_1 = {} & [(1, submitAnApplication, 12:00PM, (resource, emp_1), (age, 25), (gender, male), \\ & ..., (1, callClients, 02:00PM, (resource, emp_2), (age, 25), (gender, male))] \\ trace_2 = {} & [(2, makeAnOffer, 10:00AM, (resource, emp_3), (age, 30), (gender, female)), ..., \\ & (2, verifyDocuments, 02:00PM, , (resource, emp_4), (age, 30), (gender, female))] \end{aligned}$$

Fig. 1. Extract of a loan application process.

Outcome-oriented PPM methods predict the outcome of an ongoing case, given its (incomplete) trace. In a typical binary PPM method, the outcome of a case may be positive (e.g., a client accepted the loan offer) or negative (the client did not accept the offer). Accordingly, a precondition for applying a PPM method is to notion case outcomes and historical data about case outcomes. In the above example, this means that for each trace, we need to know whether or not the customer accepted the loan offer. An event log in which each trace is labeled with a case outcome is called a *labeled event log.*

PPM methods typically distinguish between an offline training phase and an online prediction phase. Based on historical (completed) cases, a predictive model (specifically a classification model) is trained in the offline phase. This model is then used during the online phase to make predictions based on incomplete traces. A typical approach to train models for PPM is to extract all or a subset of the prefixes with length k of the labeled trace in an event log and associate the full trace's label to every prefix extracted from the trace. A dataset of this form is called a *labeled prefix log.* A labeled prefix log is a set of prefixes of traces, each one with an associated case outcome (positive or negative).

$$\begin{aligned} vector_1 = [&((age, 25), (gender\_male, 1), (gender\_female, 0)),\\ &((res\_emp_1, 1), (res\_emp_2, 0), (res\_emp_3, 0), (res\_emp_4, 0)),\\ &((A\_submit\_an\_application, 1), ((A\_communicate\_clients, 0),\\ &((A\_make\_an\_offer, 0), ((A\_verify\_documents, 0)), (sum\_time, 0)] \end{aligned}$$

Fig. 2. Aggregate encoding for $trace_1$ with $k = 1$.

We use the labeled prefix log to train a machine learning algorithm to build a predictive monitoring model. However, we need first to encode the prefixes in the prefix log of each trace as so-called *feature vectors* (herein called *trace encoders*). Teinemaa et al. [11] propose and evaluate several types of trace encoders and find that *aggregation encoder* consistently yields models with high accuracy.

An aggregate encoder is a function that maps each prefix of a trace to a feature vector. Simply, it encodes each case attribute as a feature (or one-hot encode categorical case attributes). For each numerical event attribute, use an aggregation method (e.g., sum) over the sequence of values taken by this attribute in the prefix. For every categorical event attribute, encode every possible value of that information as numerical features. This information refers to the number of times this value has appeared in the prefix. An example of applying aggregate encodings to $trace_1$ with $k = 1$ is shown in Fig. 2.

2.2 Prescriptive Process Monitoring

Various PrPM methods have been proposed in prior work. Fahrenkrog et al. [5] introduce an approach to generate single or multiple alarms when the probability of a case leading to an undesired outcome is above a threshold (e.g., 70%).

Each alarm triggers an intervention, which reduces the probability of a negative outcome. Their method optimizes the threshold empirically w.r.t a gain function.

Metzger et al. [8] use ensemble methods to compute predictions and reliability estimates to trigger interventions. They introduce policy-based reinforcement learning to find and learn when to trigger proactive process adaptation. This work targets the problem of learning when to trigger an intervention rather than the question of whether or not to trigger an intervention. Both the technique of Metzger et al. and that of Fahrenkrog et al. work under the assumption that the number of interventions that may be triggered at a given point in time is unbounded. In contrast, in this paper, we consider resource constraints.

Weinzerl et al. [13] propose a PrPM technique to recommend the next activity in each ongoing case of a process, to maximize a given performance measure. This previous study does not consider an explicit notion of intervention. Thus, it does not consider the cost of intervention nor the fact that an intervention may only be triggered if a resource is available to perform it.

2.3 Causal Inference

Causal Inference (CI) [14] is a collection of techniques to discover and quantify cause-effect relations from data. Causal inference techniques have been used in a broad range of domains, including process mining.

In [3], the authors introduce a technique to find guidance rules following Treatment → Outcome relation, which improves the business process by triggering an intervention when a condition folds. They generate rules at design time in the level of groups of cases that will be validated later by domain experts. More recently, in [2], they address another target problem: reducing the cycle time of a process using interventions to maximize a net gain function. Both works [3] and [2] consider the estimation of the treatment effect. However, they assume that interventions with a positive impact occur immediately and do not examine the finite capacity of resources.

Causal inference techniques are categorized into two main frameworks [7]: (1) *Structural Causal Models* (SCMs), which consist of a causal graph and structural equations [1]. SCM focuses mainly on estimating the causal effects through a causal graph which a domain expert manually constructs. (2) *Potential outcome frameworks* focus on learning the treatment effects for a given treatment-outcome set (T, Y). Our work utilizes the latter, which focuses on automatic estimation methods rather than manually constructed graphs.

We use potential outcome models to estimate the treatment effect hereafter called *conditional average treatment effect (CATE)* from observational data. In particular, we use an *orthogonal random forest (ORF)* algorithm that combines tree-based models [1] and double machine learning [4] in one generalized approach [9]. It estimates the $CATE$ on an outcome Y when applying a treatment T to a given case with features X.

ORF requires input to be in the form of $input = \{(T_i, Y_i, W_i, X_i)\}_{i=1}^{n}$ for n instances. For each instance i, T_i is described by a binary variable $T \in \{0, 1\}$, where $T = 1$ refers to treatment is applied to a case and $T = 0$ that it is not. Y_i

refers to the observed outcome. W_i describes potential confounding properties, and X_i is the information achieving heterogeneity.

3 Approach

The primary objective of our approach is to determine whether or not to treat a given case and when an intervention takes place to maximize the total gain. To learn whether or not to treat, we build predictive and prescriptive models in the *learning phase*. Then, the *resource allocator* selects when to treat.

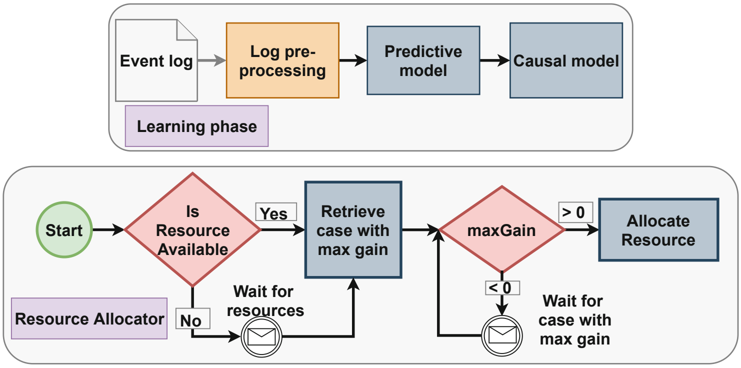


Fig. 3. Proposed approach

The approach consists of two phases, as shown in Fig. 3. In the learning phase, we prepare the event log to build two machine learning models. The first one is a model that estimates the undesired case outcome probability. The second one is the causal model to estimate the impact of a given intervention on the outcome of a case. The predicted probability of the negative outcome and the estimated treatment effect are used to determine the net gain in the resource allocation phase. Below, we explain each step in Fig. 3.

3.1 Log Preprocessing

Log preprocessing is an essential step in our approach that includes data cleaning, k-prefix extraction, prefix encoding, and identifying the outcome of cases and interventions that we might apply to reduce the probability of negative outcomes. For data cleaning, prefix extraction, and encoding, we follow the same approach proposed by Teinemaa et al. [12]. The setting of outcome and intervention is process-dependent which means we first need to understand the business process objective. Next, we analyze the log to find what interventions could affect the outcome of a given case by reducing the probability of negative outcomes.

3.2 Predictive Model

We build a predictive model to estimate the probability that cases will end with the undesired outcome. We use the estimated probabilities as a threshold τ that we optimize empirically to decide if we move forward to estimate the treatment effect and define gains or not.

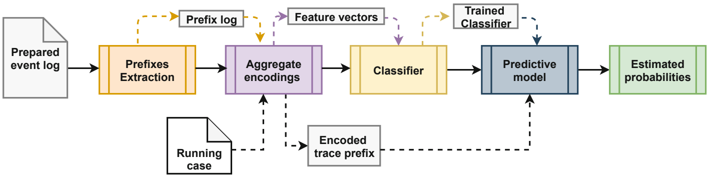


Fig. 4. Predictive model steps.

In order to build a predictive model, as shown in Fig. 4, first, we extract prefixes of length k from every trace that results in a so-called *prefix log*. This prefix extraction guarantees that our training log is similar to the testing log. For instance, If we have a complete trace containing seven events, we extract prefixes up to five events. Then we will have five incomplete traces starting with a trace containing only one event till a trace carrying five events. Next, in the aggregate encodings step, we encode each trace prefix into a fixed-size feature vector (see example in Fig. 2). Finally, we use the encoded log to train a machine learning method to estimate the probability of the undesired outcome.

This article deals with an outcome-oriented PPM problem, a classification problem from a machine learning perspective. The output from training a classification technique is a predictive model to estimate the probability of the undesired outcome (i.e., P_{uout}) of running cases.

3.3 Causal Model

We use ORF to build a causal model to estimate the treatment effects or the $CATE$ of an intervention in a given case. An advantage of using ORF w.r.t. other causal models is that it handles well high-dimensional feature spaces. This is useful in our setting because event logs have many event attributes with categorical values, leading to feature explosion.

To estimate CATE using ORF, the input needs to be in the form of $input = \{(T_i, Y_i, W_i, X_i)\}_{i=1}^{n}$ for n instances. For each instance i, T_i is the accepted treatment. Y_i refers to the observed outcome. T_i and Y_i come from the preprocessing step (see Sect. 3.1), and they might be differ from one process to another based on the process objective. W_i describes the potential confounding variables, and X_i is the information achieving heterogeneity. In this work, we deal with an outcome-oriented loan application process it means the purpose is to increase the rate of successful loan applications via treating ongoing applications. We hypothesized that the intervention increases the number of successful applications, and we assume that the treatment is identified beforehand. X and W are obtained from the encoded log. X is a feature vector that carries both case and event attributes, including the activity names, resources, and other extracted features,e.g., temporal attributes. We consider all these attributes to achieve the heterogeneity of the intervention effect. In this work, we assume that all log attributes X are too possible confounders W. Nevertheless, X and W may not be the same variables where a domain expert can specify which features would be removed from W if they do not improve the outcome.

Next, and based on the above descriptions, we train an ORF to estimate the treatment effect. The output from training an ORF technique is a causal model used to estimate $CATE$ for running cases.

3.4 Resource Allocator

We trained two models in the learning phase: the predictive one to estimate the probability that a case will end with the undesired outcome P_{uout} and the causal model to estimate the $CATE$ of utilizing an intervention in a given case. We use both models with the *resource allocator* to decide whether or not to treat a given case and when the intervention takes place to maximize the *total gain*.

Regularly triggering interventions in cases may come with gain; however, it comes at a cost. Therefore, to define the *total gain*, we determine the costs with and without intervention if the predictive model gives a probability higher than a specific threshold τ. Especially, suppose the intervention cost is relatively expensive as opposed to the advantage that it could afford. In that case, it becomes more critical to decide whether or not to treat a given case.

A suitable threshold is not identified beforehand. One solution is to define and optimize the threshold empirically to obtain maximal gain instead of a random fixed value. The threshold is used to ensure that a given case has a high probability of ending with the undesired outcome, i.e., $P_{uout} > \tau$.

Definition 1. ***Cost with no intervention.*** *$cost(c_{id}, T_{i=0})$ The cost when c_{id} ends with an undesired outcome without applying the intervention; therefore, $i = 0$ is shown in Eq. 1. The P_{uout} is the estimated probability of the undesired outcome from the predictive model, and c_{uout} is the cost of the undesired outcome.*

$$cost(c_{id}, T_{i=0}) = P_{uout} * c_{uout} \tag{1}$$

Definition 2. ***Cost with intervention.*** *$cost(c_{id}, T_{i=1})$ The cost when c_{id} ends with an undesired outcome with applying the intervention; therefore, $i = 1$ is shown in Eq. 2. The $CATE_1$ is the estimated causal effect of applying $T_{i=1}$ to c_{id} resulting from the ORF model. c_{T_1} is the cost of employing $T_{i=1}$ to c_{id}.*

$$cost(c_{id}, T_{i=1}) = (P_{uout} - CATE_1) * c_{uout} + c_{T_1} \tag{2}$$

Now, we have the costs with ($cost(c_{id}, T_{i=1})$) and without ($cost(c_{id}, T_{i=0})$) the intervention, the estimated probability (P_{uout}), and $CATE_1$ in our pocket. The next step is defining the *gain* from applying $T_{i=1}$ to c_{id} that enables the highest cost reduction based on Eqs. 1 and 2, as shown in Eq. 3. The gain decides whether or not to treat c_{id}, which solves the first part of our problem.

Definition 3. ***Gain.*** $gain(c_{id}, T_{i=1})$

$$gain(c_{id}, T_{i=1}) = cost(c_{id}, T_0) - cost(c_{id}, T_{i=1}) \tag{3}$$

For example, suppose we have an event log with six cases (see Table 1), the $c_{uout} = 20$, and the $c_{T_1} = 1$. We have two situations where we do not calculate the costs with and without intervention and, therefore, the gain. The first one is presented with $c_{id} = C$ where the estimated probability is below a certain threshold, for instance, $\tau = 0.5$. The other one is given with $c_{id} = F$, where there is no positive effect of applying intervention to the case; though, the $P_{uout} > \tau$. Other cases fulfill the conditions of having $P_{uout} > \tau$ and $CATE_1 > 0$.

Table 1. An example of defining gain.

c_{id}	P_{uout}	c_{uout}	c_{T_1}	$CATE_1$	$cost(c_{id}, T_0)$	$cost(c_{id}, T_{i=1})$	$gain(c_{id}, T_{i=1})$
A	0.55	20	1	0.3	11	6	5
B	0.64	20	1	0.12	12.8	11.4	1.4
C	**0.4**	20	1	–	–	–	–
D	0.8	20	1	0.13	16	14.4	1.6
E	0.9	20	1	0.22	18	14.6	3.4
F	0.51	20	1	**−1.2**	–	–	–

The second part of the problem is deciding when we treat a given case assuming that intervention fulfills the required conditions, i.e., $P_{uout} > \tau$ and $CATE_1 > 0$. We use the *resource allocator* to tackle this part.

The resource allocator monitors the availability of resources to allocate them efficiently. Allocating resources to c_{id} raises another question: how long, i.e., treatment duration, the allocated resource is blocked to apply $T_{i=1}$.

A simple way to define the treatment duration (hereafter T_{dur}) is to set it as a fixed value based on the domain knowledge. However, the variability of T_{dur} might affect the net gain; therefore, we examine three different distributions for the T_{dur}, i.e., fixed, normal, and exponential.

Finally, based on the domain knowledge that tells us how many resources are available to apply $T_{i=1}$, we keep an ordered list of the max gains for each running case c_{id}. Once we have an available resource, we allocate it to apply $T_{i=1}$ to c_{id} with the max gain in our ordered list and block it for T_{dur}.

For example, in Table 1, suppose res_1 and res_2 are available. First, we allocate res_1 to $c_{id} = A$ and res_2 to $c_{id} = B$ and block them for T_{dur}. Then, $c_{id} = D$ enters; but, we can not treat it since there are no available resources.

Accordingly, we keep $c_{id} = D$ and $c_{id} = E$ (that comes later) in our sorted list. We assume that cases keep coming to our system, implying that the sorted list may eventually be extended with other cases with positive gains. Whenever a resource becomes available, if a case in the sorted list has a positive gain, we allocate resources to the one with max gain. Moreover, our approach allocates resources to different cases simultaneously, and various instances update the available resources.

4 Evaluation

We conducted an evaluation to address the following research questions:

RQ1. To what extent the total gain depends on the number of available resources?

RQ2. To what extent the total gain depends on the variability of the T_{dur}?

RQ3. When allocating resources to cases with higher gain versus cases with higher undesired outcome probability, what is the total gain?

4.1 Dataset

We evaluate our approach using one real-life event log, namely *BPIC2017*[1], corresponding to a loan origination process. In this event log, each case corresponds to a loan application. Each application has an outcome. The desired one occurs when offering clients a loan, and clients accept and sign it. While the undesired one occurs when the bank cancels the application or the client rejects the offer. The log contains $31,413$ applications and 1,202,267 events.

We used all possible attributes in the log as input to the predictive and causal models. Furthermore, we extracted other features, e.g., the number of offers, event number, and other temporal information, e.g., the hour of the day, day of the month, and month. We extracted prefixes of length less than or equal to the 90^{th} percentile of the case lengths in the log to avoid bias from long cases. We encoded the extracted prefixes using *aggregate encoding* to convert them into a fixed-size feature vector (see Sect. 2.1).

To obtain the best performance of either predictive or causal models, event log, i.e., a *loan application process*, preprocessing is an essential step. In addition to the preprocessing given by [12], we define the outcome of cases based on the end activity. We represent cases that end with *"A_Pending"* events as a positive outcome, where cases that have *"A_Denied"* or *"A_Cancelled"* events are adverse outcomes that need intervention. Then, we define the intervention that we could apply to minimize the unsuccessful loan applications based on the winner report of the BPIC challenge [10]. They report that making more offers to clients increases the probability of having *"A_Pending"* as an end stat. Accordingly, we represent cases with only one offer to be treated where $T = 1$. In contrast, cases with more than one offer should not be treated, then $T = 0$.

[1] https://doi.org/10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b.

4.2 Experiment Setup

We use an *XGBoost*[2] model to estimate the probability of negative case outcomes, i.e., P_{uout}. XGBoost has shown good results on various classification problems [6], including outcome-oriented PPM [12]. We use the following parameters to train the XGBoost model: learning rate of 0.2, subsample of 0.89, max tree depth of 14, Colsample by tree of 0.4, and a min child weight of 3.

We use *ORF* to estimate the $CATE$ as implemented in the *EconMl*[3] package. We use the following parameters: min leaf size of 50, a max depth of 20, a subsample ratio of 0.4, and lambda regularization with parameter 0.01.

The predictive and causal models follow the same workflow as any machine learning problem. We temporally split the log into three parts (60% - 20% - 20%) to simulate real-life situations to tune and evaluate these models. Mainly, we arrange cases using their timestamps. We use the opening 80% for training (60%) and tuning (20%), and the rest (20%) to evaluate model performance. Table 2 shows the configurations of the proposed approach.

We vary the c_{uout} values to make them more significant than the c_{T_1} value to give a meaningful result. We found that the higher c_{uout} related to c_{T_1}, the more net gain. Accordingly, we applied the higher value of the c_{uout} in our experiments with different treatment distributions and an empirically optimized threshold to answer our research questions.

We assume that the estimated $CATE$ is accurate and, hence, allocating resources will decrease a case's probability of a negative outcome. We compare our approach to a purely predictive baseline proposed in [5], where the interventions are triggered as soon as $P_{uout} > \tau$. In other words, we allocate resources to cases with the highest P_{uout} instead of cases with max gain, and we consider the $CATE$ as the new gain we achieve from treating cases.

Table 2. Configurations of the proposed approach

$\#resources$	c_{uout}	c_{T_1}	τ	T_dur (sec)
$1, 2, \ldots 10$	$1, 2, 3, 5, 10, 20$	1	$0.5, 0.6, \ldots 0.9$	Fixed $= 60$ Normal $\in \{1, 60\}$ Exponential $\in \{1, 60\}$

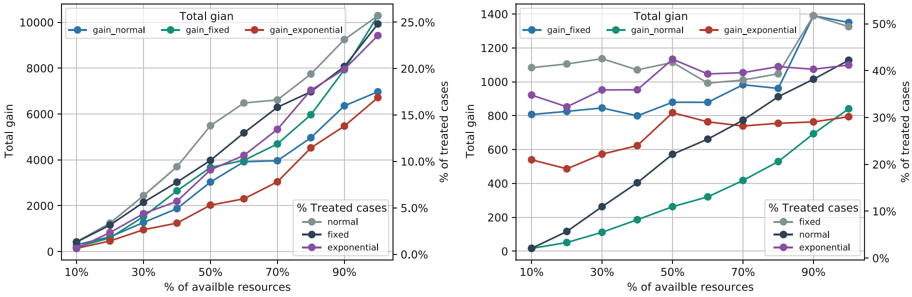
4.3 Results

We present the results of our proposed approach by exploring the effects of available resources on the total gain and the percentage of treated cases, taking into account the variability of T_{dur} (RQ1 and RQ2). Figure 5a shows how the total gain and percentage of treated cases evolve as we increase the number of available resources (RQ1). When the number of available resources increases, both metrics increase. Meanwhile, if the available resources reach above 50%, the

[2] https://github.com/dmlc/xgboost.

[3] https://github.com/microsoft/EconML.

total gain almost increases considerably. For example, with fixed distribution in Fig. 5a, when the percentage of available resources is above 50%, the total gain rises markedly compared to the situation where the rate of resources is below 50%. That is because more cases are treated when more than half of the resources become available.



(a) RQ1 and RQ2

(b) RQ3

Fig. 5. Total gain and % of treated cases w.r.t % available resources

Moving to RQ2, we experiment with three T_{dur} distributions, i.e., fixed, normal, and exponential. Figure 5a shows that the fixed distribution gives more net gain because there is less variability in the distribution of resources among cases that need intervention than normal and exponential distributions where the level of variability decreases, respectively. Accordingly, the net gain highly depends on the variability of treatment duration.

To answer RQ3, we allocate resources to cases with the highest P_{uout} instead of cases with max gain. We consider the $CATE$ a new gain from treating cases (see Fig. 5b). Therefore, we need a threshold τ to determine whether or not to intervene depending on the P_{uout}. There are two approaches to set a threshold: first, and based on a given threshold, e.g., $\tau = 0.5$, if there are available resources and the undesired outcome above the given threshold, we trigger an intervention. The second is to use an empirical threshold proposed by [5], where authors compute an optimal threshold based on historical data. We varied the threshold as shown in Table 2. However, the results are different based on the T_{dur} distribution. Where $\tau = 0.5$, the normal distribution gives more net gain than other thresholds. While $\tau = 0.6$, the exponential distribution delivers the higher net gain. Moreover, with $\tau = 0.7$, the fixed distribution wins. The results of optimizing the threshold are available in the supplementary material[4], where we show how the total gain changes w.r.t different thresholds and different T_{dur}.

We observe that our approach consistently leads to higher net gain, under the same amount of consumed resources, than the purely predictive baseline. For example, under a fixed distribution, treating 25% of cases with our approach

[4] https://zenodo.org/record/5538113#.YVSdhSVRV8I.

(cf. Fig. 5a) leads to a net gain of 10000, while in the predictive method (Fig. 5b), treating twice more cases (50% of cases) yields a net gain of only 1400. This suggests that the combination of causal inference with predictive modeling can enhance the efficiency of prescriptive process monitoring methods.

4.4 Threats to Validity

The evaluation comes with an external validity threat (lack of generalizability) due to its reliance on only one event log. The evaluation is preliminary and ought to be followed up with additional experiments using other datasets.

We simulated a scenario where we could trigger interventions at any time point. Also, we assume that the effect of this intervention will be to reduce the probability of adverse outcomes by the estimated $CATE$. There is a threat to ecological validity because the $CATE$ might not reflect the actual treatment effect due to unobserved confounders.

The evaluation is limited to one feature encoding method and one machine learning algorithm. Experimenting with other encodings and algorithms is a direction for future work.

5 Conclusion

We introduced a prescriptive monitoring approach that triggers interventions in ongoing cases of a process to maximize a net gain function under limited resources. The approach combines a predictive model to identify cases that are likely to end in a negative outcome (and hence create a cost) with a causal model to determine which cases would most benefit from the intervention in their current state. These two models are embedded into an allocation procedure that allocates resources to case interventions based on their estimated net gain. A preliminary evaluation suggests that it treats fewer cases and allocates resources more effectively than a baseline method that relies only on a predictive model.

In the proposed approach, an intervention is triggered whenever the estimated net gain of treating this case is maximal, relative to other cases. Under some circumstances, this may lead to treating a case at a suboptimal time. For example, in a loan origination process, calling a customer two days after sending an offer may be more effective than doing so just one day after the offer. The expected gain is not just depending on utilizing the intervention. It rather depends on the time we trigger the intervention. Accordingly, If we decide to wait until the state of the cases changes and do not intervene, it will reduce the uncertainty and probably achieving more gain. Our approach would trigger the intervention “call customer” one day after the offer if the expected benefit is positive and there is no other case with a higher net gain. An alternative approach would be to allocate resources based on the estimated net gain of a case intervention at the current time and the expected gain of intervening in the same case at a future time. An avenue for future work is to combine the proposed approach with a method that optimizes the time point when an intervention is triggered in a case. A related avenue for future work is to consider constraints on the moment when interventions can be triggered on a case. For example, calling a customer to follow up on a loan offer does not make sense if the loan offer has been canceled or the customer has not yet received a loan offer.

Another limitation of the proposed approach is that it assumes that there is a single type of intervention. In reality, there may be multiple types of interventions (e.g., call the customer, send a second loan offer). Another future work direction is to handle multiple types of interventions.

Reproducibility. The implementation and source code of our approach can be found at https://github.com/mshoush/PrescriptiveProcessMonitoring.

References

1. Athey, S., Tibshirani, J., Wager, S.: Generalized random forests. Ann. Stat. **47**(2), 1148–1178 (2019)
2. Bozorgi, Z.D., Teinemaa, I., Dumas, M., La Rosa, M.: Prescriptive process monitoring for cost-aware cycle time reduction. In: ICPM (2021)
3. Bozorgi, Z.D., Teinemaa, I., Dumas, M., La Rosa, M., Polyvyanyy, A.: Process mining meets causal machine learning: discovering causal rules from event logs. In: ICPM, pp. 129–136. IEEE (2020)
4. Chernozhukov, V., et al.: Double/debiased machine learning for treatment and structural parameters (2018)
5. Fahrenkrog-Petersen, S.A., et al.: Fire now, fire later: alarm-based systems for prescriptive process monitoring. arXiv preprint arXiv:1905.09568 (2019)
6. Fernández-Delgado, M., Cernadas, E., Barro, S., Amorim, D.: Do we need hundreds of classifiers to solve real world classification problems? J. Mach. Learn. Res. **15**(1), 3133–3181 (2014)
7. Guo, R., Cheng, L., Li, J., Hahn, P.R., Liu, H.: A survey of learning causality with data: problems and methods. ACM Comput. Surv. **53**(4), 1–37 (2020)
8. Metzger, A., Kley, T., Palm, A.: Triggering proactive business process adaptations via online reinforcement learning. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) BPM 2020. LNCS, vol. 12168, pp. 273–290. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58666-9_16
9. Oprescu, M., Syrgkanis, V., Wu, Z.S.: Orthogonal random forest for causal inference. In: ICML, pp. 4932–4941. PMLR (2019)
10. Povalyaeva, E., Khamitov, I., Fomenko, A.: BPIC 2017: density analysis of the interaction with clients. BPI Challenge (2017)
11. Teinemaa, I., Dumas, M., Leontjeva, A., Maggi, F.M.: Temporal stability in predictive process monitoring. Data Min. Knowl. Disc. **32**(5), 1306–1338 (2018). https://doi.org/10.1007/s10618-018-0575-9
12. Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M.: Outcome-oriented predictive process monitoring: review and benchmark. ACM TKDD **13**(2), 1–57 (2019)
13. Weinzierl, S., Dunzer, S., Zilker, S., Matzner, M.: Prescriptive business process monitoring for recommending next best actions. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) BPM 2020. LNBIP, vol. 392, pp. 193–209. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58638-6_12
14. Xu, G., Duong, T.D., Li, Q., Liu, S., Wang, X.: Causality learning: a new perspective for interpretable machine learning. arXiv preprint arXiv:2006.16789 (2020)

Quantifying Explainability in Outcome-Oriented Predictive Process Monitoring

Alexander Stevens(✉), Johannes De Smedt, and Jari Peeperkorn

Research Centre for Information Systems Engineering, KU Leuven, Leuven, Belgium
alexander.stevens@kuleuven.be

Abstract. The growing interest in applying machine and deep learning algorithms in an Outcome-Oriented Predictive Process Monitoring (OOPPM) context has recently fuelled a shift to use models from the explainable artificial intelligence (XAI) paradigm, a field of study focused on creating explainability techniques on top of AI models in order to legitimize the predictions made. Nonetheless, most classification models are evaluated primarily on a performance level, where XAI requires striking a balance between either simple models (e.g. linear regression) or models using complex inference structures (e.g. neural networks) with post-processing to calculate feature importance. In this paper, a comprehensive overview of predictive models with varying intrinsic complexity are measured based on explainability with model-agnostic quantitative evaluation metrics. To this end, explainability is designed as a symbiosis between interpretability and faithfulness and thereby allowing to compare inherently created explanations (e.g. decision tree rules) with post-hoc explainability techniques (e.g. Shapley values) on top of AI models. Moreover, two improved versions of the logistic regression model capable of capturing non-linear interactions and both inherently generating their own explanations are proposed in the OOPPM context. These models are benchmarked with two common state-of-the-art models with post-hoc explanation techniques in the explainability-performance space.

Keywords: Predictive Process Monitoring · XAI · Machine Learning · Deep Learning

1 Introduction

Sparked by the growing research on machine learning, the analysis of processes through data-driven approaches has seen a surge under the label process mining [1]. Recently, the subtrack of predictive process monitoring [14] has known a strong uptake as it allows identifying trends in processes concerning the obtainment of particular goals (e.g. will customers be awarded credit?), impeding bottlenecks, and whether particular activities will occur in the future. As the concrete goal is to predict the future state as accurately as possible, the anticipated trend

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 194–206, 2022.
https://doi.org/10.1007/978-3-030-98581-3_15

is to increase the predictive performance with the use of deep learning instead of the more classical machine learning models [4,14]. The intrinsic complexity of these predictive models, however, causes a lack of transparency of the model, intertwined with the inability to interpret the predictions. In predictive process monitoring, several papers have already suggested model-agnostic explainability techniques on top of the machine learning models, e.g. SHAP/LIME [10], with similar developments in a deep learning context [3,12]. Nevertheless, the accuracy by which these post-hoc explainability techniques reflect the effective model behaviour of the predictive model is often inadequate. Moreover, identifying faithful explanations that are interpretable remains a challenge for black box models. This has already been raised by XAI proponents [11], advising to adopt inherently interpretable models rather than trying to explain black box models when it comes down to high-stake decision-making. Nonetheless, a description of the technical limits of post-hoc XAI techniques is deemed out of scope.

To the best of our knowledge, none of these works have addressed the systematic comparison of interpretable models versus post-hoc explainability in the context of OOPPM. To this end, this paper introduces a definition of explainability that allows to compare different predictive models based on model-agnostic quantitative measures. Furthermore, given the need for more inherently interpretable machine learning models which excel in terms of the predictive performance–explainability trade-off [5,11], the Logit Leaf Model (LLM) and the Generalized Logistic Rule Model (GLRM) are adapted to the OOPPM context. The former naturally clusters the data with a decision tree and builds linear models which are directly interpretable in the leave nodes. The latter creates binary rules from the input data with a generalized logistic rule model which is less transparent but has been shown to outperform even very intricate inference mechanisms such as neural networks [16]. We benchmark these techniques with two established techniques aimed at introducing post-hoc explainability, i.e., XGBoost with Shapley values, and recurrent neural networks with attention. The rest of the paper is organized as follows. First, Sect. 2 provides a brief overview of the preliminaries. Next, Sect. 3 defines how explainability can be quantitatively measured, while introducing two models to the field of OOPPM. This is followed by an experimental evaluation in Sect. 4, where the experimental setup, the implementation details and the results are reported. Finally, the models are compared alongside the conclusion in Sect. 5.

2 Preliminaries

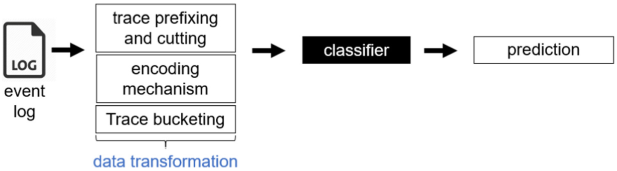


Fig. 1. Preliminary steps (simplified)

OOPPM relies on the use of historic process data recorded in event logs. An event log is a list of traces which represent the enactment of a particular case within an information system [1]. Moreover, a trace is considered as a sequence of timestamped events which are tuples of p features $[x_1, \ldots, x_p]$ such as an activity name, timestamp, and so on. On the other hand, machine and deep learning models such as the ones used for OOPPM are built to work on tabular data, with every row representing a new instance while having a fixed length of feature values. In order to use event logs in combination with AI models, the data transformation steps from Fig. 1 need to be applied.

First, (trace) prefixes are extracted from the completed cases to be able to learn, preferably incrementally, from the development of the traces. To this end, a prefix log is typically derived, which is the extracted event log that contains all the prefixes of each case in the original event log.

The second data transformation step describes the encoding mechanism [15] that enables the user to work with a varying amount of features. An often used encoding is frequency aggregation, which takes the frequencies of the categorical values while calculating the summary statistics of the numeric values (min, max, mean, sum and std). This transformation step results in a trace prefix of indefinite length being displayed as a row with a fixed amount of features, with $x'_{i,j}$ the frequency/statistics of instance $i \in [1 \ldots n]$ on the transformed feature x'_j, $j \in [1 \ldots p]$. Nonetheless, this encoding mechanism neglects the order of the timestamped events and therefore results in a loss of information. By contrast, the use of frequency aggregation in step-based models such as recurrent neural networks becomes superfluous given their sequential setup. To exploit this efficiently, a low-dimensional representation of discrete features in the form of embeddings is an often performed encoding [12,14]. This mapping transforms the categorical feature to a vector of continuous numbers in a meaningful way. The use of embeddings is preferred over one hot-encoding, where high-cardinality features cause the feature space to explode while simultaneously ignoring the similarity between these vectors.

The last step before the data can be fed to a model is trace bucketing, a commonly used data transformation step that supports the discovery of heterogeneous segments in the data while creating separate models for each of them [15]. Techniques such as K-nearest neighbours or K-Means clustering measure the (dis)similarity between traces depending on the parameter k. However, while bucketing can effectively diminish the runtime performance [15], the clustering does not necessarily result in an intuitive or interpretable outcome. E.g., clustering techniques can base their grouping on a high number of dimensions that are not interpretable. Furthermore, there is no guarantee that the use of a bucketing technique will effectively improve performance [15]. The above was tested by benchmarking against the single bucketing technique, in which only one bucket is created [15].

As a final step, the model can be used to make certain predictions $y^* = F(x_j)$ with F the model/function to make a prediction based on the features x_j with $j \in [1 \ldots p]$.

3 Explainability in OOPPM

This Section introduces a general definition of explainability based on XAI concepts, which can be used to evaluate OOPPM techniques. Next, two interpretable models with varying model complexity are introduced to OOPPM. The Logit Leaf Model (as presented in this paper) is a transparent and interpretable model. The second algorithm is the Generalized Logistic Rule Model that uses column generation to find the optimal set of rule-based features and able to create its own explanations.

3.1 Explainability Through Interpretability and Faithfulness

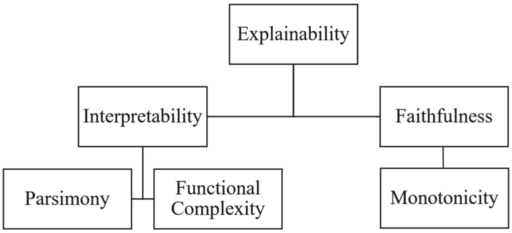


Fig. 2. Explainability Through Interpretability and Faithfulness

The use of the predictive models for high-stake decision-making processes is finding its way to ever more applications. While simple models are able to generate their own explanations, XAI tries to approximate the behaviour of the model with post-hoc explainable techniques such as e.g. Shapley values, feature importance, etc. This leads to a widespread urge to evaluate the faithfulness (and interpretability) of these models, by looking at whether the original task model (e.g. black box model) is accurately reflected by the explainability model (e.g. post-hoc explainability technique).

First of all, even though often used interchangeable, there is a subtle difference between interpretability and explainability. This boils down to the fact that understanding the internal working of the model is different from the ability to link the inputs with its predicted output in a faithful way. Moreover, an unambiguous and simple interpretation of a prediction has a substantial loss of trustworthiness if it does not accurately represent the effective behaviour of the model. To emphasize, a simple explanation generated for a rain forecast prediction could be: *'if the grass is green, it will rain'*, which is easy to interpret, but is unfaithfulness to the actual behaviour of rain.

The necessity to distinguish between faithfulness and interpretability was defined by [5] and is adapted in this paper for OOPPM purposes. As a result, the combination of *interpretability* (further decomposed in parsimony and functional complexity) and *faithfulness* (by means of monotonicity), allows quantifying and thus evaluate explainability in an OOPPM context (see Fig. 2).

Parsimony is a property of interpretability that discusses the complexity of the model [5], an often used metric for linear regression models. In this paper, the parsimony of a model F is quantified by the amount of features in the resulting model. This can be seen as the number of features with non-zero weights e.g., linear regression coefficients with a corresponding weight different from 0, in other cases the non-zero weights provided by the post-hoc feature importance. As a result, the parsimony of a model is maximally equal to the total amount of features. Moreover, a parsimonious (i.e. simple) model corresponds to a low value for C_F.

Assume features x_i with a_i the weight of the features $i \in [1 \ldots p]$ indicated by the feature importance of a model, where the total parsimony C_F is calculated as followed:

$$C_F = \sum_{i=1}^{p} C(i) \text{ with}$$

$$C(i) = \begin{cases} 0, & \text{if } a_i > 0, \\ 1, & \text{otherwise.} \end{cases} \tag{1}$$

Functional complexity is, alongside parsimony, a metric of model complexity and measures how strong the model is dependent of the features [8].

Assume the prediction of an instance i by the model F is indicated by $\hat{y}_i = F(x_{i,1}, \ldots, x_{i,p})$. Furthermore, the prediction after feature permutation is defined by $\hat{y}_i^* = F(x_{i,1}, ., x_{i,j}^*, ., x_{i,p})$, where $x_{i,j}^*$ is a randomly permuted feature value. The total functional complexity is calculated as the amount of prediction changes *before* and *after* permutation for all the instances and features, divided by the number of instances and the parsimony C_F of the model. Therefore, the functional complexity is quantified by regarding a feature as 'used' when changing the feature changes the prediction. A lower value for U_F means less model complexity and therefore higher model interpretability. This paper introduces a slightly different perspective from the original computation [8], where the functional dependency of a model was originally determined by examining *how often* the model predictions change when changing the value $x_{i,j}$ of an instance $i \in [1 \ldots n]$ on a feature $j \in [1 \ldots p]$.

$$U_F = \frac{1}{n} \frac{1}{C_F} \sum_{j=1}^{p} \sum_{i=1}^{n} u_{i,j} \text{ with}$$

$$u_{i,j} = \begin{cases} 1, & \text{if } \hat{y}_i \neq y_i^*, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Monotonicity is the notion that describes the extent to which the feature importance ranking of the explainability model is faithful to the ranking of the task model feature importance. By definition, monotonicity is ensured (i.e. M = 1) if the model is able to create its own explanations, i.e. inherently generated explanations [9]. The monotonicity of a (post-hoc explainability) model therefore needs to be quantified by the Spearman's correlation coefficient [9].

$$M = \rho(a, e)$$

with $a = (|a_1|, ..., |a_p|)$ containing the absolute values of the feature weights of the task model and $e = (|e_1|, ..., |e_p|)$ the absolute values of the feature weights of the explainability model. This correlation coefficient is a non-parametric measure that takes a value between [−1, 1] and describes the association of rank. A perfectly faithful model has a correlation coefficient of +1, where a loss in faithfulness corresponds with a value closer to 0. Consequently, a negative value corresponds to a negative rank association between the two feature importance weights.

3.2 Logit Leaf Model

Evidently, the use of inherently interpretable models for high-stake decision-making is preferred over black box models [11], which leads to the introduction of the Logit Leaf Model. This model is constructed as a hybrid of logistic regression and decision tree clustering, with the former model applied in the leaves of the latter to create predictions. An initial version was proposed by [2] in the context of churn prediction where in each leaf node, only the variable that contributes to the maximal Akaike Information Criterion decrease is added, until the stopping criteria are met. Here, we adapt the general idea to the context of OOPPM.

The adapted algorithm starts similarly as a simple decision tree, with the tree splitting the data in an iterative manner into smaller, more homogeneous samples. All the possible splits are evaluated based on the information gain obtained by the decrease in entropy, where after the best possible split is performed. The stopping decision consist out of two different approaches: a maximum number of tree levels together with a minimal number of samples in a leaf. Hence, all the possible models that abide the stopping decision rules are evaluated based on their final predictive performance where only the optimal model is returned. Lastly, the assignment decision of the leave nodes is revised, where a logistic regression model is learnt for each segment of the data instead of performing majority voting.

The strength of decision trees lies in the ability to discover XOR-style interactions, as opposed to the model failing when it comes to discovering linear relations between the predictor variables. By contrast, logistic regression models are unable to deal with these interaction effects but manage to handle linear relations well. As a result, the logit leaf model overcomes both disadvantages while exploiting their respective strengths. The nature of this model ensures that the explanations consist out of a combination of decision tree rules and logistic regression coefficients, making it a faithful model by design.

The model does, however, have two major drawbacks. Firstly, while decision trees can capture the interaction effects, the logistic regression model might not deal with them well in case they are still left in impure splits. This can result in reduced predictive performance. Secondly, the model can still become large when many coefficients have higher/lower values, impeding the overall interpretability.

3.3 Generalized Logistic Rule Model

The second introduced model is the Generalized Logistic Rule Model [16], which combines the linear elements from logistic regression together with a conjunctive ruleset, where each rule is constructed as a conjunction of binarized features making the GLRM model a rule ensemble [6]. Here, the assumption of a Gaussian distribution of the residuals in a general linear regression model is relaxed and generalized for different distributions. By induction, a logistic regression model is therefore also a generalized linear model [16].

In order to find the optimal set of rule-based features, column generation is performed using integer programming to improve the objective function [16]. First, the GLRM model transforms the original features to rule-based features before making predictions. To this end, numerical features are binarized through bi-directional comparisons to a set of thresholds, while categorical features are one-hot encoded.

In the final model, the probability of y being classified as 'deviant' is predicted as $log(z)$, where z is a linear combination of the discovered rules.

$$log(z) = \frac{1}{(1 + e^{-z}))} \tag{3}$$

Similar to a logistic regression, the GLRM model is able to create coefficients for a single feature, but by extension also for an AND-combination of two features. Furthermore, the rule-based features can handle both linear and non-linear dependencies (analogous to LLM) as an improvement over the competencies of the logistic regression model. The strength of the GLRM model lies in the ability to reduce the amount (and length) of rules with the use of regularization parameters, a means to improve the interpretability of the rules. Moreover, λ_0 denotes the fixed cost of each rule (penalizes the amount of the rules), while λ_1 is the additional cost of each literal in rule (penalizes the length of a rule). This ensures that GLRM can compete directly in terms of performance while providing its own, relatively simple explanations.

The intrinsic complexity of this model, i.e. a column generation subproblem solved using integer programming, can be seen as a drawback in the context of high-stake decision making. Next, generalized linear models are known to be sensitive to outliers. The last drawback is the need of relatively large sample sizes, and an exponential increase of binarized features with an increase in the amount of predictor variables.

4 Experimental Evaluation

In this Section, the two benchmark models are briefly discussed, while indicating which post-hoc explainability techniques are used on top of these black box models. Next, the different event logs and their corresponding statistics are elaborated on. This is followed by detailed information about the implementation steps performed in this experiment. Finally, an overview of the quantitative metric results is given, which is subsequently summarized in Sect. 5.

4.1 Benchmark Models

XGBoost is one of the most widely-used ensemble methods in machine learning due to its ability to outperform most of the existing models. Several studies in a predictive process monitoring context have already used this gradient boosting machine [15], where weak learners are improved after each iteration to a final strong learner by incorporating the loss function of the previous weak learner(s). As this is a black box model, Shapley values need to be calculated in order to explain the model [13]. The Shap value for each instance-feature combination is obtained, whereby the calculation is based on coalitional game theory. Therefore, XGBoost is not an interpretable model, as the inherit complexity is what bestows the predictive abilities on this black box model. The feature importance is calculated by the average of the amount that each feature split point improves the purity (i.e. Gini index) weighted by the number of instances in the respective nodes, across all the decision trees within the model.

The second model is a recurrent neural network with Long Short-Term Memory (LSTM), with the long-term relations and dependencies encoded in the cell state vectors, therefore solving the vanishing gradient problem. The advantage of LSTM over classical machine learning models lies in the ability to model time-dependent and sequential data tasks, where the categorical values are encoded in embeddings. Similar to XGBoost, the complexity of the internal representation of an LSTM neural network does not allow for inherent explanations of predictions. Consequently, recent work in deep learning to predict the next activity have come with solutions to provide post-hoc explanations [3,7], whereby the use attention layers to create post-hoc explainability in predictive process monitoring stems from [12]. In [3], an LSTM model in combination with Shapley values allow the user to identify the influence of certain features in the different steps of the process, while [7] focuses on creating local post-hoc explanations with the use of a surrogate decision tree. In addition, [17] introduces a widely-used approach of machine learning to offer explainability in the light of deep neural networks for remaining-time and next-activity predictions respectively. Finally, in the case of long short-term neural networks, the feature importance of the task model is approximated with the use of a perturbation method.

4.2 Event Logs

The first event log TF1 contains notifications from an Italian local police force, e.g. the reason, the total amount, and the amount of repayments. The original event log has 1,198,366 events divided over 129,615 cases (see Table 1). The second event log BPIC2017 assembles the execution history of a loan application process in a certain Dutch financial institution. The dataset contains events related to a particular loan application, with the label indicating if the loan application was accepted (regular), or not (deviant). The last event log BPIC2015 assembles events from the second Dutch municipality (see [15]), pertaining to the building permit application process. The different event logs can be found at the website of 4TU Centre for Research Data[1].

Table 1. Event Logs

event log	events	cases	cutoff len.	features (orig.)	features (agg. enc.)	features (select.)
TF1	460,556	129,615	10	21	254	3
BPIC2017	1,198,366	31,413	20	26	259	10
BPIC2015	41,202	753	40	21	391	50

4.3 Implementation

The event logs are split on an 80/20 ratio, with the cases ordered based on their timestamp and only the first 80% used for training purposes (after cutting the events of the training cases that overlap with the test period), an analogue implementation approach to [15]. Next, the data transformation steps as described in Fig. 1 are performed. After trace prefixing and cutting (with a predefined cut-off length), different sequence encoding techniques are implemented for the machine learning algorithms (i.e. aggregation encoding) and the LSTM model (i.e. embedding). Therefore, the total amount of deduced columns *after aggregation encoding* in Table 1 is only applicable for the machine learning models. Lastly, no trace bucketing technique has been applied for any of the respective models. Instead, in order to improve runtime performance and interpretability, feature selection is performed, where only uncorrelated predictor features (with $\geq 10\%$ Pearson correlation with the target feature) are selected, which made trace bucketing unnecessary. Furthermore, the hyper optimization for the machine learning models is performed with the use of hyperopt[2], while the LSTM neural network has an analogue setting to [12], with the predictive function transformed into a binary outcome-oriented prediction by stripping of the final layer and inserting a sigmoid output layer instead. As a final remark, detailed information about design implementations and parameters are provided, to enhance reproducible results[3].

[1] https://data.4tu.nl/.

[2] http://hyperopt.github.io/hyperopt/.

[3] https://github.com/AlexanderPaulStevens/OOPPM.

Table 2. Quantitative Metrics (overview)

	Traffic Fines				BPIC2017				BPIC2015			
	LLM	**GLRM**	**XGB**	**LSTM**	**LLM**	**GLRM**	**XGB**	**LSTM**	**LLM**	**GLRM**	**XGB**	**LSTM**
Parsimony	2	3	3	2	9	6	10	7	18.5	13	40	15
Functional Complexity	0.00	0.00	0.57	0	0.97	0.55	0.25	0.00	0.29	0.15	1.24	0.12
Monotonicity	1	1	1	−1	1	1	0.42	−0.43	1	1	0.31	−0.12

4.4 Quantitative Metrics Results

The parsimony of a model is described with the use of an absolute number instead of, e.g., a ratio, allowing for comparability between event logs with different dimensions. Intuitively, the parsimony of a model displays the amount of features used in the explainability model, with on average the highest value denoted for the XGBoost model in contrast to the lowest value for the rule-based GLRM. Furthermore, the parsimony of the LLM tends to increase more compared to the LSTM with an increased amount of predictor variables.

The functional complexity of each model describes the dependency of the model on the features, and only makes sense when analysing the same event log. Again, the XGBoost model reports the highest value on average, while the LSTM neural network has the lowest functional dependency (on average) on its features. This intuitively boils down to the fact that changing a value in an LSTM neural network has a smaller effect on the value of the prediction due to the more complex inference structure that makes for more stable predictions. Furthermore, this metric is by design (as the parsimony metric is in the denominator instead of the total amount of column) able to demonstrate that e.g. the XGBoost model is *also* functionally dependent on feature(s) that were assigned a zero-attribution, as a value >1 for event log BPIC2015 is reported. Further insights are that the functional complexity of the GLRM and LLM do not have a linear relationship with the basic statistics from Table 1, where the functional complexity in the XGB model seems to depend on the number of selected features.

As the faithfulness of both LLM and GLRM are guaranteed by definition (the task model and the explainability model are the same), only monotonicity values for the XGBoost and LSTM model have to be calculated. For the BPIC2017 event log, the Spearman's rank correlation coefficient of the XGBoost vs. Shapley value feature importance is 0.42, where the underlying message is that the Shapley values do not accurately reflect the model behaviour, indicating a loss of faithfulness. For the LSTM neural network, the feature importance calculated with the attention values versus feature importance based on the perturbation importance are the values used to calculate the monotonicity, with a negative value of −0.43 reported in Table 2 for the event log BPIC2017. This interesting value is visualized in Fig. 3, where it is clear that there is uncertainty about the influence of the time component on the model.

Lastly, the performance of the different models over the different event logs show that the introduced models are competitive with the less interpretable models (with an outlier value for the LSTM on the event log BPIC2015).

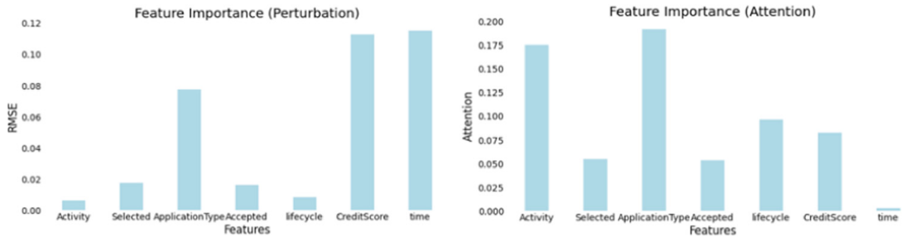


Fig. 3. BPIC2017 LSTM (Attention vs. Perturbation)

5 Conclusion

While data fuels the advances in machine learning and artificial intelligence, the centre of attention has mostly been on the computational aspects, thereby often neglecting the interpretation, actionability, and implications of the results. Moreover, an easily interpretable explanation must also be faithful to the effective model of behaviour, with the quantitative evaluation of explainability techniques on top of models with varying intrinsic complexity as an increased necessity. To this end, this paper has introduced a definition (explainability as a symbiosis between interpretability and faithfulness) and quantitative metrics (parsimony, functional complexity and monotonicity) to evaluate and rank different algorithms based on their explainability. Moreover, it is desirable that a model uses a small amount of features for its predictions (low parsimony), which are as functionally independent as possible (low functional complexity), without compromising on faithfulness (monotonicity of 1).

Furthermore, this paper also introduced two improved versions of the logistic regression model, which were found to have comparative performance results when compared with the two benchmark models. In addition, both the GLRM and LLM show better (or at least comparable) overall results based on parsimony, functional complexity and monotonicity on the three event logs with varying statistics. Lastly, the faithfulness of the explanations is ensured for these models by definition, while the study shows that the post-hoc explainability models on top of the XGBoost and LSTM models are associated with an imperfect faithfulness. As a result, the use of GLRM is recommended (over LLM) due to the overall better results. This predictive model can handle both linear and non-linear dependencies and the amount (and length) of rules can be reduced with the use of regularization parameters, which is favourable to the parsimony and possibly the functional complexity.

Future research consists of evaluating the impact of the index sequence encoding technique, seeking justification for the obtained negative values of the monotonicity of the LSTM, and identifying the most important components for evaluating model explainability. Additionally, this paper will be extended to a benchmarking study with additional methods (classic logistic regression, CNN, etc.) and metrics.

References

1. van der Aalst, W.M.P.: Process Mining - Data Science in Action, 2nd edn. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
2. Caigny, A.D., Coussement, K., Bock, K.W.D.: A new hybrid classification algorithm for customer churn prediction based on logistic regression and decision trees. Eur. J. Oper. Res. **269**(2), 760–772 (2018)
3. Galanti, R., Coma-Puig, B., de Leoni, M., Carmona, J., Navarin, N.: Explainable predictive process monitoring. In: ICPM, pp. 1–8. IEEE (2020)
4. Kratsch, W., Manderscheid, J., Röglinger, M., Seyfried, J.: Machine learning in business process monitoring: a comparison of deep learning and classical approaches used for outcome prediction. Bus. Inf. Syst. Eng. **63**(3), 261–276 (2021). https://doi.org/10.1007/s12599-020-00645-0
5. Markus, A.F., Kors, J.A., Rijnbeek, P.R.: The role of explainability in creating trustworthy artificial intelligence for health care: a comprehensive survey of the terminology, design choices, and evaluation strategies. J. Biomed. Inform. **113**, 103655 (2021)
6. McCullagh, P., Nelder, J.A.: Generalized Linear Models. Springer, Heidelberg (1989)
7. Mehdiyev, N., Fettke, P.: Explainable artificial intelligence for process mining: a general overview and application of a novel local explanation approach for predictive process monitoring. CoRR abs/2009.02098 (2020)
8. Molnar, C., Casalicchio, G., Bischl, B.: Quantifying model complexity via functional decomposition for better post-hoc interpretability. In: Cellier, P., Driessens, K. (eds.) ECML PKDD 2019. CCIS, vol. 1167, pp. 193–204. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-43823-4_17
9. Nguyen, A., Martínez, M.R.: On quantitative aspects of model interpretability. CoRR abs/2007.07584 (2020)
10. Rizzi, W., Di Francescomarino, C., Maggi, F.M.: Explainability in predictive process monitoring: when understanding helps improving. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) BPM 2020. LNBIP, vol. 392, pp. 141–158. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58638-6_9
11. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. **1**(5), 206–215 (2019)
12. Sindhgatta, R., Moreira, C., Ouyang, C., Barros, A.: Exploring interpretable predictive models for business processes. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) BPM 2020. LNCS, vol. 12168, pp. 257–272. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58666-9_15
13. Sindhgatta, R., Ouyang, C., Moreira, C.: Exploring interpretability for predictive process analytics. In: Kafeza, E., Benatallah, B., Martinelli, F., Hacid, H., Bouguettaya, A., Motahari, H. (eds.) ICSOC 2020. LNCS, vol. 12571, pp. 439–447. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-65310-1_31

14. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive business process monitoring with LSTM neural networks. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 477–492. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_30
15. Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M.: Outcome-oriented predictive process monitoring: review and benchmark. ACM Trans. Knowl. Discov. Data **13**(2), 17:1–17:57 (2019)
16. Wei, D., Dash, S., Gao, T., Günlük, O.: Generalized linear rule models. In: ICML. Proceedings of Machine Learning Research, vol. 97, pp. 6687–6696. PMLR (2019)
17. Weinzierl, S., Zilker, S., Brunk, J., Revoredo, K., Matzner, M., Becker, J.: XNAP: making LSTM-based next activity predictions explainable by using LRP. In: Del Río Ortega, A., Leopold, H., Santoro, F.M. (eds.) BPM 2020. LNBIP, vol. 397, pp. 129–141. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-66498-5_10

SA4PM 2021: 2nd International Workshop on Streaming Analytics for Process Mining

2nd International Workshop on Streaming Analytics for Process Mining (SA4PM)

Streaming Process Mining is an emerging area in process mining that spans data mining (e.g. stream data mining; mining time series; evolving graph mining), process mining (e.g. process discovery; conformance checking; predictive analytics; efficient mining of big log data; online feature selection; online outlier detection; concept drift detection; online recommender systems for processes), scalable big data solutions for process mining and the general scope of online event mining. In addition to many other techniques that are all gaining interest and importance in industry and academia. The SA4PM workshop aims at promoting the use and the development of new techniques to support the analysis of streaming-based processes. We aim at bringing together practitioners and researchers from different communities, e.g., Process Mining, Stream Data Mining, Case Management, Business Process Management, Database Systems, and Information Systems who share an interest in online analysis and optimization of business processes and process-aware information systems with time, storage, or complexity restrictions. Additionally, SA4PM aims to attract research results on scalable algorithmic process mining solutions in general, given that the work addresses how such efficient solution would function under streaming settings. The workshop aims at discussing the current state of ongoing research and sharing practical experiences, exchanging ideas, and setting up future research directions.

The workshop started with an interesting invited talk by *Matthias Weidlich,* titled: "From Complex Event Recognition to Processes and Back – A Reflection on Existing Solutions and Open Challenges" where he shed light on scenarios and respective solutions for the integration of complex event recognition and process management. Then he pointed to open questions in the area.

This 2nd edition of the workshop attracted 7 international submissions, one of which was redirected to another workshop before the reviewing due to relevance. Each paper was reviewed by at least three members of the Program Committee. From these submissions, the top 3 were accepted as full papers for presentation at the workshop. The best rejected paper was invited for a work-in-progress talk without being included in the proceedings. All presenters got the chance to interact with the audience during a poster session. The workshop was held in a hybrid setting to enable online attendance to interact with the talks. The papers presented at the workshop provided a mix of novel research ideas and focused on online anomaly detection, online predictive monitoring, and streaming analysis of consumer behavior.

Anna Wimbauer et al. focus on online anomaly detection in online process monitoring when tracking local deviations over multiple process instances. The proposed method, called PErrCas, additionally visualizes correlations of deviation points. PErrCas provides knowledge about current cascades of deviations to give process analysts a starting point for rational root cause analysis if processes leave their in-control parameters. The method monitors deviations online and maintains cascades of varying timespans. As such, the approach avoids defining an observation window beforehand, which is a significant advantage due to its impracticability to predefine expected cascade properties in exploratory scenarios.

Next, *Suhwan Lee et al.* addressed the problem of continuous performance evaluation for business process outcome monitoring after defining it as a gap in the literature. Without such a continuous evaluation, users may be unaware of the performance of predictive models, resulting in inaccurate and misleading predictions. Their paper fills this gap by proposing a framework for evaluating online process outcome predictions, comprising two different evaluation methods. These methods are partly inspired by the literature on streaming classification with delayed labels and complement each other to provide a comprehensive evaluation of process monitoring techniques: one focuses on real-time performance evaluation, i.e., evaluating the performance of the most recent predictions, whereas the other focuses on a progress-based evaluation, i.e., evaluating the ability of a model to output correct predictions at different prefix lengths. The authors presented an evaluation involving three publicly available event logs, including a log characterized by concept drift.

Finally, *Yorick Spenrath, et al.* presented work on online prediction of aggregated retailer consumer behavior. Their observation is that the ability to make predictions on an individual level is useful, as it allows retailers to accurately perform targeted marketing. However, with the expected large number of consumers and their diverse behavior, making accurate predictions on an individual consumer level is difficult. Their approach presents a framework that focuses on this trade-off in an online setting. By making predictions on a larger number of consumers at a time, they improve the predictive accuracy but at the cost of usefulness, as one can say less about the individual consumers. The framework is developed in an online setting, where they update the prediction model and make new predictions over time. They show the existence of the trade-off in an experimental evaluation on a real-world dataset consisting of 39 weeks of transaction data.

We hope that the reader will find this selection of papers useful to keep track of the latest advances in the stream process mining area. We are looking forward to keeping bringing new advances in future editions of the SA4PM workshop.

November 2021

Organization

Workshop Chairs

Andrea Burattin	Technical University of Denmark, Denmark
Marwan Hassani	Eindhoven Univ. of Technology, The Netherlands
Sebastiaan van Zelst	Fraunhofer Inst. for Appl. Inf. Tech., Germany
Thomas Seidl	Ludwig-Maximilians-Univ. München, Germany

Program Committee

Agnes Koschmider	Kiel University, Germany
Ahmed Awad	University of Tartu, Estonia
Eric Verbeek	Eindhoven Univ. of Technology, The Netherlands
Felix Mannhardt	Norwegian Univ. of Science and Tech., Norway
Florian Richter	Ludwig-Maximilians-Univ., München, Germany
Francesco Folino	ICAR -CNR, Italy
Frederic Stahl	German Research Center for AI (DFKI), Germany
Jochen De Weerdt	KU Leuven, Belgium
Marco Comuzzi	UNIST, Korea
Matthias Weidlich	Humboldt-Universität zu Berlin, Germany

Online Prediction of Aggregated Retailer Consumer Behaviour

Yorick Spenrath(✉), Marwan Hassani, and Boudewijn F. van Dongen

Eindhoven University of Technology, Eindhoven, The Netherlands
{y.spenrath,m.hassani,b.f.v.dongen}@tue.nl

Abstract. Predicting the behaviour of consumers provides valuable information for retailers, such as the expected spend of a consumer or the total turnover of the retailer. The ability to make predictions on an individual level is useful, as it allows retailers to accurately perform targeted marketing. However, with the expected large number of consumers and their diverse behaviour, making accurate predictions on an individual consumer level is difficult. In this paper we present a framework that focuses on this trade-off in an online setting. By making predictions on a larger number of consumers at a time, we improve the predictive accuracy but at the cost of usefulness, as we can say less about the individual consumers. The framework is developed in an online setting, where we update the prediction model and make new predictions over time. We show the existence of the trade-off in an experimental evaluation on a real-world dataset consisting of 39 weeks of transaction data.

Keywords: Consumer Behaviour · Stream Analysis · Clustering

1 Introduction

Knowing the future behaviour of consumers is important to help retailers plan ahead [3]. One way to do so is by predicting how consumers will behave on an individual level. Knowing which consumers are expected to increase or decrease

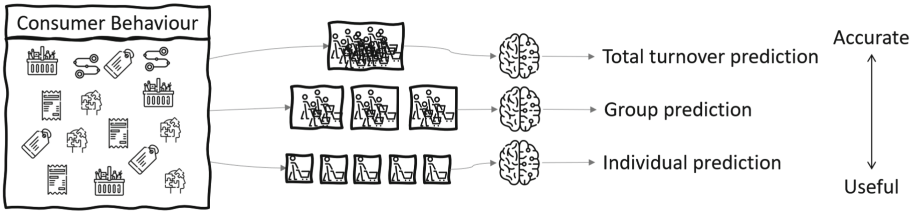


Fig. 1. Overview of the problem. Making predictions for individual consumers is more useful, but less accurate. Making predictions for all consumers (i.e. the entire retailer) is easier, but not useful on individual consumers. This paper balances the two by making predictions using groups of consumers.

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 211–223, 2022.
https://doi.org/10.1007/978-3-030-98581-3_16

their spending allows retailers to apply more targeted marketing strategies. Unfortunately, the human nature of consumers makes it difficult to accurately predict on an individual level. Two consumers can behave similar for some time, but then quite different in the next week. Another way to make predictions is by taking all consumers together, for example by predicting the turnover of the next week based on the turnover of the past weeks. This improves the accuracy of the predictions as we effectively remove outliers from individual consumers, but it also reduces the information we get about the individuals. This trade-off is schematically presented in Fig. 1.

In this paper we aim to strike a balance between accuracy and usefulness. Instead of making predictions on individual consumers we make predictions on *groups* of consumers. The advantage is an increase in accuracy with respect to making predictions for individual consumers as we remove the effect of outliers on the prediction. At the same time, we increase the usefulness of the prediction with respect to the prediction on all consumers together. This is because the predictions are on a limited number of consumers at a time. We apply our framework in a streaming setting, at regular intervals we discover clusters of consumers to update the prediction model. We as such make the following contributions: 1) we propose a framework to overcome the loss in prediction accuracy for diverse consumers by making the predictions on carefully selected clusters of consumers in a streaming environment, 2) we show its effectiveness on a real-world dataset from the supermarket domain and 3) we show that over time, making predictions on clusters does not decrease the accuracy of a downstream prediction task, in contrast to making predictions on individual consumers.

The rest of this paper is organized as follows: we first present our framework in Sect. 2 and evaluate it in Sect. 3. We then discuss how this paper relates to existing work in Sect. 4. Finally, we conclude the paper in Sect. 5.

2 Framework

In this section we start by giving an overview of our framework, we present the details and formalization of each step in the subsections. The data used can be considered analogous to concepts in process mining: events are represented by transactions, cases by consumers, and timestamps by purchase dates. Multiple events involving the same consumer constitute to (part of) the consumer journey.

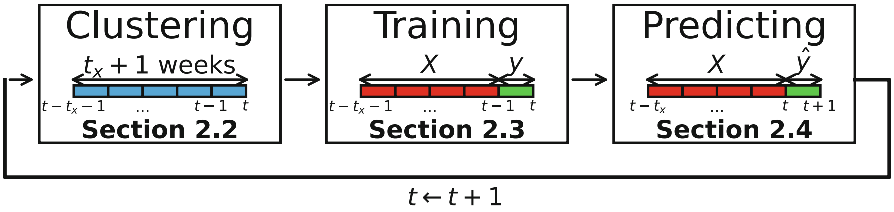


Fig. 2. Overview of the framework

These terms are interchangeable. For the description of the general framework we stick to ‘events’ and ‘consumers’, but use ‘transactions’ in the application details of this paper.

Like most other real-life applications, these journeys can vary wildly between different consumers. The goal is to make a prediction about one or more future events of a consumer. For our application this is the total spend over a given period of time, the sum of the spend in the separate events.

The framework starts with this collection of events containing information on a detailed level. This first step is to abstract from these single events. We do so by aggregating events per week and per consumer. This results in a vector of descriptive values that describes all visits of a consumer in a single week. We refer to this as *features*. The details of this are discussed in Sect. 2.1.

The framework consists of a loop of roughly three steps as presented in Algorithm 1 and schematically shown in Fig. 2. At time t, we use the past $\tau \in \mathbb{N}_+$ weeks of data, $[t-\tau, t)$, to make a prediction about the next week of data, $[t, t+1)$. We do not do so on an individual consumer basis, but on a cluster of consumers. In other words, we learn the behaviour of a *group of consumers* and make a prediction on their behaviour *as a group*. To this end we first cluster the consumers based on the their features from $[t - \tau - 1, t)$ into $k \in \mathbb{N}_+$ separate clusters. The details of this clustering are discussed in Sect. 2.2. Next, we construct a training dataset to train a Recurrent Neural Network (RNN) regression model. Each datapoint in this dataset represents one cluster. The predictor space consists of the features in each week from $[t - \tau - 1, t - 1)$, averaged over the consumers in the cluster. The target value is the turnover in $[t - 1, t)$, averaged over all consumers in the cluster. The details of this training and the RNN architecture are described in Sect. 2.3. Finally, we make a prediction for every cluster using

Algorithm 1: Overview of the framework

```
input  : Stream of events, in batches of one week at time $t$
output: Clusters $g_j^t$ of consumers and predictions $\hat{y}_j^t$ for the average turnover
        per consumer in $g_j^t$ after every week $t$
while True do
    Cluster consumers based on $[t - \tau - 1, t)$
        Extract descriptive features for consumers ▷ Section 2.1
        Create clusters of consumers ▷ Section 2.2
    Update LSTM
        One sequence per cluster over $[t - \tau - 1, t - 1)$ as predictor
        Turnover per cluster for $[t - 1, t)$ as target
        ▷ Section 2.3
    Make predictions
        One Sequence per cluster over $[t - \tau, t)$ as predictor
        Predict turnover per cluster for $[t, t + 1)$
        ▷ Section 2.4
    $t \leftarrow t + 1$
```

Table 1. Values derived from a single transaction.

Index	Value	Description
v_1	`Freshness`	Fraction of perishable items
v_2	`Item value`	Average value of the items bought
v_3	`Product density`	Average frequency of each product
v_4	`Total value`	Total price paid for the transaction
v_5	`Total item count`	Total number of items purchased

data $[t-\tau, t)$, predicting the average consumer spend of a cluster for $[t, t+1)$. The details of the prediction part are discussed in Sect. 2.4.

2.1 Features

Our framework starts with a set of events $\mathcal{L}$. Each such an event e has a timestamp $e.time$, a unique consumer identifier $e.cid \in \mathcal{C}$, a label $e.label$ in some activity space $\mathcal{A}$, and additional values $e.v_1$ through $e.v_m$ that contain further information on the event. The first step in our framework consists of summarizing multiple of these events over a period of time. For a time period $[t_1, t_2)$ and a consumer identified by c, we combine all events e that satisfy the predicate $t_1 \leq e.time < t_2 \wedge e.cid = c$, we indicate this set of events as $\mathcal{L}'$. Over these events we compute the fraction of each label $a \in \mathcal{A}$ as $freq(a) = |\{e \in \mathcal{L}' | e.label = a\}| / |\mathcal{L}'|$.

For the additional values $e.v_1$ through $e.v_m$ we define functions h_1 through h_m, with $h_i : \mathcal{P}(\mathcal{L}) \rightarrow \mathbb{R}$, which aggregates the values $e.v_i$ in $\mathcal{L}'$. We also record the number of events $|\mathcal{L}'|$. In total, we therefore have $|\mathcal{A}| + m + 1$ descriptions that together summarize the events in $\mathcal{L}'$. We refer to these as *features* in the remainder of the paper, and indicate them as $F = f_1, f_2, \ldots, f_{|F|}$, where $|F| = |\mathcal{A}| + m + 1$.

For the purpose of this paper, each event is a transaction made by a consumer at a retailer. The consumer identifier $e.cid$ is shared between transactions of the same consumer, which is known because consumers hold loyalty cards that uniquely identify them at each purchase. The timestamp $e.time$ is the date (and time) of the transaction. The label $e.label$ provides a description of the contents of the purchase. This labelling is based on an extension from earlier work [12]. It roughly consists of learning *eight* separate clusters over a large collection of transactions, based on the categories and quantities of the products in a transaction. Finally, the values $e.v_1$ through $e.v_5$ provide additional aggregate information on the transaction, as indicated by Table 1.

We therefore have $|\mathcal{A}| = 8$ (labels), $m = 5$ (values in Table 1), and as such a total of $|F| = 14$ features that describe the events over a period of time. With the help of domain knowledge[1], we define h_1 through h_3 as the mean, and h_4 and h_5

[1] The authors gratefully thank the company BrandLoyalty for making their data available for this project and their useful feedback on the framework.

as the sum of the values of the individual events. The latter two, aggregating the `Total value` and `Total item count`, are summed instead of averaged, as this helps distinguishing consumers with a large total spend from consumers with a smaller total spend.

The way in which our framework is constructed allows it to be generalized for the use in certain business processes as well. The event logs of such processes consist of labelled events that belong to a case with some case identifier, and possibly contain additional values per event [1]. This more generic application is beyond the scope of this paper.

2.2 Clustering

In the clustering step, we group consumers with similar behaviour over $[t-\tau-1, t)$, conform the left box of Fig. 2. We apply the aggregation of Sect. 2.1 to each separate week $[t-\tau-1, t-\tau), [t-\tau, t-\tau+1), \ldots, [t-1, t)$. As such, each consumer is described by $|F|$ features at $\tau+1$ points in time, over the period $[t-\tau-1, t)$. While we learn how these values evolve over time during training, we still want consumers with similar evolution to be grouped together. For each feature f_i we extract a linear fit over the values of f_i in those weeks as $f_i(t') = a_i \cdot t' + b_i$ with residuals r_i. Consumers with similar a_i, b_i, and r_i will have a similar average value for f (b_i), a similar increase/decrease (a_i), and a similar fitness to a linear trend (r_i). We compute these a_i, b_i, and r_i for each feature f_i, and cluster based on the resulting $3 \cdot |F|$ coefficients.

As consumers can have different points in time where they first visit the store, some may not have started their visits in or before the first week of the clustering period. We exclude those consumers who have their first purchase after $t-\tau$ in the clustering step, i.e. they are not assigned a cluster and clusters are not based on these consumers.

For the clustering we apply a Lloyd's algorithm [10] with the Euclidean distance, to find a given number of k clusters. In our studies, we evaluate a dataset of tens of thousands of consumers at every clustering step. As such, we apply an approximation to the Euclidean distance. Instead of taking the real space for each dimension, we divide each dimension in a discrete number of bins. This effectively reduces the number of actual datapoints, since some consumers may be in the same bin in every dimension. This also allows a more efficient calculation of the Euclidean distance[2]. After the clustering step at time t we have a clustering $G^t = \{g_1^t, g_2^t, \ldots, g_k^t\}$, with $g_i^t \subseteq \mathcal{C}$ and $g_j^t \cap g_i^t = \emptyset$ for $i \neq j$.

2.3 Training

In the training step we extract features and ground truth from the same time frame we use in the clustering. We use the period $[t-\tau-1, t-1)$ for the predictor values, and $[t-1, t)$ for the value of the total turnover that is to be

[2] More details on this approximation can be found at github.com/YorickSpenrath/ICPM2021/blob/main/BitBooster.pdf.

predicted, depicted in the centre box of Fig. 2. For the purpose of our framework, we require any model that uses these sequences of data to make predictions. For the scope of this paper we use a long-short term memory RNN (LSTM) as it learns from sequential data and has the advantage of allowing incremental training. Each training point is a matrix where each row is one element of the sequence, and each column is one feature. More specifically, for a cluster $g_i^t \in G^t$ we compute a matrix M_i^t such that $M_i^t[a, b]$ is the value of feature f_b over $[t - \tau - 2 + a, t - \tau - 1 + a)$, averaged over all consumers in g_i^t.

For the LSTM architecture we use the one defined by [14], staying as close to that work as possible, only changing the input and output layers to match our input and output representations. In an online training setting, we train the model from scratch in the first time step, and update it at every next time step.

2.4 Predicting

During the predicting we construct the predictor values in the same way as discussed in Sect. 2.3, using the clusters found in Sect. 2.2. In other words, during a time step we use the same clusters to construct training and testing points. We use $[t-\tau, t)$ to construct the sequence, i.e. we shift predictor period by 1 week, as depicted in the right box of Fig. 2. We use these predictor sequences to predict the average turnover of a consumer in the cluster over the week $[t, t+1)$, indicated as $\hat{y}_j^t$ for cluster g_j^t. This results in k predictions, one for each cluster. From this we can compute the total expected turnover by summing the products of the cluster sizes and cluster predictions. We define this value as $\hat{T} = \sum_{j=1}^{k} |g_j^t| \cdot \hat{y}_j^t$.

3 Experimental Evaluation

In this section we discuss the experimental evaluation of the method described in Sect. 2. For the experiment, we use data from a real-life retailer. Because of privacy restrictions, we cannot disclose all details. The transaction data comes from 39 weeks and contains over 160000 consumers with at least one purchase in that time. The number of visits per consumer varies between 1 and 312 with a mean of 55.0 and a standard deviation of 21.8. We discuss the experimental setup in Sect. 3.1 and then present and discuss the results in Sect. 3.2.

3.1 Experimental Setup

For the experiments we vary the two parameters of our framework: the sequence length τ and the number of clusters k. For τ we take $2, 4, 6, 8, 10$. For k we take values $250, 500, 750, 1000$ and 2000η for $\eta = 1 \ldots 5$. We also add two special cases. The first has $k = 1$: all consumers belong to the same cluster. The second does not use clusters, i.e. every consumer has its own cluster. This is indicated by $k = |\mathcal{C}|$. These latter two can be regarded as competitors: existing solutions that do not use clustering to improve predictions. In total, we conduct $5 \cdot 11 = 55$ experiments.

Each prediction estimates the average turnover per consumer in each cluster. The ground truth of this is the actual average turnover per consumer in the cluster. Based on this, we define three metrics to asses the quality of each parameter combination at each time step. The first is the root mean square error (**RMSE**) on the cluster predictions. This is a measure of the prediction accuracy. We further identify the 10% of the consumers that have the largest decrease in their turnover with respect to the previous week. Formally, we label each consumer with `True` if their decrease in turnover is among the 10% highest of all consumers (top decile), and `False` otherwise. We can do this using the actual turnover (as true label) and using the predicted turnover of the cluster they belong to (as predicted label). Using this classification, we then compute the $\mathbf{F_1}$ score at each time step as a second quality metric. Finally, we compute the predicted total turnover using $\hat{T}$ from Sect. 2.4 and compare this with the actual total turnover. We compute the absolute percentage error (**APE**), $100\% \cdot |T - \hat{T}|/T$, as a third metric. The latter two metrics are measures of usefulness.

Our aim is to answer the following questions. 1) How does τ influence the **RMSE** (prediction accuracy)? 2) How does k influence the **RMSE** (prediction accuracy)? 3) How does k influence the $\mathbf{F_1}$ on the top decile (usefulness)? 4) How does k influence the **APE** in $\hat{T}$ (usefulness)? 5) How does k influence the $\mathbf{F_1}$ *over time* (usefulness)? 6) What are the considerations for a 'good' value for k (balance)? For the first four questions we average all of the above metrics over time; starting at $t = 11$. This is because if $\tau = 10$, the first prediction made is at $t = 11$. In this way, we average the same number prediction metrics for every experiment. The implementation of the framework is open-source and can be found at www.github.com/YorickSpenrath/ICPM2021.

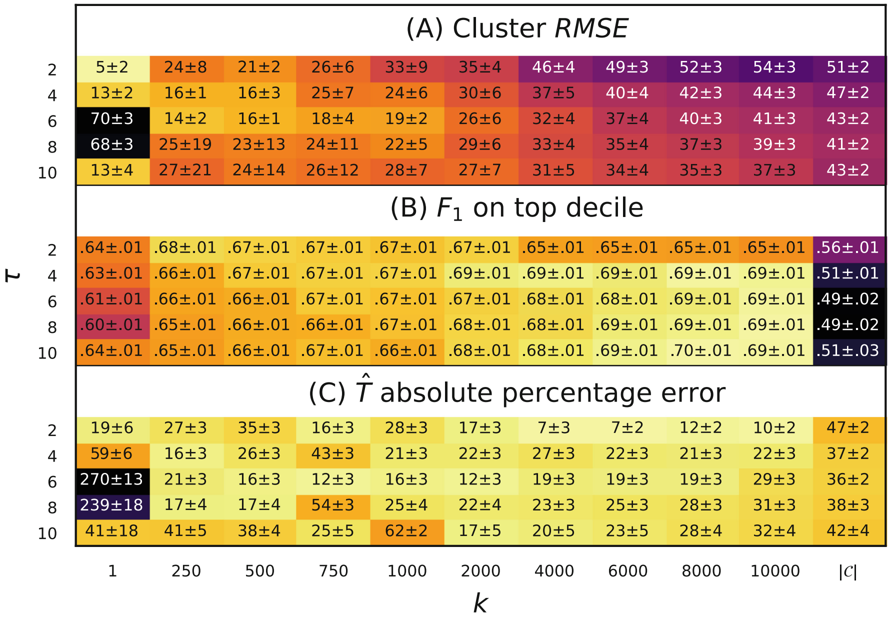
3.2 Results

In this section we present and discuss the experimental results. Each combination of k and τ delivers one value for **RMSE**, $\mathbf{F_1}$ and **APE**, averaged over time. The results are presented in Fig. 3. Each row contains the results for one value of τ, increasing k from left to right, each column contains the results for one value of k, increasing τ from top to bottom.

The Effect of $\boldsymbol{\tau}$ on RMSE (Prediction Accuracy). We first analyze the effects of τ on the **RMSE** in Fig. 3A as this will be relevant in the discussions on k. We distinguish between values $k \leq 1000$ and $k > 1000$.

$\mathbf{k > 1000}$ For larger k, all experiments show a clear decrease of **RMSE** with increasing τ. This is expected, as a higher τ means that the data sequences for each cluster are longer and the LSTM can learn from a longer period of time, which benefits its performance [18].

$\mathbf{k \leq 1000}$ For smaller k, the **RMSE** is not as consistently decreasing with increasing τ. The reason for this is that for a lower value of k we have fewer clusters of consumers and hence fewer training points. This makes the model possibly less stable, as it has less data to improve its performance. As a result, some models may fail to perform as expected. This means that longer sequences (higher τ) may result in poorer predictions than shorter sequences (lower τ).



(A) Cluster *RMSE*

τ	1	250	500	750	1000	2000	4000	6000	8000	10000	$\lvert C \rvert$
2	5±2	24±8	21±2	26±6	33±9	35±4	46±4	49±3	52±3	54±3	51±2
4	13±2	16±1	16±3	25±7	24±6	30±6	37±5	40±4	42±3	44±3	47±2
6	70±3	14±2	16±1	18±4	19±2	26±6	32±4	37±4	40±3	41±3	43±2
8	68±3	25±19	23±13	24±11	22±5	29±6	33±4	35±4	37±3	39±3	41±2
10	13±4	27±21	24±14	26±12	28±7	27±7	31±5	34±4	35±3	37±3	43±2

(B) F_1 on top decile

τ	1	250	500	750	1000	2000	4000	6000	8000	10000	$\lvert C \rvert$
2	.64±.01	.68±.01	.67±.01	.67±.01	.67±.01	.67±.01	.65±.01	.65±.01	.65±.01	.65±.01	.56±.01
4	.63±.01	.66±.01	.67±.01	.67±.01	.67±.01	.69±.01	.69±.01	.69±.01	.69±.01	.69±.01	.51±.01
6	.61±.01	.66±.01	.66±.01	.67±.01	.67±.01	.67±.01	.68±.01	.68±.01	.69±.01	.69±.01	.49±.02
8	.60±.01	.65±.01	.66±.01	.66±.01	.67±.01	.68±.01	.68±.01	.69±.01	.69±.01	.69±.01	.49±.02
10	.64±.01	.65±.01	.66±.01	.67±.01	.66±.01	.68±.01	.68±.01	.69±.01	.70±.01	.69±.01	.51±.03

(C) $\hat{T}$ absolute percentage error

τ	1	250	500	750	1000	2000	4000	6000	8000	10000	$\lvert C \rvert$
2	19±6	27±3	35±3	16±3	28±3	17±3	7±3	7±2	12±2	10±2	47±2
4	59±6	16±3	26±3	43±3	21±3	22±3	27±3	22±3	21±3	22±3	37±2
6	270±13	21±3	16±3	12±3	16±3	12±3	19±3	19±3	19±3	29±3	36±2
8	239±18	17±4	17±4	54±3	25±4	22±4	23±3	25±3	28±3	31±3	38±3
10	41±18	41±5	38±4	25±5	62±2	17±5	20±5	23±5	28±4	32±4	42±4

Fig. 3. Results for the experiments. Lighter colours indicate a better value, the colour scale is with respect to the values in each separate table. The **RMSE** is in monetary units, the **APE** are percentages.

The Effect of *k* on RMSE (Prediction Accuracy). We next analyze the influence of k (left to right) on the **RMSE**. With the exception of $k = 1$, from Fig. 3A we clearly see that the **RMSE** increases (worse predictions) with k. This is to be expected, as an increase in the number of clusters means that each cluster is smaller in size. As a result of this, the values in the clusters that are used to construct the sequences fed to the LSTM are based on fewer consumers. This increases the effect of outliers on the mean feature value and hence decreases the quality of the sequences on which the LSTM is trained. With the exception of $\tau \in [6, 8]$, $k = 1$ follows a similar trend, outperforming all other values of k. The exception for $\tau \in [6, 8]$ is likely caused by the above-mentioned instability: the model is only updated with a single value every training step. This may still result in a decent model (as evident from other τ values) though there is no guarantee. While the **RMSE** for the other τ values is consistently lower than those for other k, we do note that $k = 1$ makes a single predictions for all consumers together. This means that little to nothing can be said about the individual consumers making it less useful than many clusters, where the predictions are on fewer consumers at a time.

The Effect of *k* on $\mathbf{F_1}$ (Usefulness). Figure 3B clearly shows the inverse effect of Fig. 3A in terms of k. An increase in k shows a clear increase in the $\mathbf{F_1}$, making the resulting models more useful. There does seem to be a limit to this though, once we make predictions on individual consumers ($k = |\mathcal{C}|$), the $\mathbf{F_1}$ decreases again. We expect the cause of this to be the further increase in **RMSE** for this value, though future research should look into this.

The Effect of *k* on APE (Usefulness). As depicted in Fig. 3C, the relation between k and **APE** is less obvious than for the other two metrics. For lower values of τ, a higher value of k is preferred. For mid range values of τ, lower values of k perform better. For higher values of τ, the mid range of k shows a better performance. It is difficult to find the exact reason for this. One explanation is that each cluster prediction can either be too high (over-predicting) or too low (under-predicting). Summing all these predictions to compute the total turnover then effectively self-corrects these errors. There are then two factors affecting the total turnover error: the **RMSE** of the individual predictions and the self-correction effect. As discussed, the former increases with k. It is reasonable that the latter effect is more present for higher k. The question is which of these effects is stronger. As discussed before, the **RMSE** decreases with τ. As such, for lower τ, the stronger self-correction effect for higher k is more important to get a decent total turnover prediction. For higher values of τ, the reduced **RMSE** appears to allow lower values of k (with $k > 1000$) to be preferred. For the mid-range of τ values, it is not exactly clear why even lower k values are better, though this can be caused by the same model instability effect discussed before.

The Effect of *k* on $\mathbf{F_1}$ over Time (Usefulness). The results above have shown the effect of k on prediction accuracy and usefulness as average over all time steps. We now discuss some of the effects that can be viewed as the model progresses over time, presented in Fig. 4. The numbers in the bottom row of Fig. 3B are the averages of these plots. From the figure we clearly see the importance of making predictions in groups of consumers. The $\mathbf{F_1}$ for making individual predictions ($k = |\mathcal{C}|$) quickly diminishes over time, stabilizing up to 0.25 points lower than the other experiments. Next to this, we see the effect of external events at three points in time, indicated by vertical dotted lines. These external events are likely to cause a sudden change in the consumer behaviour, decreasing the predictive accuracy in the time around them. This results in a sudden drop in $\mathbf{F_1}$, especially for the first and third event.

The Optimal Value of *k*. Based on the above results there are three considerations for the optimal value of k: Model stability (lower k means fewer training points), Cluster stability (lower k means better averaged clusters), and Cluster detail (higher k makes predictions closer to individual consumers).

4 Related Work

One class of supervised learning is called ‘bucketing’. In bucketing, datapoints from a training set are first clustered using some clustering method, and a separate machine learning model is trained on each cluster. This approach is also extensively used in predictive process mining. Examples of such works are [5] (offline) and [6] (online). Our approach is different in the sense that we do not train one model per cluster, but train and update a single model using datapoints that are each extracted from a single cluster. While having a different target, the work in [4] applies clustering for the same reason as we do. The aim of that work is to discover process models that describe the sequences in an event log. A difficulty in discovering such process models is the variability in sequences. As a solution, the authors iteratively split the collection of sequences to create smaller event logs to create better models. The splitting is based on clustering to combine comparable sequences, much like our approach.

In Sect. 2.1 we described how we summarize a sequence of events over a period of time into a feature vector. This process is referred to as *encoding* or *embedding*. In the process mining field, different techniques of encoding exist. The most frequently used method to limit the number of events considered (prefix) to create evenly-sized vectors. These vectors then either list only the labels ($e.label$), or also each of the attributes ($e.v_1$ - $e.v_m$). Examples of the use of this encoding are [5,6,9]. The disadvantage of this approach in our use case is that the number of events in a given time frame is highly relevant, limiting to a fixed-number of first events would lose this information. The frequency-based encoding we apply is also used in for example [11]. [9] further uses Hidden Markov Models (HMMs): a value of likeliness that a sequence belongs to the target class based on initially learned HMMs is added to the feature space. A more dedicated approach is to find relevant subsequences and count their frequencies, such as in [2,4]. While this can be highly relevant as an addition to our current encoding, it is computationally expensive to find which subsequences are relevant.

In terms of predictive process mining, this paper is part of a class of outcome prediction solutions. [14] adopts LSTMs to predict the remainder (suffix) of a case by repeated next activity predictions. The same target is predicted in [15] but then with the use of deep adversarial models. In [9], the authors predict whether an active case will be compliant or not according to the business process owner, leveraging complex encoding case as explained above. The same

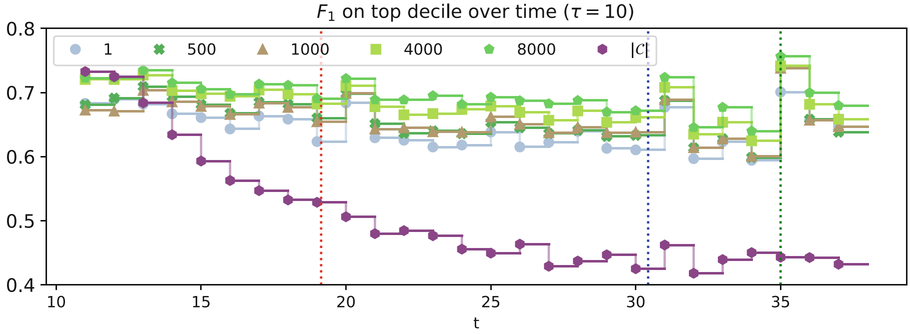


Fig. 4. Progression of $\mathbf{F_1}$ over time for $\tau = 10$. The vertical lines indicate significant external influences that can influence consumer behaviour.

prediction task is executed by [5], which makes use of the bucketing described above. Using techniques from text mining, [16] aims to early signal whether a case will have a outcome that requires intervention using unstructured textual information from events. A more detailed summary of recent outcome-oriented tasks can be found in [17]. Next to this, literature contains specific to consumer behaviour prediction. Examples of these include the next interaction [8], losing a consumer (churning) [3,7,13], and life-time value [3]. Of these, [8] also uses a process mining oriented approach, and [13] also uses Neural Networks for their predictions. The work of [3] further suggests the use of automatically learned features over handcrafted ones for the prediction.

5 Conclusion and Future Work

In this paper we proposed a framework to make predictions about future events of consumer behaviour, aiming to strike a balance between accuracy and usefulness. Larger clusters lead to better predictions but say less about the individual consumers, and vice-verse for smaller clusters. Apart from this, a lower number of clusters likely causes the prediction model to be less stable as fewer training points are available. We also demonstrated the benefit of clustering consumers over time. When making predictions on an individual consumers, the usefulness ($\mathbf{F_1}$) rapidly decreases over time, this effect is not seen when consumers are grouped together for the prediction.

For future research, several directions can be identified. The most important one is how the size of the dataset affects the considerations for the optimal number of clusters. Another direction is to replace the linear fit clustering method. The framework operates on events that belong to consumers, and as such existing clustering methods from the process mining field, such as [2,9], are alternatives to this. Finally, at each time step the clusters are recomputed. An extension lies in incorporating past information on clusters, such that longer-term similarities in consumer behaviour can also be considered.

References

1. Van der Aalst, W.: Process Mining, 2nd edn. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
2. Bose, J.C.R.P.: Process mining in the large. Ph.D. thesis, Eindhoven University of Technology (2012)
3. Chamberlain, B.P., Liu, B., Pagliari, R., Deisenroth, M.P.: Customer lifetime value prediction using embeddings. In: KDD 2017 ADS 2017 (2017)
4. De Medeiros, A.K.A., et al.: Process mining based on clustering: a quest for precision. In: ter Hofstede, A., Benatallah, B., Paik, H.-Y. (eds.) BPM 2007. LNCS, vol. 4928, pp. 17–29. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78238-4_4
5. Di Francescomarino, C., Dumas, M., Maggi, F.M., Teinemaa, I.: Clustering-based predictive process monitoring. IEEE Trans. Serv. Comput. **12**(6), 896–909 (2019)

6. Di Francescomarino, C., Ghidini, C., Maggi, F.M., Rizzi, W., Persia, C.D.: Incremental predictive process monitoring: how to deal with the variability of real environments (2018). www.arxiv.org/abs/1804.03967
7. Günesen, S.N., Şen, N., Yıldırım, N., Kaya, T.: Customer churn prediction in FMCG sector using machine learning applications. In: Mercier-Laurent, E., Kayalica, M.Ö., Owoc, M.L. (eds.) AI4KM 2021. IAICT, vol. 614, pp. 82–103. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-80847-1_6
8. Hassani, M., Habets, S.: Predicting next touch point in a customer journey: a use case in telecommunication. In: Proceedings ECMS, vol. 35, no. 1, pp. 48–54 (2021)
9. Leontjeva, A., Conforti, R., Di Francescomarino, C., Dumas, M., Maggi, F.M.: Complex symbolic sequence encodings for predictive monitoring of business processes. In: Motahari-Nezhad, H.R., Recker, J., Weidlich, M. (eds.) BPM 2015. LNCS, vol. 9253, pp. 297–313. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23063-4_21
10. Lloyd, S.P.: Least squares quantization in PCM. IEEE Trans. Inf. Theory **28**(2), 129–137 (1982)
11. Song, M., Günther, C.W., Van der Aalst, W.M.P.: Trace clustering in process mining. In: Ardagna, D., Mecella, M., Yang, J. (eds.) BPM 2008. LNBIP, vol. 17, pp. 109–120. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00328-8_11
12. Spenrath, Y., Hassani, M., Van Dongen, B., Tariq, H.: Why did my consumer shop? Learning an efficient distance metric for retailer transaction data. In: Dong, Y., Ifrim, G., Mladenić, D., Saunders, C., Van Hoecke, S. (eds.) ECML PKDD 2020. LNCS (LNAI), vol. 12461, pp. 323–338. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-67670-4_20
13. Tariq, M.U., Babar, M., Poulin, M., Khattak, A.S.: Distributed model for customer churn prediction using convolutional neural network. J. Model. Manag. (2021). https://doi.org/10.1108/JM2-01-2021-0032
14. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive business process monitoring with LSTM neural networks. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 477–492. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_30
15. Taymouri, F., La Rosa, M., Erfani, S.M.: A deep adversarial model for suffix and remaining time prediction of event sequences. In: Proceedings of the SDM, pp. 522–530 (2021)
16. Teinemaa, I., Dumas, M., Maggi, F.M., Di Francescomarino, C.: Predictive business process monitoring with structured and unstructured data. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM 2016. LNCS, vol. 9850, pp. 401–417. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45348-4_23
17. Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M.: Outcome-oriented predictive process monitoring: review and benchmark. ACM Trans. Knowl. Discov. Data **13**(2), 1–57 (2019)
18. Wen, Y., Zhang, W., Luo, R., Wang, J.: Learning text representation using recurrent convolutional neural network with highway layers. In: Neu-IR Workshop Proceedings, SIGIR 2016 (2016)

PErrCas: Process Error Cascade Mining in Trace Streams

Anna Wimbauer(✉), Florian Richter, and Thomas Seidl

Ludwig-Maximilians-Universität München, Munich, Germany
a.wimbauer@campus.lmu.de, {richter,seidl}@dbs.ifi.lmu.de

Abstract. Efficient and quick detection of problems is an essential task in online process monitoring. Many anomaly detection approaches excel in finding local deviations. We propose a novel approach that tracks local deviations over multiple process instances and visualizes correlations of deviation points. PErrCas provides knowledge about current cascades of deviations to give process analysts a starting point for rational root-cause analysis if processes leave their in-control parameters. PErrCas monitors deviations online and maintains cascades of varying timespans. Hence, our approach avoids defining an observation window beforehand, which is a significant advantage due to its impracticability to predefine expected cascade properties in exploratory scenarios.

Keywords: Anomaly Detection · Cascades · Trace Streams

1 Introduction

Anomaly detection has multiple applications in process mining. The most prominent scenario is conformance checking, where misbehavior of process instances is measured against a reference process model by techniques like token replay or alignments. The identified anomalies represent structural non-compliances in comparison to previous or planned executions. Temporal deviations are another focus for process anomaly detection since detecting unexpected delays or speed-ups often provides a starting point for thorough investigations. Fraud, failures, or inefficient resource usage are only a few root causes for deviations.

While the research community has published a rich collection of techniques to detect various anomalies, most works focus explicitly on correlations within cases and neglect interferences between different cases. Whether it be customer journeys, production cycles, or sequences of administrative actions, cases are handled as independent process executions, and explanations for anomalies in a case are usually expected to be caused by previous events in the same case. However, cases share a resource pool containing staff, machinery, or infrastructure. Restricting a root cause analysis to singular cases might fail if another instance has caused an issue and subsequent cases are affected by its effects. We differ between local anomalies, isolated within a singular case, and global anomalies, which originate in a particular case of an event and spread through the process using common tie points between cases.

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 224–236, 2022.
https://doi.org/10.1007/978-3-030-98581-3_17

This work presents a novel online approach to identify process error cascades in a trace stream. Error cascades are typically not artificially implemented in processes. Since many processes contain a dynamic resource scheduling, e.g., the staff is assigned depending on current situations like workload or environmental influences, static cascade knowledge has limited value. Error cascades have two additional properties besides their various lifetimes, defined as the timespan between the actual event and the last moment that the cascade influences events.

Many cascades affect only structurally subsequent events according to the process. E.g., delayed transporters in logistic processes delay following transports, which might delay further transports waiting for the first segment. In specific processes, deviations may cause feedback in the process. Delays in production processes often cause previous and following actors to traverse into idle states. Depending on the process design, this allows preponing of cases in contrast to their scheduled execution. If the processes do not allow resources reallocation, previous actors also switch to a delay status.

The remaining important property of cascades is complexity. Typically, most cascades contain only a few correlated actions. Complex cascades with long correlation chains of affected actions are infrequent but provide valuable insights for later investigations. Large distances between root causes and detected deviations are typical scenarios where manual analysis fails to establish the causal connection.

2 Related Work

Correlations between different database objects have been extensively researched in the domain of sequential pattern mining [5]. Regarding sequential pattern mining on data streams, traditional SPM algorithms are required to overcome memory and performance restrictions and are therefore not always suitable to be applied on data streams directly. Marascu and Masseglia [9] propose an approximate algorithm called SMDS (Sequence Mining in Data Streams) primarily designed for Web usage data streams that can handle the complexity of streaming data. In their approach, user transactions are processed in batches. For each batch, the users are clustered based on their surfing behavior adding users to the most similar cluster or creating a new cluster. In [7,14] research on online sequential pattern mining is continued. However, this research direction focuses on totally ordered sequences. Event-based processes allow concurrent executions of events, and anomalies are propagated non-linearly due to the process complexity. Moreover, we consider if two anomalies happen close in time to declare a correlation, while temporal intervals are usually neglected in sequence mining.

In the field of spatio-temporal data mining deep learning methods are used to learn traffic flow correlations to predict future traffic flow [6,13]. Since those methods depend on the spatial features and processes mostly neglect spatial data while focusing on structural positions in the process, the approaches are not directly applicable for our use case. Even if event logs include spatial data, this information might not be relevant for the causal relationship between outliers.

In Liu et al. [8] the authors aim at finding causal interactions between traffic outliers by constructing outlier causality trees and running a frequent subtree mining algorithm on them. Toosinezhad et al. [12] applied these ideas for process mining. The authors are the first to solve a significant task, as the origins of process failures are not always found within the same process instance since the real world is interconnected. Their approach does not consider anomalies for each case individually but anomalies over various cases and their correlations. Hence, Toosinezhad et al. proposed a method to tackle this challenge and introduce a novel perspective of process anomaly detection.

As cases proceed in a process, their irregular behavior might disrupt the entire system causing further anomalies. Toosinezhad et al. divide the dataset into batches and construct one cascade graph per batch. The partitioning into specific intervals, like weeks, requires prior knowledge of certain cascade properties. Instead, we expand our cascades incrementally without batch restrictions. We create new cascades when incoming outlier events are not correlated to already identified cascades. Finally, we cluster the constructed cascades to give generalized cascade patterns, allowing quicker analysis by emphasizing the prominent structures.

The Performance Spectrum miner presented in [3] uses a descriptive analysis to reviel performance patterns. In [11] Senderovich et al. use both intra- and inter-case features to predict case properties. However, to the best of our knowledge, Toosinezhad et al. proposed the only work on detecting anomaly cascades in processes so far.

3 Preliminaries

The proposed method is applied to trace streams. A trace stream $S : \mathbb{N} \rightarrow \mathbb{N}$ is a mapping from natural numbers to the case identifier domain. Such a trace stream can be efficiently generated from an event stream, as already described in [10]. On case-level, each case contains finitely many events.

Definition 1 (Case-Level Event). *A case-level event e is a tupel e = $(c,$ $a,$ $t)$ containing a case identifier $\#_{case}(e) = c$, an activity label $\#_{activity}(e) = a$ and a timestamp $\#_{time}(e) = t$. The case-level event may also contain additional attributes.*

Regarding intervals between case-level events, we define segment-level events. These are then aggregated into cascades which are modelled as graphs and represent the causal dependencies on the process level.

Definition 2 (Segment-Level Event). *A segment-level event s is a tupel s = $(sn,$ $c,$ $st,$ $et)$ containing a segment-name $\#_{segment}(s) = sn$, a case identifier $\#_{case}(s) = c$, a start-time $\#_{start}(s) = st$ and an end-time $\#_{end}(s) = et$. Every segment-level event s is composed of two case-level events e_i and e_j, where $\#_{case}(e_i) = \#_{case}(e_j) = c$, $(\#_{activity}(e_i), \#_{activity}(e_j)) = sn$, $\#_{time}(e_i) = st$ and $\#_{time}(e_j) = et$. It must hold that $\#_{time}(e_i) < \#_{time}(e_j)$ and there is no e_k with $\#_{case}(e_k) = c$ such that $\#_{time}(e_i) < \#_{time}(e_k) < \#_{time}(e_j)$.*

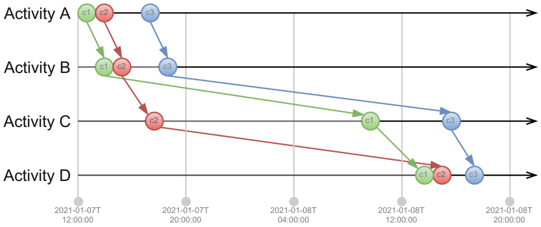


Fig. 1. Example process time line

Definition 3 (Error Cascade). *An error cascade is a directed graph $g = (V, E)$, where each node n in V represents a set of outliers $S = s_1, \cdots, s_k$ in one segment $\#_{segment}(s_1) = \cdots = \#_{segment}(s_k)$. There is an edge from node n_i to n_j, if outliers in n_j are correlated to preceding outliers in n_i.*

Each node has a *heat value* that gives information about the last time an outlier occurred in this segment. It is computed as an exponentially moving average to consider all past segment-level event outliers aggregated in this node. We declare a node as active if the time difference between the starting time of the current outlier and the heat value of the node is lower than a predefined *activity threshold* th_a. The activity threshold defines the time span in which we assume two outliers to be correlated. If the activity threshold is one day, an outlier can affect the process performance for one day. Henceforth, if the time difference between the heat value and a new outlier is greater than the activity threshold, a causal relationship between the outlier set of that node and the new outlier is impossible. We call a cascade active as long as at least one of its nodes is still active.

4 Online Cascade Mining

In this section, we define the three main steps of our method. Our approach operates on trace streams. We first scan for process segments that take an unusually long (or short) time for each incoming trace. We then check for each outlier if it is correlated to an already existing active cascade, in which case we add the outlier to the correlated cascade. If it is not correlated to an existing cascade, the outlier forms the start of a new cascade. These first two steps are performed on each trace consecutively. The last step is carried out in an offline phase once a set of cascades has accumulated. We cluster the cascades and compose all cascades in one cluster to a cascade pattern.

4.1 Outlier Segment-Level Events

For each incoming trace, we generate the segment-level events from consecutive case-level events and search for temporally deviating segment-level events. Figure 1 shows an example process with four activities A, B, C, and D. Cases

c_1, c_2 and c_3 arrive shortly after one another and traverse through the process at different paces. Every circle on the timeline symbolizes a case-level event. It means, e.g., that case c_1 underwent activity A at 12:35 on the seventh of January 2021. Since there are four successive activities, we have three segment-level events per case: *A:B*, *B:C* and *C:D*. All three cases transition from activity A to activity B fairly quickly, then c_1 gets delayed in segment B:C. This leads to further delays of case c_2 in segment C:D and case c_3 in segment B:C. We can already see that segment-level events *B:C* - c_1, *C:D* - c_2 and *B:C* - c_3 will be marked as outliers.

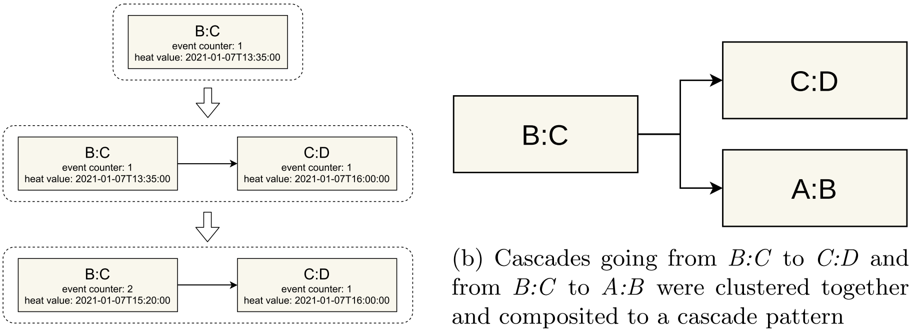
Formally we declare a segment-level event an outlier if its z-score $Z(s) = \frac{\Delta t - \mu_{segment}}{\sigma_{segment}}$ is higher than a certain outlier threshold th_o. With $\Delta t = \#_{end}(s) - \#_{start}(s)$ being the duration of the segment-level event. The mean $\mu_{segment}$, the variance $\sigma^2_{segment}$ and the number of events per segment $k_{segment}$ are stored for each segment and updated with every incoming segment-level event.

4.2 Error Cascade Construction

When a new outlier arrives, we check whether it correlates to any currently active cascades. In this case, it is "added" to this cascade. If an outlier is not correlated to an active cascade, a new cascade is started. Over time older cascades become inactive node by node, and new cascades are started and built up. If an outlier segment-level event s and a cascade fulfill one of the two following cases we assume that they are correlated.

1. Segment-level event s belongs to the same segment as a node n in the cascade and $\#_{start}(s) - \#_{heat}(n) < th_a$. The cascade already includes a set of outliers in the same segment that is still active, in a sense that the time difference between heat value and starting time of the outlier does not exceed the activity threshold. In this case, outlier s is added to node n by increasing the event counter by one and updating the heat value: $\#^{new}_{heat}(n) = \#_{start}(s) - [0.25 \cdot (\#_{start}(s) - \#^{old}_{heat}(n))]$
2. Segment-level event s and a node n in the cascade share a common activity and $\#_{start}(s) - \#_{heat}(n) < th_a$. Since outlier segment-level event s and the outliers of node n are close in time and overlap in their segments, we assume that the anomalous behaviour of s is correlated to the segment-level events aggregated in node n. In this case a new node n_{new} for segment $\#_{segment}(s)$ is appended to the cascade such that *event counter* $= 1$ and $\#_{heat}(n_{new}) = \#_{start}(s)$. An edge is added from n to n_{new} symbolizing the correlation between n and n_{new}.

If a segment-level event s is not correlated to a cascade, we assume that none of the preceding events are correlated to this outlier. As stated above, the new outlier event s then marks the start of a new cascade. We start a new cascade by generating a new cascade graph with one node. In the same way as a new node is added to an existing cascade, the first node of the new cascade has $\#_{segment}(s)$ as segment and *event counter* $= 1$ and $\#_{heat}(n) = \#_{start}(s)$. If the cascade still contains one node once it becomes inactive, we delete it and


B:C
event counter: 1
heat value: 2021-01-07T13:35:00
B:C
event counter: 1
heat value: 2021-01-07T13:35:00
C:D
event counter: 1
heat value: 2021-01-07T16:00:00
B:C
event counter: 2
heat value: 2021-01-07T15:20:00
C:D
event counter: 1
heat value: 2021-01-07T16:00:00
B:C
C:D
A:B

(a) Example cascade building process

(b) Cascades going from *B:C* to *C:D* and from *B:C* to *A:B* were clustered together and composited to a cascade pattern

Fig. 2. Example Cascade Mining

regard the corresponding outlier (or outliers) as standalone. Figure 2a describes the incremental cascade building process for our example. *B:C* is the first node of the new cascade, because outlier *B:C* - c_1 could not be added to an existing cascade. Next comes outlier *C:D* - c_2 which is correlated to node *B:C* because they overlap in activity *C* and are temporally close. A new node *C:D* with an edge from *B:C* to *C:D* is added to the cascade. The third outlier *B:C* - c_3 is correlated to both existing nodes. Since there is already an active node for segment *B:C* the outlier is added to this node by updating the event counter and heat value.

4.3 Cascade Patterns

In the first two steps, we process the traces and the outliers within these traces consecutively. Every time a specific time has passed, and a set of cascades could be collected within this period, the last step is carried out. We then cluster these cascades in an offline phase to search for patterns within the cascades, i.e., patterns of correlated segments. Alternatively, one of the various online clustering algorithms (see [15]) could be applied to every error cascade that is no longer active. This however is not in the scope of this paper.

We first cluster the cascade set by applying DBSCAN [4]. We chose the DBSCAN clustering algorithm [4], because it can find clusters of arbitrary shapes and can handle noise. The clustering provides a grouping into similar cascade graphs and filters out noisy or rare cascades simultaneously. To apply the algorithm, we define a distance measure within the cascade space. For the distance between two cascade graph we use the maximum common subgraph metric as presented in [1]. To get more representative clustering results, we assign an additional weight to every cascade. If a cascade weights 2, the clustering algorithm handles the cascade as if it was contained in the set twice. As weights, we choose the average number of segment-level event outliers that nodes in this cascade contain. Adding weights is necessary because there might be cascades that stay active for a long time. If correlated outliers come in at frequent intervals, we

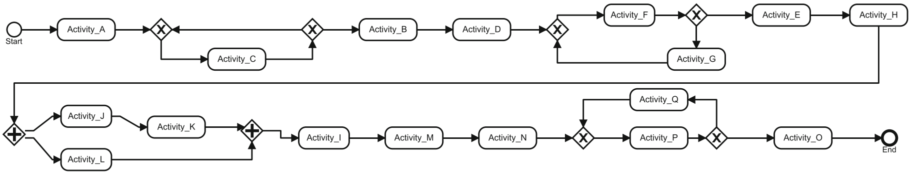


Fig. 3. Underlying process model for the synthetic data

always add them to the same cascade. This continuously prolongs the cascades activity, and no new cascades with the same cascade pattern are generated. Without adding any weights, DBSCAN would declare these cascade graphs as noise, even though they represent many segment-level event outliers.

Finally, we compose all cascade graphs within a cluster into one cascade pattern. Composing the cascades means we summarize all nodes and edges from the individual graphs in one graph, the cascade pattern. The cascade pattern provides a good overview of the various cascades in the respective cluster.

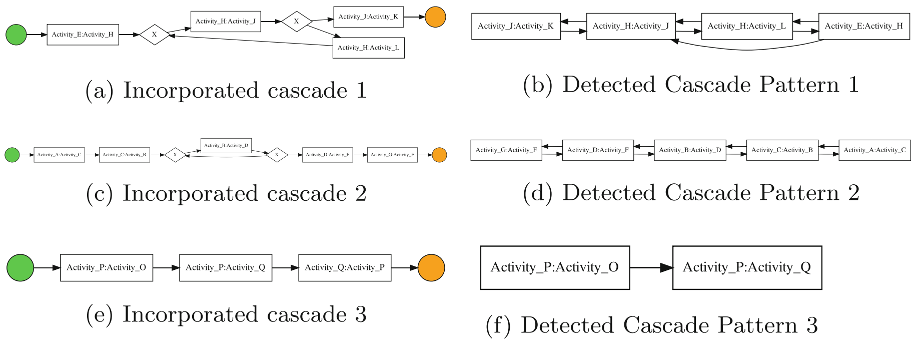
Clustering and composing the cascades aims at generating a relatively small, manageable and easy to interpret result set. Different cascade patterns represent distinct groups of outlier correlations. The compression is a significant advantage compared to [12], where the number of resulting frequent cascades tends to be very large, and there are often large groups of very similar frequent cascades.

Getting back to our example, let us assume that we retrieved a few more cascades from *B:C* to *C:D*. Additionally, cascades from *B:C* to *A:B* were detected. These cascades were grouped into the same cluster by the clustering algorithm, and we compose these cascades into the cascade pattern shown in Fig. 2b. This cascade pattern visualizes in an intuitive way that delays in segment *B:C* were correlated to delays in both segment *C:D* and *A:B*. The final cascade pattern then forms a good basis for possible process improvements.

5 Evaluation

5.1 Synthetic Data

In the following we present our results from testing our method on synthetic and real life data. We first tested our approach on synthetic data, as this way, we could verify the results we obtained from our method. For DBSCAN clustering we use the following parameters: $\epsilon = 0.4$ and $minPts$ is set to the 75%-quantile of the cascade weights, but at least 4. For the synthetic data we used the processes and logs generator PLG2 [2] to generate an eventlog, based on the process model shown in Fig. 3. We then spread all traces over one year and introduced noise by randomly delaying every event (normally distributed with $\mu = 30$, $\sigma^2 = 25$ minutes). Finally, we incorporated the three cascades shown in Fig. 4, by delaying events in the corresponding segments. The cascades occur 300, 50 and 12 times and have an approximate length of one day, one week and one month.



(a) Incorporated cascade 1

(b) Detected Cascade Pattern 1

(c) Incorporated cascade 2

(d) Detected Cascade Pattern 2

(e) Incorporated cascade 3

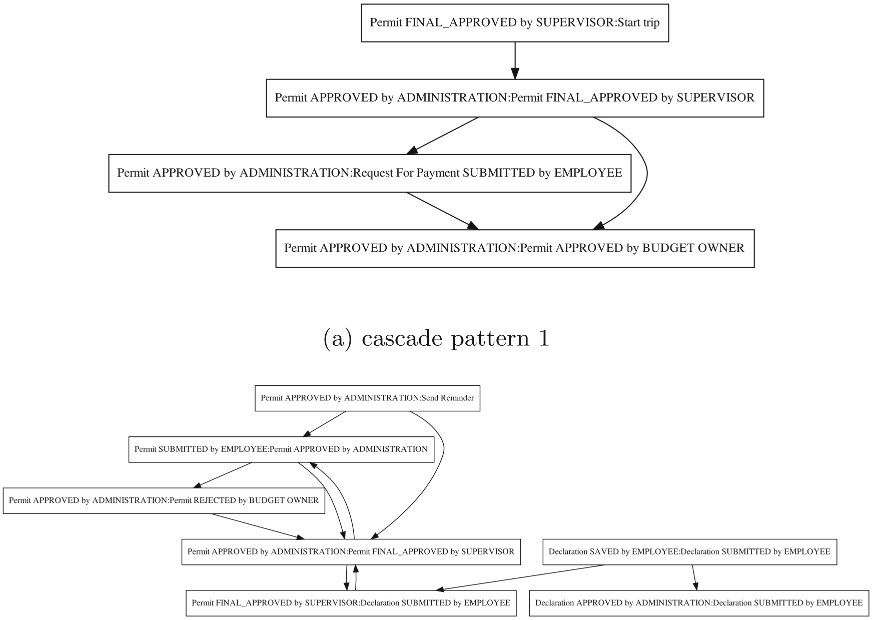
(f) Detected Cascade Pattern 3

Fig. 4. Induced and detected cascades in the synthetic log with $activity\_threshold = 1, outlier\_threshold = 5$ and $\epsilon = 0.4$

We tested our approach with different parameters, achieving the best results with an *activity threshold* of 1 day and an *outlier threshold* of 5. During the cascade detection phase 882 segment-level event outliers were detected and assigned to 167 cascades graphs. Out of these 167 graphs 121 were deleted before clustering because they contained only one node. In the end we received 46 cascade graphs, which were then grouped into 3 clusters (and some outliers) and composited to the 3 cascade patterns shown in Fig. 4. This complies with the number of cascades from the ground truth. Cascade 1 and 2 are nearly identical to the induced cascades and also have a maximum common subgraph (mcs) similarity of 1.00 with the ground truth cascade. Cascade 3 is missing its last segment node, which leads to a mcs similarity of 0.67.

To test our approach on datasets with different quality we increased noise in our dataset. As described earlier we first generated an event log without any noise (using PLG2) and induced the three cascades in a second step. To create synthetic logs with increasing noise, we introduced noise to the control flow of the initial event log using PLG2. To this end, we chose increasing parameters (0 to 40 promille) for the trace missing head, trace missing tail, trace missing episode, perturbed event order probability. We generated five logs for each noise parameter and averaged the results over these five logs, since the results varied due to randomness in the event log creation process.

The tested parameters and corresponding results are shown in Fig. 7. The F1-score was calculated by comparing each detected cascade pattern with the ground truth cascade it was most similar to. F1 nodes only considers correctly/wrong assigned nodes, whereas the total F1-score considers nodes and edges. The overall recall and the F1-score for nodes are significantly higher than the overall F1-score, which is mainly due to additional edges in the detected cascade patterns (Compared to the incorporated cascade patterns, the detected cascades contain more undirected instead of directed edges.). These additional edges are detected since the cascades were incorporated into the data in close intervals. Because of this, a cascade might still be active when delays of a subsequent cascade start.


Permit FINAL_APPROVED by SUPERVISOR:Start trip
Permit APPROVED by ADMINISTRATION:Permit FINAL_APPROVED by SUPERVISOR
Permit APPROVED by ADMINISTRATION:Request For Payment SUBMITTED by EMPLOYEE
Permit APPROVED by ADMINISTRATION:Permit APPROVED by BUDGET OWNER
Permit APPROVED by ADMINISTRATION:Send Reminder
Permit SUBMITTED by EMPLOYEE:Permit APPROVED by ADMINISTRATION
Permit APPROVED by ADMINISTRATION:Permit REJECTED by BUDGET OWNER
Permit APPROVED by ADMINISTRATION:Permit FINAL_APPROVED by SUPERVISOR
Declaration SAVED by EMPLOYEE:Declaration SUBMITTED by EMPLOYEE
Permit FINAL_APPROVED by SUPERVISOR:Declaration SUBMITTED by EMPLOYEE
Declaration APPROVED by ADMINISTRATION:Declaration SUBMITTED by EMPLOYEE

(a) cascade pattern 1

(b) cascade pattern 2

Fig. 5. Exemplary cascade patterns retrieved from BPI 2020 dataset

Figure 7 shows that even though the quality of the results decreases slightly with increasing noise, it still stays at a pretty high level and our approach can deliver meaningful results.

To compare our results, we slightly adapted the method from [12] to our use case and implemented it using python. We tested the approach on our synthetic log with different parameters and achieved the best results (i.e. all cascades were detected, with minimum result set size) with *outlier threshold* $= 5$, *time interval* $= 60$ (batch length in days) and *minimum support* $= 3$ (for frequent subgraph mining). The resultset consisted of 153 cascades, where each cascade covered parts of the incorporated cascade patterns, and every cascade pattern was represented entirely by at least one frequent subgraph. Even though all incorporated cascade patterns were detected, the size of the result set was considerable, making it very difficult to interpret it. Furthermore, many frequent subgraphs differed from other subgraphs in only one node or edge and thus did not contribute any new valuable information. We observed that the size of the result set could vary significantly for different time intervals. At the same time, it is challenging to choose an appropriate time interval because it cannot be derived from the structure of the process. The size of the time interval defines the maximum duration of a cascade. However, this information is not given in a real-life cascade mining scenario, which means that by choosing a too small time interval, one might neglect

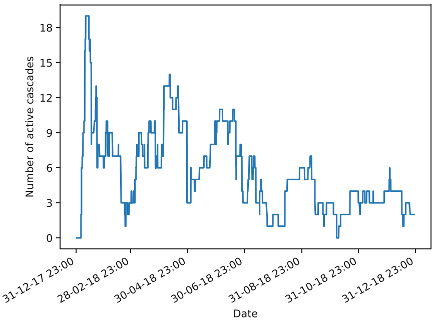


Fig. 6. Number of active cascades, travel permit log of BPI 2020 dataset

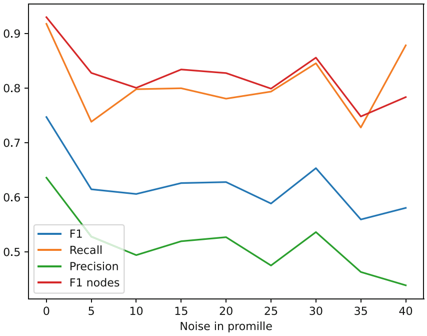


Fig. 7. Results on synthetic data with increasing noise

longer-lasting cascades. At the same time, a smaller time interval might be desirable, as it leads to a smaller result set. Cascades of cascade pattern 3 (see Fig. 4e) have an approximate length of 30 days. This cascade pattern was only detected entirely from a time interval of 30 days onward. For a bi-weekly interval, 3 of 106 frequent cascades had an mcs-similarity of 0.67 to cascade 3. For all the lower intervals, cascade 3 was not detected at all.

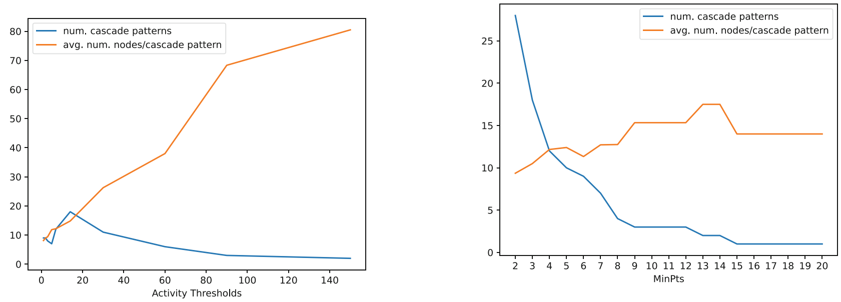
In conclusion, our approach yielded a far smaller result set (3 vs. 153 detected cascade patterns) that still contained the same amount of information. At the same time, we achieved good results even in a streaming scenario (compared to an event log), where we had to process traces consecutively.

5.2 Travel Reimbursement Process

In addition to the synthetic data, we also tested our approach on real-world process data that was published for the BPI Challenge 2020[1]. The data was collected from the travel reimbursement process at TU/e in 2017 and 2018 and contained files for different subprocesses. Travel reimbursement is a process present in nearly every company and thus forms a good basis for our evaluation. For international trips, employees have to request a travel permit before starting the trip. At the end of the trip, they can request reimbursement of their costs. We chose this process for our tests because here, an array of delays can, in the worst-case, risk the entire trip. For our experiments we used the travel permits log, which contains the described process, and reduced it to traces in 2018.

With an activity threshold of 7 days and an outlier threshold of 5, we detected 12 cascade patterns, showing two examples in Fig. 5. 528 segment-level event outliers were grouped into 124 cascades (+ 19 deleted cascades with one node). As Fig. 6 shows, the number of active cascades changed in waves and decreased

[1] https://icpmconference.org/2020/bpi-challenge/
DOI: https://doi.org/10.4121/uuid:52fb97d4-4588-43c9-9d04-3604d4613b51.



(a) Results for different activity thresholds with a constant *minPts* of 4

(b) Results for different values of *minPts* with a constant activity threshold of 7 days

Fig. 8. Parameter Sensibility (constant outlier threshold of 5)

over the year. A maximum number of 18 cascades was active at the beginning of the year, which might be due to many requests regarding trips later in the year.

Figure 8 shows how the results vary for different parameters. We observed that with an increasing activity threshold, the number of detected cascade patterns decreases while the average number of nodes per cascade pattern increases (Fig. 8a). For a large activity threshold, e.g. 150 days, cascade nodes stay active for a very long time. New incoming outliers are declared correlated to existing cascades for a longer time, and no new cascades are started. This leads to larger cascades and thus also larger cascade patterns. New cascades are started more frequently for smaller activity thresholds, resulting in more cascades and fewer nodes per cascade. At this point, it needs to be mentioned that an activity threshold of 150 days or even 60 days is probably very unrealistic for this kind of process. The activity threshold resembles the time in which an anomaly can affect process performance. A proper value for the activity threshold can be picked in the context of the process structure, and in contrast to the time interval from [12] no prior knowledge of the cascades is needed.

The number of cascade patterns also decreases with an increasing *minPts* (input parameter DBSCAN) (Fig. 8b). The *minPts* parameter can be used as an importance regulator. The higher it is, the fewer cascade patterns are detected and the more cascades each pattern represents.

6 Conclusion

With our novel approach PErrCas, we are able to track correlated outliers over multiple process instances by continuously adding outliers to existing cascades and creating new cascades. We differentiate between two different correlations: accumulations of outliers in one segment and correlated outliers in different segments. The set of cascades can be analyzed in regular intervals to create cascade patterns and get an overall picture of the cascades. This continuous approach avoids defining an observation window beforehand. Instead, we consider how long an outlier can affect future process performance and track cascades as long they influence process performance. A useful extension of our work would be to discover a good candidate threshold for this automatically.

So far, our method only works on trace streams because we need entire traces to build segment-level events and detect outliers. Future work could examine how error cascades can be detected in event streams. Another issue for future work is the correlation between outliers. We declare outliers to be correlated if they are close in time and their segments overlap. However, there are also many other ways in which two anomalies could be correlated.

References

1. Bunke, H., Shearer, K.: A graph distance metric based on the maximal common subgraph. Pattern Recogn. Lett. **19**(3–4), 255–259 (1998)
2. Burattin, A.: PLG2: multiperspective process randomization with online and offline simulations. In: BPM (Demos), pp. 1–6. Citeseer (2016)
3. Denisov, V., Fahland, D., van der Aalst, W.M.P.: Unbiased, fine-grained description of processes performance from event data. In: Weske, M., Montali, M., Weber, I., vom Brocke, J. (eds.) BPM 2018. LNCS, vol. 11080, pp. 139–157. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98648-7_9
4. Ester, M., Kriegel, H.P., Sander, J., Xu, X., et al.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD, vol. 96, pp. 226–231 (1996)
5. Fournier-Viger, P., Lin, J.C.W., Kiran, R.U., Koh, Y.S., Thomas, R.: A survey of sequential pattern mining. Data Sci. Pattern Recogn. **1**(1), 54–77 (2017)
6. Guo, S., Lin, Y., Feng, N., Song, C., Wan, H.: Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 922–929 (2019)
7. Laur, P., Symphor, J., Nock, R., Poncelet, P.: Mining sequential patterns on data streams: a near-optimal statistical approach. In: Proceedings of the 2nd International Workshop on Knowledge Discovery from Data Streams (2005)
8. Liu, W., Zheng, Y., Chawla, S., Yuan, J., Xing, X.: Discovering spatio-temporal causal interactions in traffic data streams. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1010–1018 (2011)
9. Marascu, A., Masseglia, F.: Mining sequential patterns from temporal streaming data. In: Proceedings of the 1st ECML/PKDD Workshop on Mining Spatio-Temporal Data (MSTD 2005), pp. 1–13. Citeseer (2005)
10. Richter, F., Maldonado, A., Zellner, L., Seidl, T.: OTOSO: online trace ordering for structural overviews. In: Leemans, S., Leopold, H. (eds.) ICPM 2020. LNBIP, vol. 406, pp. 218–229. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-72693-5_17
11. Senderovich, A., Di Francescomarino, C., Ghidini, C., Jorbina, K., Maggi, F.M.: Intra and inter-case features in predictive process monitoring: a tale of two dimensions. In: Carmona, J., Engels, G., Kumar, A. (eds.) BPM 2017. LNCS, vol. 10445, pp. 306–323. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-65000-5_18

12. Toosinezhad, Z., Fahland, D., Köroğlu, Ö., Van Der Aalst, W.M.: Detecting system-level behavior leading to dynamic bottlenecks. In: 2020 2nd International Conference on Process Mining (ICPM), pp. 17–24. IEEE (2020)
13. Wu, Y., Tan, H.: Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework. arXiv preprint arXiv:1612.01022 (2016)
14. Xu, C., Chen, Y., Bie, R.: Sequential pattern mining in data streams using the weighted sliding window model. In: 2009 15th International Conference on Parallel and Distributed Systems, pp. 886–890. IEEE (2009)
15. Zubaroğlu, A., Atalay, V.: Data stream clustering: a review. Artif. Intell. Rev. **54**(2), 1201–1236 (2020). https://doi.org/10.1007/s10462-020-09874-x

Continuous Performance Evaluation for Business Process Outcome Monitoring

Suhwan Lee[1(✉)], Marco Comuzzi[2], and Xixi Lu[1]

[1] Utrecht University, Utrecht, The Netherlands
{s.lee,x.lu}@uu.nl
[2] Ulsan National Institute of Science and Technology, Ulsan, Republic of Korea
mcomuzzi@unist.ac.kr

Abstract. While a few approaches to online predictive monitoring have focused on concept drift model adaptation, none have considered in depth the issue of performance evaluation for online process outcome prediction. Without such a continuous evaluation, users may be unaware of the performance of predictive models, resulting in inaccurate and misleading predictions. This paper fills this gap by proposing a framework for evaluating online process outcome predictions, comprising two different evaluation methods. These methods are partly inspired by the literature on streaming classification with delayed labels and complement each other to provide a comprehensive evaluation of process monitoring techniques: one focuses on real-time performance evaluation, i.e., evaluating the performance of the most recent predictions, whereas the other focuses on progress-based evaluation, i.e., evaluating the ability of a model to output correct predictions at different prefix lengths. We present an evaluation involving three publicly available event logs, including a log characterised by concept drift.

Keywords: predictive monitoring · process outcome · event stream

1 Introduction

The process mining research in recent years has started focusing on the *online* realisation of typical use cases, such as process discovery [5] and conformance checking [4]. In the *online* perspective, an event log is a stream of events, which become available for analysis as soon as they are logged. Conversely, the traditional *offline* perspective considers an event log as a batch of events logged in a certain time span.

On the one hand, the online perspective naturally brings some benefits: online models need not waiting for a large number of events to be accumulated in an event log before performing an analysis; they also allow updating the analytic models in real time when a new event is received, and, consequently, they may naturally adapt to concept drift in the process generating the events [13]. On the other hand, this perspective also poses a number of challenges: new techniques

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 237–249, 2022.
https://doi.org/10.1007/978-3-030-98581-3_18

must be developed to adapt to the streaming nature of events; owing to the finite memory assumption of streaming analytics, only a limited number of recent events can be available for the analysis at any given time [8]; finally, run time may become a concern, since models may need to be updated with every new event received and before the next event will be received.

This paper focuses on the predictive monitoring use case in process mining and, more specifically, on the continuous evaluation of the predictions of process outcomes [16], whereby the objective is to predict the (usually binary) outcome label of a running process case and to continuously evaluate these predictions. For instance, the possible outcome of a case would be that the personal loan request is accepted or rejected in a loan application process.

In the offline perspective, the outcome prediction problem is solved by encoding the completed cases into feature-label vectors, which are then used to train and test a predictive classification model. Besides the obvious need to consider online classification techniques for developing the predictive model, in the online perspective the outcome prediction is an instance of the *delayed labels* [10] online classification task: while in the batch perspective all the feature vectors and labels of completed cases are available for training and testing, in the online perspective the label of a case normally becomes available only when the last event of that case is received. This is an issue to be taken into account when updating the predictive model and, consequently, to assess its performance.

The contribution of the paper is to develop two performance evaluation methods specifically-tailored to online outcome predictive monitoring. These methods are developed adapting the notion of *continuous evaluation* [8], which recently has emerged as a novel perspective for evaluating the performance in streaming classification with delayed labels, to the domain of process outcome prediction.

The paper is organised as follows. Related work is discussed in the next section. Section 3 introduces the overall framework, while the performance methods are presented in Sect. 4. The experimental results are reported in Sect. 5, while conclusions are drawn in Sect. 6.

2 Related Work

Several approaches recently have been proposed to deal with online process discovery [2,6] and online conformance checking [4,17].

As far as process predictive monitoring is concerned, Maisenbacher and Weidlich [13] have proposed to use incremental classifiers to deal with an event log as a stream of events, specifically aiming at creating outcome prediction models that can adapt to concept drift. They propose to evaluate the models using average accuracy across all the labels received in the stream and they evaluate their approach on different concept drifts injected in a single artificially-generated event log. Baier et al. [1] have investigated the issue of optimal data selection point for retraining an offline predictive model when a concept drift is observed in an event stream.

In the more general field of streaming classification, Žliobaitė [18] first has identified the issue of delayed labels, suggesting to map dynamically the distribution of the labels to detect the concept drift. Grzenda et al. [8] recently have introduced the continuous evaluation methodology for streaming classification with delayed labels, whereby the performance of a model is evaluated for each observation considering the amount of time left before the arrival of the corresponding label.

3 Continuous Prediction Evaluation Framework

Given the first n positive natural numbers $\mathbb{N}_n^+$ and a target set S, a sequence s is a function $s : \mathbb{N}_n^+ \rightarrow S$ mapping integer indexes to the elements of S. Given a set of activity labels A, the domain $\mathbb{N}^+$ of timestamps, and a set of I attribute domains D_i, we define the set of event attributes as $E = A \times \mathbb{N}^+ \times [D_1 \times \ldots \times D_i \times \ldots \times D_I]$. A trace σ is a sequence of n events $\sigma : \mathbb{N}_n^+ \rightarrow E$. We denote with $\mathcal{T}$ the universe of sequences of events and with $\mathcal{E}$ the event universe, with $\mathcal{E} = E \times J$, where J is a set of possible case ids. An event stream is an infinite sequence $\Psi : \mathbb{N}^+ \rightarrow \mathcal{E}$.

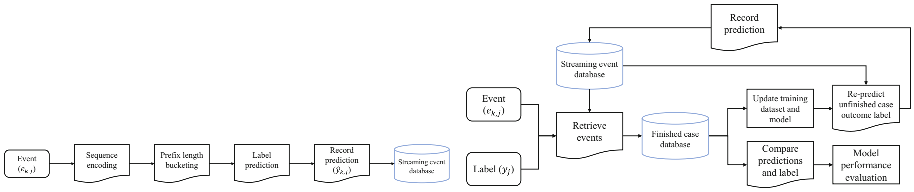
For simplicity, we write events as $e_{k,j}$, where k indicates their position in a trace and traces as $\sigma_j = \langle e_{1,j}, \ldots, e_{i,j}, \ldots, e_{N_j,j} \rangle$, where N_j is the number of events in the trace σ_j. The function $t : \mathcal{E} \rightarrow \mathbb{N}^+$ returns the timestamp of an event. The prefix function $pref : \mathcal{T} \times \mathbb{N}^+ \rightarrow \mathcal{T}$ returns the first p events of a trace, i.e., $pref(\sigma_j, p) = \langle e_{1,j}, \ldots, e_{p,j} \rangle$, with $p \leq N_j$. Note that, for the evaluation, event streams are generated from event logs in which multiple events may have the same timestamp. For the events that have the same timestamp, we assume that the ordering of the events in an event log reflects their true ordering and use this order in the stream to calculate prefixes.

A trace σ is associated with a binary outcome label and, without loss of generality, we assume that the value of this label becomes known with the last event $e_{N_j,j}$ of a trace. Therefore, we define a labelling function as a partial function $y : E \nrightarrow \{0, 1\}$, which returns the label of a trace in correspondence of its last event. For clarity and with an abuse of notation, we denote the label of a trace σ_j as y_j.

A sequence encoder is a function f, with $f : \mathcal{T} \rightarrow \mathcal{X}_1 \times \ldots \times \mathcal{X}_w \times \ldots \times \mathcal{X}_W$ mapping a prefix into a set of features defined in the domains $\mathcal{X}_w$. A process outcome prediction model *pom* is a function $\hat{y} : \mathcal{X}_1 \times \ldots \times \mathcal{X}_w \times \ldots \times \mathcal{X}_W \rightarrow \{0, 1\}$ mapping an encoded prefix into its predicted label.

In offline settings, prefixes may be divided into separate buckets and a different prediction model may be maintained (trained/tested) for each bucket of prefixes. We adopt the same design in this work considering prefix-length bucketing [11] of traces: a different predictive model pom_k is trained and tested using a set of prefixes of length $k = 1, \ldots, K$, where the maximum prefix K may vary for each event log. Thus, we define an outcome prediction framework pof as a collection of outcome prediction models pom_k, that is, $pof = \{pom_k\}_{k=1,\ldots,K}$. We use index-based encoding of prefixes [11], in which features in a prefix are generated for each event in it. We use one-hot encoding for the categorical attributes, such as the activity or the resource label, whereas continuous values are encoded as is. As classifiers, we consider incremental streaming classifiers that can be updated when a new label is received [9].

The processing of one event $e_{k,j}$ belonging to trace σ_j is schematised in Fig. 1. Note that this way of processing events applies after a given grace period, which is defined by a specific number L of labels received. That is, during the grace period, the labels received are only used to train the models in the framework. The event $e_{k,j}$ may either be the last of σ_j, i.e., $k = N_j$, in which case the label y_j becomes known, or not. When an event is not the last one of its trace (see Fig. 1a), it is used to generate a new prefix $pref(\sigma_j, k)$. Then, a prediction $\hat{y}_{k,j}$ for the new prefix $pref(\sigma_j, k)$ can be computed using the model pom_k. Receiving the last event $e_{k,j}$, with $k = N_j$ of a trace σ_j and its label (see Fig. 1b) enables (i) to evaluate all the predictions $\hat{y}_{n,j}$ that have been generated for the prefixes $pref(\sigma_j, n)$, with $n = 1, \ldots, \max\{K, N_j\}$ using the model pom_n (evaluation before training) and (ii) to update the models pom_n, with $n = 1, \ldots, \max\{K, N_j\}$ in the framework, owing to the availability of new labelled prefixes. Finally, it is possible (iii) to compute a new set of predicted labels $\hat{y}_{l,k}$, with $l \neq j$ and $k = 1, \ldots, \max\{N_l, N_j\}$ for all the prefixes for which a label has not been yet received (train and retest).



(a) Processing of an event without label (b) Processing of an event with label

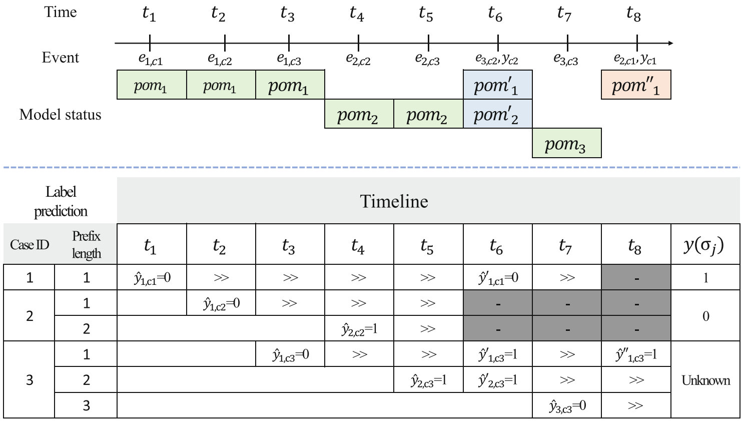
Fig. 1. An overview of the continuous evaluation framework

Next, we propose the novel methods to evaluate the performance of an online outcome prediction framework pof.

4 Performance Evaluation Methods

One of the major challenges in streaming classification is the performance evaluation, particularly in cases, such as the one of online process outcome prediction, in which the labels are *delayed*. The challenge arises because of the dynamic nature of the classification models considered in the framework: the models available to generate predictions are updated with each new label received; therefore, the same observation may be associated with different predictions generated by different versions of the model that applies to it.

Figure 2 exemplifies what stated above in the context of the proposed framework, considering 3 process cases and prefix length up to 3. First, note that different versions of the same model pom_k are generated along the considered timeline. In particular, a new version of pom_k is generated when a new label y_j for a case σ_j, with $N_j < k$, is received. Second, new predictions for prefixes of length k are generated each time a new version of pom_k is available. Finally, note that a prediction can only be evaluated when the corresponding label becomes available. In the example, the predictions generated for all the prefixes of case 3 cannot be evaluated because the label of case 3 has yet to be received at t_8.



Label prediction		Timeline								
Case ID	Prefix length	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	$y(\sigma_j)$
1	1	$\hat{y}_{1,c1}=0$	>>	>>	>>	>>	$\hat{y}'_{1,c1}=0$	>>	-	1
2	1		$\hat{y}_{1,c2}=0$	>>	>>	>>	-	-	-	0
	2				$\hat{y}_{2,c2}=1$	>>	-	-	-	
3	1			$\hat{y}_{1,c3}=0$	>>	>>	$\hat{y}'_{1,c3}=1$	>>	$\hat{y}''_{1,c3}=1$	Unknown
	2					$\hat{y}_{2,c3}=1$	$\hat{y}'_{2,c3}=1$	>>	>>	
	3							$\hat{y}_{3,c3}=0$	>>	

Fig. 2. Evaluation methods: supporting example

We propose two ways to approach the issue of performance evaluation of the proposed framework: using a local observation timeline within a process case or a real-time global perspective on recent process cases. The former is inspired by the literature on streaming classification with delayed labels [8], whereas the latter is a novel perspective that we argue is specifically tailored to the context of process outcome predictive monitoring.

4.1 Evaluating Performance Using a Local Timeline

The local timeline perspective on performance evaluation in streaming classification is also referred to as the *continuous evaluation* of a model [8]. In the context of process outcome predictive monitoring, it translates naturally into evaluating the performance along a timeline that establishes the *progress* of the execution of a case. The traditional view of case progress in predictive monitoring is the *prefix length*, i.e., measuring the progress of a case using the number of events that have occurred in it. Therefore, we define a continuous evaluation method by prefix length.

Continuous Evaluation by Prefix Length. The objective of the continuous evaluation by prefix length is to evaluate the performance of an online outcome classification framework at each prefix, i.e., to answer the question "*How likely is the framework to output a correct prediction for a running trace at prefix length k?*".

The design of a suitable performance measure starts from aggregating the predictions available for a case at a given prefix length, in order to obtain one reference value for each trace for which a label has been received at each prefix length. Inspired by the literature on streaming classification with delayed labels [8], we aggregate multiple predictions using a majority rule. That is, given the set $\hat{Y}_{k,j} = \{\hat{y}^l_{k,j}\}_{l=1,\ldots,L}$ of L predictions available for trace σ_j at prefix length k, and given $\hat{Y}^o_{k,j} = \{y \in \hat{Y}_{k,j} : y = o\}$ as the set of predictions evaluating to o, with $o \in \{0,1\}$, the aggregated prediction for σ_j at prefix k is:

$$\hat{y}^{agg}_{k,j} = \begin{cases} 1 & \text{if } |\hat{Y}^1_{k,j}| \geq |\hat{Y}^0_{k,j}| \\ 0 & \text{otherwise} \end{cases}$$

Once the multiple predictions for a case at a given prefix length have been aggregated, the performance can be evaluated using any of the standard confusion matrix-based performance measure for classification. For instance, given the accuracy $acc(\hat{y}_j)$ of an individual prediction for trace σ_j at any prefix length:

$$acc(\hat{y}_j) = \begin{cases} 1 & \text{if } \hat{y}_j = y_j \\ 0 & \text{otherwise} \end{cases}$$

the accuracy $acc_k(pof)$ of an outcome prediction framework pof at prefix length k is defined as:

$$acc_k(pof) = \frac{1}{J} \cdot \sum_{j=1}^{J} acc(\hat{y}^{agg}_{k,j})$$

where J is the number of traces in the stream (or labels received).

For example, in Fig. 2, let us consider only the traces $c1$ and $c2$, for which the label has been received. The most frequent prediction at prefix length $k = 1$ for both trace $c1$ and $c2$ is 0 (no predictions equal to 1 are available). Given that the label of $c1$ and $c2$ are 1 and 0, respectively, the accuracy of the framework at prefix length $k = 1$ is 0.5.

4.2 Real-Time Model Performance

This method for performance evaluation considers a global perspective on recent predictions obtained by the framework, answering the question "*How likely are the most recent prediction(s) obtained from a model to be eventually correct?*" Instead of aggregating the performance at given progress rates or prefix lengths for cases, in the real-time method we first define w as the size of a test window containing the traces $\{\sigma_w\}_{w=1,\ldots,W}$ associated with the latest W labels y_w that have been received. We then consider the average of the performance across all the predictions available, at any prefix length, for each trace σ_w in this window.

Table 1. Descriptive statistics of event logs used in the evaluation

	# cases	# events	# activity	# variants	Avg events/case	Median events/case	# true labels	# false labels
BPIC 2015_1	1199	52217	289	1100	43.55	44	506	693
BPIC 2017	1878	23941	22	376	12.75	12	576	1302
IRO5K	1000	10756	20	111	10.76	11	237	763

When a new label is received, then, to accommodate this new trace, the trace in the window associated with the oldest label received is removed from the window.

Given $\hat{Y}_w$ as the set of predictions $\hat{y}_{k,w}$ available for a trace σ_w at any prefix length k, the real-time accuracy $acc_{rt}(pof)$ of an outcome predictive framework is then defined as follows:

$$acc_{rt}(pof) = \frac{1}{W} \cdot \sum_{w=1}^{W} \left[\frac{1}{|\hat{Y}_w|} acc(\hat{y}_{k,w}) \right].$$

Let us consider $W = 2$ in the example of Fig. 2. At t_8, the traces $c1$ and $c2$ are included in the window, because they are associated to the last 2 labels received. For $c1$, there are 2 predictions available (at $t1$ and $t6$), all incorrect. For $c2$, there are 2 predictions available (one correct $t2$ and one incorrect at $t4$). Therefore, the real-time accuracy at t_8 for $W = 2$ of the framework is 0.25.

5 Experimental Analysis and Results

We consider 3 publicly accessible event logs. The BPIC 2015_1[1] is a log from a Dutch municipality of a process for granting building permissions. The outcome label in this log is 1 (true) when a trace contains the activity 'create procedure confirmation', and 0 otherwise. The BPIC 2017[2] event log refers to a personal loan request process at a Dutch financial institute. The outcome label evaluates to 1 (true) if a request is accepted, and 0 otherwise. The IRO5K[3] event log is a synthetic log regarding the assessment of loan applications [12]. The outcome label evaluates to 1 (true) if a request is accepted, and 0 otherwise. The two BPIC logs have been chosen because they are real world event logs that have been used in the previous research on outcome predictive monitoring [16] and they differ greatly in terms of variability. Specifically (see Table 1), the BPIC 2015 event log shows a higher number of activity labels and trace variants in respect of BPIC 2017. The IRO5K event log has been chosen because it is characterised by process drift.

[1] at: https://data.4tu.nl/articles/dataset/BPI_Challenge_2015_Municipality_1/12709154/1.

[2] at: https://data.4tu.nl/articles/dataset/BPI_Challenge_2017_-_Offer_log/12705737.

[3] at: https://data.4tu.nl/articles/dataset/Business_Process_Drift/12712436.

The process outcome prediction is an instance of early time series prediction and the research community focuses on building an accurate model for early predictions [16]. We consider a different maximum prefix length for each event log: 44 for BPIC 2015_1, 14 for BPIC 2017 and 11 for IRO5K. The minimum prefix length is set to 2 for all event logs.

As streaming classifiers, we consider 3 different tree-based incremental classifiers typically adopted in streaming classification: the Hoeffding Tree Classifier (HTC) [9], the Hoeffding Adaptive Tree Classifier (HATC) [3], and the Extremely Fast Decision Tree (EFDT) [14]. These algorithms are tree-based classifiers which incrementally construct split points depending on the confidence for information gain.

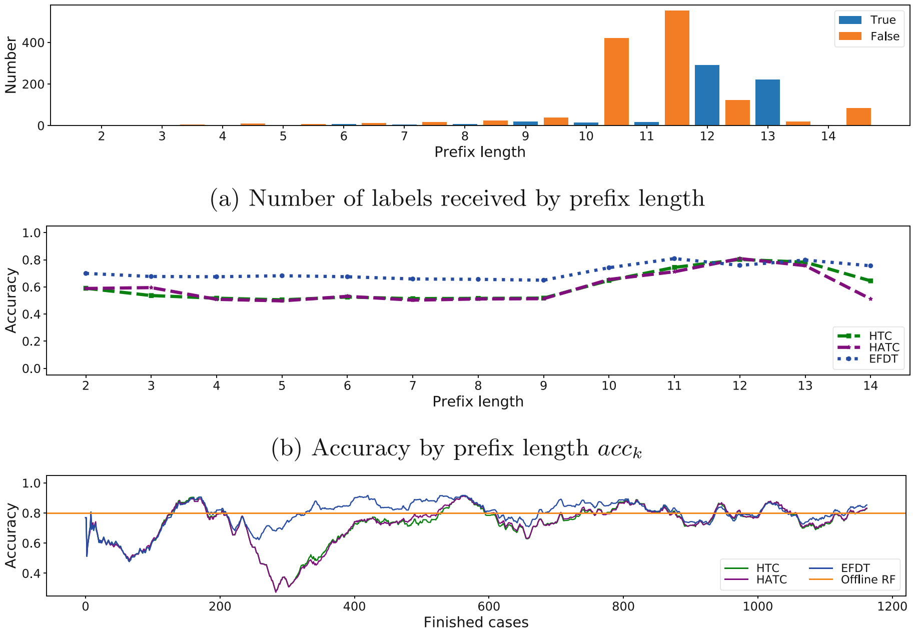
Any streaming classification framework normally requires a grace period to allow a proper initialisation of the classifier [7]. As grace period in all experiments, we consider 200 labels received. That is, until the 200-th label is received, the events without label in the stream are not processed and the labels are used only to update the classification models. We consider the implementation of the classifiers provided by the Python package River [15], setting 100 observations for the classification tree leaf between split attempts and maximising information gain as split criterion in all streaming classifiers.

For the real-time performance evaluation, we consider $W = 50$ cases as window size and we also include as a baseline the results obtained using an offline outcome predictive model developed using the Random Forest (RF) classifier, implemented using the Python package 'scikit-learn' with 100 estimators, using a 70/30 train/test split and 10-fold cross-validation. Finally, to support the discussion we also plot for each log the number of true and false labels received at each prefix length.

The code and data to reproduce the experiments presented in this section, as well as additional results that have been omitted in this section due to lack of space, are available at https://github.com/ghksdl6025/streaming_prediction4pm.

Figure 3b shows the continuous evaluation results for BPIC2017 using the prefix length method. Regarding the prefix length method, the EFDT shows a better performance than the other classifiers. Generally, the accuracy of the classification increases after prefix length 9.

From the results, we can observe that the continuous evaluation method provides diagnostic information to help deciding which model to deploy. The prefix length method reveals that EFDT performs better than other classifiers for ongoing cases until prefix length 11. Therefore, the EFDT classifier should be preferred if the predictions obtained from the outcome decision framework are used to take the decision after an event in a case has occurred (e.g., "What's the best thing to do after the client has replied?"), and that event normally happens within the first 11 executed in a trace.



(a) Number of labels received by prefix length

(b) Accuracy by prefix length acc_k

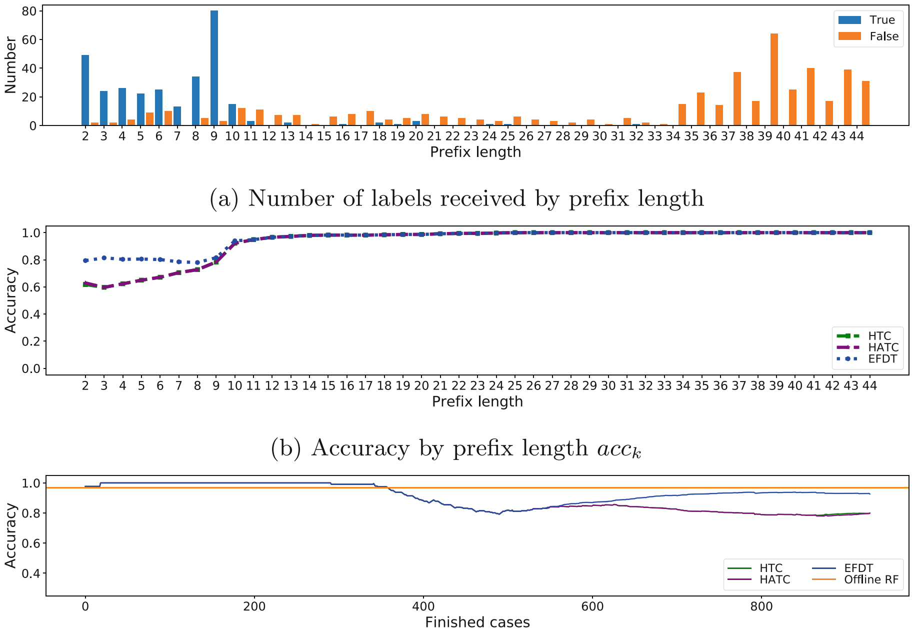
(c) Real-time accuracy acc_{rt} ($W = 50$)

Fig. 3. BPIC 2017 log experiment results.

Figure 3c shows the real-time perspective results for BPIC2017. Except for a sharp drop in the accuracy of HATC and HTC after the 200-th label received, the accuracy remains above 0.6 and comparable with the one of the offline baseline. From a diagnostic standpoint, the real-time performance perspective generally reveals whether the predictive framework outputs correct predictions for all cases *now*, on cases recently finished. It also can provide diagnostic information when there is a sudden change in the process, showing how each model performs after such a change.

For the BPIC2015_1 log (see Fig. 4), let us first consider the real-time evaluation method (Fig. 4c). We observe that after approximately 350 labels received, the performance of all models drops significantly. This may be caused by an (unknown) concept drift in the event log. After 600 cases, we see that EFDT recovers whereas the performance of the other two models does not improve until the end. Therefore, also in this case EFDT emerges as more likely to recover after a drop in the performance than HATC and HTC.

For the IRO5K event log, Fig. 5b shows the results of the continuous evaluation by prefix length. The results for this log must be interpreted considering the distribution of labels received across prefix lengths. Until prefix length 6 no new labels are received, which justifies the constant high accuracy by prefix length until then (given that the models trained during the grace period are



(a) Number of labels received by prefix length

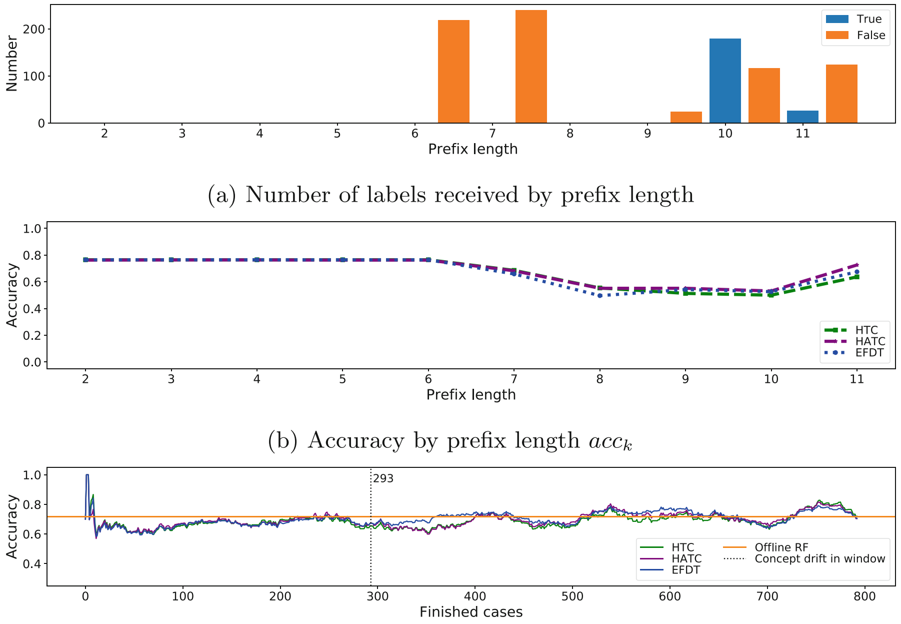
(b) Accuracy by prefix length acc_k

(c) Real-time accuracy acc_{rt} $(W = 50)$

Fig. 4. BPIC 2015_1 log experiment results.

fairly accurate). Then most false labels are received at prefix length 6 and 7, whereas no true labels are received before prefix 10. Therefore, the models used to generate predictions change substantially after prefix 6, which may justify the drop in performance.

More insightful for this event log is the analysis of the real-time performance, which is shown in Fig. 5c. In particular, there is no specific drop of the accuracy when the process drift occurs. The EFDT classifier, in particular, actually increases its accuracy after the concept drift. Generally, after the concept drift occurs the performance of all classifiers recovers relatively quickly (in less than 100 labels received) and, until the end of the stream, remains higher for most time in respect of the offline baseline. This can be interpreted as encouraging evidence that the proposed framework with the EFDT model, at least in this case, naturally adapts to concept drift in the process generating the event stream.



(a) Number of labels received by prefix length

(b) Accuracy by prefix length acc_k

(c) Real-time accuracy acc_{rt} ($W = 50$); the concept drift occurs after the 293-th label is received.

Fig. 5. IRO5K log experiment results.

6 Conclusions

In this paper, we propose a continuous performance evaluation framework for online process outcome prediction techniques. Moreover, we propose two concrete evaluation methods, which assess the performance of the prediction techniques from both a local perspective and a global real-time perspective. The experimental analysis of our framework on the three real-life logs has shown that our framework can reveal very interesting results from different perspectives and provide novel insights into how predictive models perform.

As far as the experimental analysis is concerned, the streaming classifier EFDT has emerged as the best performing and robust classifier for outcome prediction with event streams. This confirms the claim of the proposers of the EFDT classifier that it should be preferred to other incremental tree classifiers based on the Hoeffding bound in most application scenarios [14]. Unexpectedly, although HATC is specifically designed to adapt to concept drift, our evaluation framework shows that EFDT appears as the best classifier at dealing with concept drifts in the event logs. This may be due to the window selection of EFDT, which simply adapts quickly to the new data, whereas in the other two models (HATC and HTC) there is a threshold controlling the model adaptation, which may limit the ability to adapt to small changes in the feature vectors.

As a future work, the proposed framework can be extended with various performance evaluation methods. For example, instead of the prefix-length perspective, we may also complement the framework with a method that evaluates the accuracy from the last-state perspective. Providing performance evaluation from multiple perspectives may help ease the issue of explainability of online predictive monitoring models, which should also be further investigated.

References

1. Baier, L., Reimold, J., Kühl, N.: Handling concept drift for predictions in business process mining. In: 2020 IEEE 22nd Conference on Business Informatics (CBI), vol. 1, pp. 76–83. IEEE (2020)
2. Batyuk, A., Voityshyn, V.: Streaming process discovery for lambda architecture-based process monitoring platform. In: 2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT), vol. 1, pp. 298–301. IEEE (2018)
3. Bifet, A., Gavaldà, R.: Adaptive learning from evolving data streams. In: Adams, N.M., Robardet, C., Siebes, A., Boulicaut, J.-F. (eds.) IDA 2009. LNCS, vol. 5772, pp. 249–260. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03915-7_22
4. Burattin, A., Carmona, J.: A framework for online conformance checking. In: Teniente, E., Weidlich, M. (eds.) BPM 2017. LNBIP, vol. 308, pp. 165–177. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-74030-0_12
5. Burattin, A., Sperduti, A., van der Aalst, W.M.: Heuristics miners for streaming event data. arXiv preprint: arXiv:1212.6383 (2012)
6. Burattin, A., Sperduti, A., van der Aalst, W.M.: Control-flow discovery from event streams. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp. 2420–2427. IEEE (2014)
7. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 71–80 (2000)
8. Grzenda, M., Gomes, H.M., Bifet, A.: Delayed labelling evaluation for data streams. Data Mining Knowl. Disc. **34**(5), 1237–1266 (2019). https://doi.org/10.1007/s10618-019-00654-y
9. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 97–106 (2001)
10. Krempl, G., et al.: Open challenges for data stream mining research. ACM SIGKDD Explor. Newsl. **16**(1), 1–10 (2014)
11. Leontjeva, A., Conforti, R., Di Francescomarino, C., Dumas, M., Maggi, F.M.: Complex symbolic sequence encodings for predictive monitoring of business processes. In: Motahari-Nezhad, H.R., Recker, J., Weidlich, M. (eds.) BPM 2015. LNCS, vol. 9253, pp. 297–313. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23063-4_21
12. Maaradji, A., Dumas, M., La Rosa, M., Ostovar, A.: Fast and accurate business process drift detection. In: Motahari-Nezhad, H.R., Recker, J., Weidlich, M. (eds.) BPM 2015. LNCS, vol. 9253, pp. 406–422. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23063-4_27

13. Maisenbacher, M., Weidlich, M.: Handling concept drift in predictive process monitoring. SCC **17**, 1–8 (2017)
14. Manapragada, C., Webb, G.I., Salehi, M.: Extremely fast decision tree. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1953–1962 (2018)
15. Montiel, J., et al.: River: machine learning for streaming data in python (2020)
16. Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M.: Outcome-oriented predictive process monitoring: review and benchmark. ACM Trans. Knowl. Disc. Data (TKDD) **13**(2), 1–57 (2019)
17. van Zelst, S.J., Bolt, A., Hassani, M., van Dongen, B.F., van der Aalst, W.M.P.: Online conformance checking: relating event streams to process models using prefix-alignments. Int. J. Data Sci. Anal. **8**(3), 269–284 (2017). https://doi.org/10.1007/s41060-017-0078-6
18. Žliobaite, I.: Change with delayed labeling: when is it detectable? In: 2010 IEEE International Conference on Data Mining Workshops, pp. 843–850. IEEE (2010)

PQMI 2021: 6th International Workshop on Process Querying, Manipulation, and Intelligence

6th International Workshop on Process Querying, Manipulation, and Intelligence (PQMI 2021)

The aim of the Sixth International Workshop on Process Querying, Manipulation, and Intelligence (PQMI 2021) was to provide a high-quality forum for researchers and practitioners to exchange research findings and ideas on methods and practices in the corresponding areas. *Process Querying* combines concepts from Big Data and Process Modeling and Analysis with Business Process Intelligence and Process Analytics to study techniques for retrieving and manipulating models of processes, both observed and recorded in the real-world and envisioned and designed in conceptual models, to systematically organize and extract process-related information for subsequent use. *Process Manipulation* studies inferences from real-world observations for augmenting, enhancing, and redesigning models of processes with the ultimate goal of improving real-world business processes. *Process Intelligence* looks into the application of representation models and approaches in Artificial Intelligence (AI), like knowledge representation and reasoning, search, automated planning, natural language processing, autonomous agents, and multi-agent systems, among others, for solving problems in process mining, that is automated process discovery, conformance checking, and process enhancement, and vice versa using process mining techniques to tackle problems in AI. Techniques, methods, and tools for process querying, manipulation, and intelligence have applications in Business Process Management and Process Mining. Examples of practical problems tackled by the themes of the workshop include business process compliance management, business process weakness detection, process variance management, process performance analysis, predictive process monitoring, process model translation, syntactical correctness checking, process model comparison, infrequent behavior detection, process instance migration, process reuse, and process standardization.

PQMI 2021 attracted six high-quality submissions. Each paper was reviewed by at least three members of the Program Committee. The review process led to three accepted papers. The keynote by Jessica Ambrosy entitled "Celonis PQL: A Query Language for Process Mining", which opened the workshop, introduced the audience to a domain-specific language developed by engineers at Celonis SE tailored towards a special process data model for answering process mining related questions posed by business users. The accepted paper by Alessandro Berti, Gyunam Park, Majid Rafiei, and Wil van der Aalst presented and evaluated an approach for extracting event logs from SAP ERP systems. The paper by Luciana Barbieri, Edmundo Roberto Mauro Madeira, Kleber Stroeh, and Wil van der Aalst proposed a reference architecture to support a conversational interface for process mining queries. This paper won the best paper award of the workshop. Finally, the paper by Julian Theis, Ilia Mokhtarian, and Houshang Darabi investigated the suitability of Adversarial System Variant Approximation that leverages Generative Adversarial Networks for measuring the generalization of process models discovered from logs of IT-systems.

We invite the reader to browse through the papers of the workshop in these proceedings to learn more about the latest advances in research in process querying, process manipulation, and process intelligence.

October 2021 PQMI Workshop Organizers

Organization

Workshop Organizers

Artem Polyvyanyy	The University of Melbourne
Arthur ter Hofstede	Queensland University of Technology
Claudio Di Ciccio	Sapienza University of Rome
Renuka Sindhgatta	IBM Research - India
Sebastian Sardina	RMIT University

Program Committee

Agnes Koschmider	Kiel University
Amin Beheshti	Macquarie University
Anna Kalenkova	The University of Melbourne
Catarina Moreira	Queensland University of Technology
Chiara Di Francescomarino	Fondazione Bruno Kessler-IRST
Hagen Völzer	IBM Research – Zurich
Han van der Aa	University of Mannheim
Hye-Young Paik	The University of New South Wales
Jochen De Weerdt	Katholieke Universiteit Leuven
Kanika Goel	Queensland University of Technology
María Teresa Gómez-López	Universidad de Sevilla
Maurizio Proietti	CNR-IASI
Minseok Song	Pohang University of Science and Technology
Monika Gupta	IBM Research – India
Pnina Soffer	University of Haifa
Seppe Vanden Broucke	Katholieke Universiteit Leuven
Shazia Sadiq	The University of Queensland

An Event Data Extraction Approach from SAP ERP for Process Mining

Alessandro Berti[1,2](✉), Gyunam Park[1], Majid Rafiei[1], and Wil M. P. van der Aalst[1,2]

[1] Process and Data Science Group (PADS), RWTH Aachen University, Aachen, Germany
a.berti@pads.rwth-aachen.de

[2] Fraunhofer Gesellschaft, Institute for Applied Information Technology (FIT), Sankt Augustin, Germany

Abstract. The extraction, transformation, and loading of event logs from information systems is the first and the most expensive step in process mining. In particular, extracting event logs from popular ERP systems such as SAP poses major challenges, given the size and the structure of the data. Open-source support for ETL is scarce, while commercial process mining vendors maintain connectors to ERP systems supporting ETL of a limited number of business processes in an ad-hoc manner. In this paper, we propose an approach to facilitate event data extraction from SAP ERP systems. In the proposed approach, we store event data in the format of object-centric event logs that efficiently describe executions of business processes supported by ERP systems. To evaluate the feasibility of the proposed approach, we have developed a tool implementing it and conducted case studies with a real-life SAP ERP system.

Keywords: SAP · ETL · Process Mining · Object-Centric Event Logs

1 Introduction

Process mining is a branch of data science including techniques to discover process models from event data, so-called *process discovery*, check the compliance of data against the process models, so-called *conformance checking*, and enhance process models with constraints/information coming from the event logs, so-called *enhancement*. Such techniques have been adopted by various domains, including healthcare, manufacturing, and logistics. The first step of applying the techniques is to extract event logs from the target information systems, e.g., Enterprise Resource Planning (ERP) systems. This usually requires a connection to the database(s) supporting the information system. Afterward, the extracted event log undergoes pre-processing steps to resolve various data quality issues, including incomplete information, noise, etc. These steps are usually called ETL (Extraction, Transformation, and Load). The ETL phase is usually the most time-consuming part of a process mining project [12].

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 255–267, 2022.
https://doi.org/10.1007/978-3-030-98581-3_19

ERP systems contain valuable data based on which process mining techniques provide insights regarding the underlying real-life business processes. In particular, the SAP ERP system has a significant share in the ERP market (22.5% in 2017, Gartner). Extracting data from an SAP ERP system is particularly challenging as it involves many different tables/objects. Due to its complexity, support to extracting event data from the SAP ERP system has only been limited to commercial vendors, e.g., Celonis and ProcessGold, which requires extensive interaction with domain experts. Moreover, the logs extracted by such extractors suffer from convergence/divergence problems [1]. This is due to the necessity to specify a *case notion*. A case notion is a criteria to group events that belongs to the same execution of a business process. In ERP systems, different case notions can be used for the same data. For example, in a procure-to-pay process, we could specify as case notion the order, the single item of the order, the delivery, the invoice, or the payment.

This paper proposes a novel approach to guide and ease the extraction of event logs from SAP ERP. The approach consists of two phases, i.e., 1) building *graph of relations* and 2) extracting object-centric event logs. We propose to use Object-Centric Event Logs (OCEL) as intermediate storage to collect the events extracted from different tables. OCEL does not require the specification of a case notion. Therefore, it provides flexible and comprehensive event data extraction. OCEL can be used with Object-Centric Process Mining (OCPM) techniques or flattened to traditional event logs by selecting a case notion out of objects. The proposed approach has been implemented as a prototypical extractor and evaluated using an SAP ERP system.

The rest of the paper is organized as follows. Section 2 presents some background knowledge. Section 3 presents the proposed approach. Section 4 presents a prototypal software implementing the ideas proposed in this paper. Section 5 evaluates the processes extracted by the prototypal software on top of an educational SAP instance. Section 6 presents the related work on extracting and analyzing event logs from SAP.

2 Background

This section presents some background knowledge on OCEL, convergence/divergence problems, and SAP systems.

2.1 Object-Centric Event Logs

Traditional event logs in process mining have events associated with a single case/process execution. These event logs, extracted from information systems, suffer from convergence/divergence problems [1]. We have a *convergence* problem when the same event is duplicated among different instances. This happens, for example, in an order-to-cash process, when *item* is considered as the case notion, and an event of order creation can be associated with several items. We have a *divergence* problem when several instances of the same activity happen in a case while not being causally related. This happens, for example, in an order-to-cash process, when *order* is considered as the case notion, and several instances of the same item-related activity are contained in the same order.

OCEL relax the assumption that an event is associated with a single case. Instead, in an OCEL an event can be related to several objects, where every object is associated with a type. This results in a more natural way to extract event data from a modern information system. For example, in ERP systems, the event of order creation can currently involve an order document and several items. This resolves the convergence problem (since we do not need to duplicate the events anymore) and the divergence problems (since activities related to items of an order are not associated with the case of the general order).

Recently, the OCEL standard[1] has been proposed as the mainstream format for storing object-centric event logs [5]. The format is supported by different implementations and libraries in various programming languages, e.g., Java (ProM framework) and Python. OCEL can be used to discover object-centric process models [2,3], which describe the lifecycle of different object types and their interactions. Moreover, conformance checking can be done on multiple object types [2].

2.2 SAP: Entities and Relationships

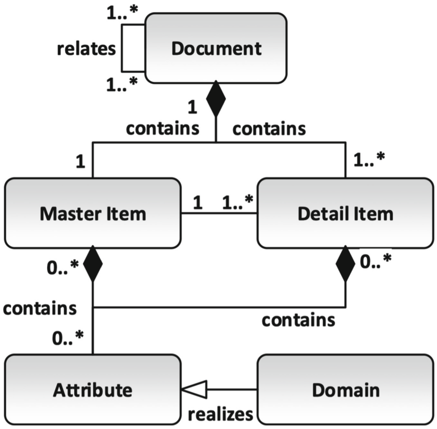


Fig. 1. Core entities of SAP ERP systems in UML 2.0 class diagram

In a broader sense, SAP ERP can be seen as a document management system. Therefore the concept of *document* is particularly important. Figure 1 introduces the document and its relevant entities and relationships among them, using UML 2.0 class diagram. First, a document represents a core business object, including

[1] http://www.ocel-standard.org/.

orders, deliveries, and payments. Each document contains a *master item* and *detail items*. For instance, a delivery document contains a delivery master item, corresponding to an order, and multiple delivery detail items, corresponding to materials in the order. A *master table* is a collection of the same type of master items, whereas a *detail table* is a collection of the same type of detail items. For instance, *EKKO* as a master table contains purchase order master items. *EKPO* as a detail table contains purchase orders detail items.

Both master and detail items contain a various number of *attribute values*, e.g., the total cost of a document or the cost of a single item. Each attribute belongs to a *domain* that encodes the type of information reported by the attribute, e.g., creation date and posting date of a document share the same domain because they are both dates.

3 Extracting Event Data from SAP ERP: Approach

Figure 2 describes an overview of our proposed approach to extract OCEL from SAP ERP systems. It consists of two phases: 1) building *Graph of Relations (GoR)* and 2) extracting OCEL. The former aims to construct a graph that describes all relevant tables of a business process. There are well-known business processes in SAP ERP, e.g., Purchase to Pay (P2P) and Order to Cash (O2C). For such business processes, target tables, where we extract event data regarding the process, are already known, e.g., *EKKO*, *RBKP*, *EKBE* for P2P and *VBAK*, *BKPF* for O2C. However, most business processes in an organization are mostly unknown and, thus, require the identification of relevant tables.

Based on the GoR, we extract OCEL by connecting them to the underlying database of SAP ERP systems. To this end, we first preprocess records of tables described in the GoR. Next, we define activity concepts relevant to the target business process using the relevant tables. Finally, based on the activity concept, we extract event data from the relevant tables.

3.1 Building Graphs of Relations

Figure 3 shows the conceptual model of three GoRs, each of which corresponds to a business process. A GoR is an undirected connected graph where the nodes are SAP tables containing the potentially interesting information and the edges show a relation among two tables based on a joint field/column. The node in the center of a GoR is a master table that is most relevant to the target process. The distance of each node from the master table shows the relevancy of the information contained in the corresponding table to the tables of interest and consequently to the corresponding type of process. Different colors in a GoR indicate different classes of tables. Each class has a unique way of defining activity concepts. As a result, different GoRs may be connected to each other. Below are the steps to construct GoRs:

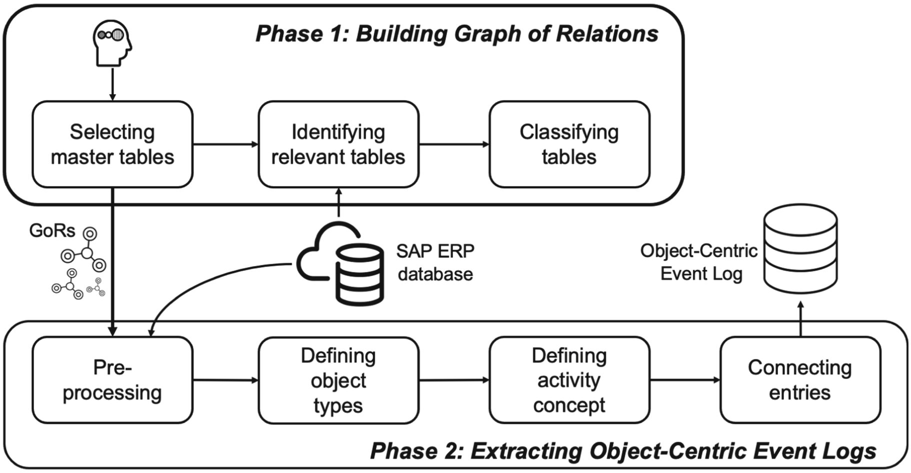
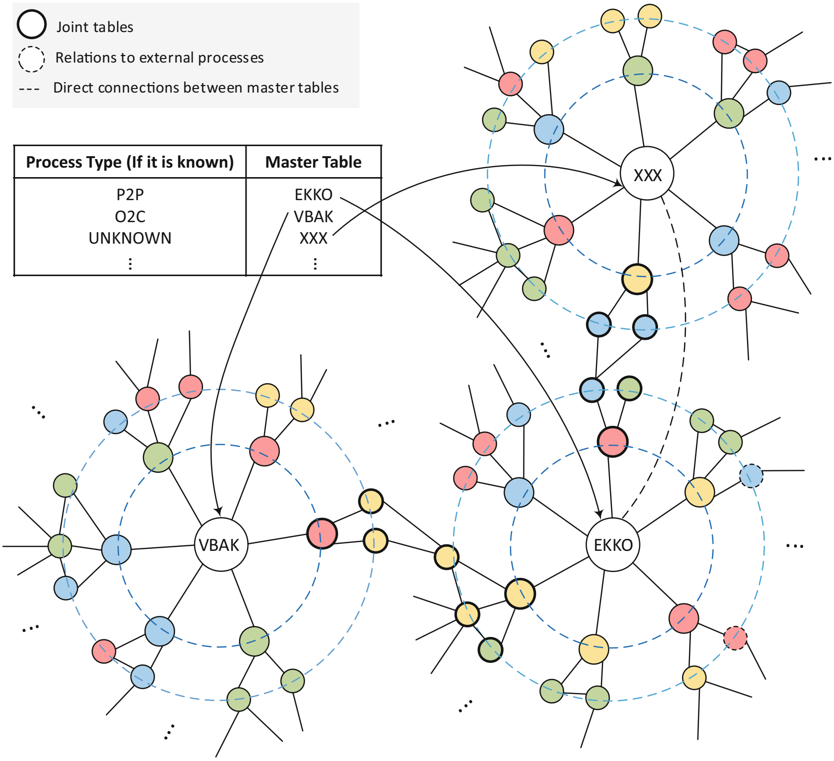


Fig. 2. Overview of extracting object-centric event logs from SAP ERP systems


Joint tables
Relations to external processes
--- Direct connections between master tables
VBAK
EKKO
XXX
...

Process Type (If it is known)	Master Table
P2P	EKKO
O2C	VBAK
UNKNOWN	XXX
⋮	⋮

Fig. 3. Conceptual model of Graph of Relations (GoRs)

Selecting Master Tables. A GoR is built upon a master table relevant to a business process to analyze. In this work, we consider relevant master tables as users' input.

Identifying Relevant Tables. Based on the given master table, we need to identify relevant tables to the master tables. Such tables become the candidates for constructing the GoR. Three different main approaches may be taken: *manual*, *automatic*, and *hybrid.*

- In the manual approach, the identification is conducted by domain experts who understand business processes and the technical details of SAP systems. In addition, the domain expert may provide a data schema to explain the entities and relationships among them.
- In the automatic approach, the identification is made automatically by exploiting existing information in the system. For instance, using the table *DD03VV*, one can extract the relationships between the tables.
- Finally, the hybrid approach exploits both manual and automated techniques. For instance, the data schema from domain experts can provide an initial set of relevant tables, which will be improved by including more relevant tables with the help of automatically generated relationships.

Classifying Tables. The last step is the classification of the identified tables into different classes. In the following, we describe five different classes.

- A *flow table* describes the status of objects that compose the target business process. It explains the creation, deletion, and update of such objects, e.g., VBFA explains the status of objects that are associated with the Order-to-Cash (O2C) process.
- A *transaction table* describes the execution of transactions (TCODE) in SAP systems.
- A *change table* describes the changes in objects of the target business process, e.g., CDHDR and CDPOS are primary change tables.
- A *record table* stores relevant attributes of objects of the target business process, e.g., the table EKKO contains the relevant attributes of purchase order documents.
- A *detail table* stores the relationships between different entities, e.g., the table EKPO stores the connection between purchase requisitions and purchase orders.

3.2 Extracting Object-Centric Event Logs

In this subsection, we explain how OCEL are extracted using GoRs. The extraction consists of four main steps; *pre-processing*, *defining activity concept*, *defining object types* and *connecting entries.*

Pre-processing. SAP tables contain a lot of data related to different companies or groups in the same company (multi-tenant system). Moreover, when invoicing/accounting tables are considered, documents are organized by their fiscal year. A pre-processing step must be performed to extract an event log of reasonable size, containing the desired behavior and a coherent set of information since document identifiers can be replicated across different organizations. To this end, the union of all the fields in the primary keys of the tables is considered, and for some of them, a filtering query is executed, e.g., on a specific company code or a specific fiscal year.

Defining Object Types. During the extraction, the entries of the master tables are transformed into events, having the columns as event attributes. Moreover, the values of all the columns except the dates and the numbers become objects of the object type given by the column's name.

Defining Activity Concept. To extract event data from GoRs, we take a divide-and-conquer approach. We first extract event data from each table and then combine them. The first step of extracting event data from each table is to define the activity concept. In the following, we explain how the activity is defined in each class of tables.

- Each row in flow tables contains a current document number, a previous document number, the type of the current document, and the type of the previous document. For instance, considering VBELN as the domain, VBELN, VBELV, VBTYP_N, and VBTYP_V in the VBFA table contain respectively the current document number, the previous document number and the current and previous document types. We define activities as the type of the current documents, i.e., the value in VBTYP_N.
- Each row in transaction tables contains a transaction code. We transform the transaction code into human-readable formats using the TSTCT table, e.g., VA02 is transformed to Change Order, which becomes the activity name.
- Each row in record tables describes the properties of an object. All the rows of the record tables are associated with the same activity, e.g., *Create document [...]* for all the rows in EKKO.
- For change tables, we suggest three approaches: (1) Transaction codes used for changes are transformed into activities, (2) Fields, updated after changes, are converted into activities, e.g., *Price Changed*, and (3) We consider both old and new fields' values and define activities, e.g., *Postpone Delivery*, by comparing old and new values of delivery dates.

Connecting Entries. In this step, the information of the detail tables is used to enrich events. For example, if an entry of the table RSEG, containing detailed information about invoices, associates an invoice identifier with an order identifier, every event associated with the invoice identifier is also associated with the order identifier in the subsequent step.

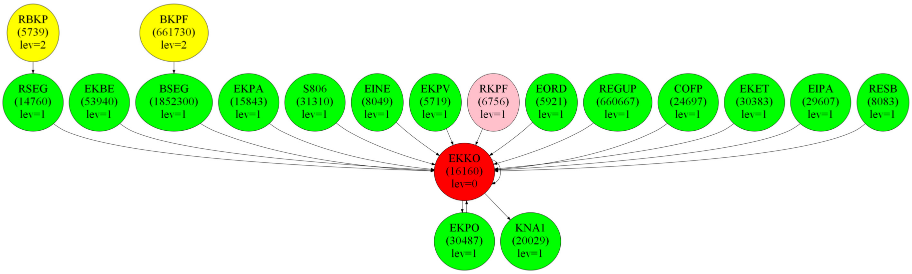


Fig. 4. A GoR built on our SAP IDES instance on the P2P process. Detail tables are colored by green, RKPF that is an additional record tables is colored by pink, and RBKP and BKPF that are additional transaction tables are colored by yellow. (Color figure online)

4 Extracting Event Data from SAP ERP: Tool

We implemented a tool in the Python3 language, available in the Github repository; https://github.com/Javert899/sap-extractor. The tool is available as a web application implemented using the Flask framework and can be launched with the command; *python main.py*. The web application can be accessed at the address; http://localhost:5000/new_extractor.html. First, the extractor asks the parameters of connection to the database supporting the SAP ERP instance. Then, it provides both a list of object classes contained in the database and a list of pre-configured sets of tables related to the mainstream processes. The next step is the construction of the GoR, which permits extending the set of tables. The following step is about pre-providing the values for the primary keys of the included tables, e.g., the client used during the connection and the fiscal year. After this step, the identification of the type of tables and the extraction occurs, which permits obtaining an OCEL, that can be flattened to a traditional event log or analyzed using object-centric techniques such as the ones provided in https://github.com/Javert899/sap-extractor.

5 Assessment

This section proposes an assessment of the proposed techniques on top of an SAP ERP IDES system. In particular, we will target the extraction of the well-known Purchase to Pay (P2P) system. A P2P process involves different steps including *approval of a purchase requisition*, *placement of a purchase order*, *invoicing from a supplier*, and *payment*. Therefore, it involves different tables in the SAP system.

5.1 Building a Graph of Relations

Selecting Master Tables. The first step in the tool is selecting a candidate table related to the process. In this case, we start from EKKO that is one of the main tables in the P2P process and contains the master information. In building the GoR, represented in Fig. 4, several other tables that are connected to EKKO are found. Given the vast number of tables contained in SAP, we applied a simple filtering based on the number of entries in each table to show the main nodes in the GoR.

Identifying Relevant Tables. Figure 4 shows other tables containing event data meaningful to extract an event log for the P2P[2]. The user needs to specify the tables to include along with the original set of tables. The GoR is therefore updated[3]. In our implementation, the master tables related to the detail tables are automatically included in the set[4].

Classifying Tables. The tool needs to categorize the tables in the set between master tables and detail tables, as the master tables contain event data, while detail tables contain the connection between different entities:

- Some tables are recognized as transactions tables: *RBKP* (containing the transactions related to the invoices) and *BKPF* (containing the transaction related to the payments).
- Some tables are recognized as record tables: *EBAN* (in which a record is a purchase requisition), *EKKO* (in which a record is an order document), and *RKPF* (in which a record is a reservation).
- Some tables are recognised as detail tables: *EKPO*, *EKPA*, *EKET*, *EKBE*, *BSEG*, *RSEG*, *RESB*[5].

5.2 Extracting Object-Centric Event Logs

In this section, we will explain the main steps of the log extraction process, including the definition of the object types and the activity concept for the extraction, and the connection between the entries given the information of the detail tables. Since we did not perform a pre-processing step, we will not assess the step here.

[2] Including EKBE, containing goods/invoice receipts, BSEG, containing detail table for payments, RSEG, containing detail data for invoices, RKPF, including inventory management data, EKPO, containing the detailed information about the purchase orders, EKPA, containing the partner roles in purchasing, and EKET, containing the scheduling agreement. We can see that the EBAN table, containing purchase requisition data, has not been included because of the filtering applied on the number of entries. However, it would be found by the method if the threshold is set to a lower value so that we will include it in the following steps.

[3] The set of tables to extract include: EKKO, EKPO, EKPA, EKET, EKBE, BSEG, BKPF, RSEG, RBKP, RKPF, RESB, EBAN.

[4] This means that BKPF, the master table of BSEG, containing the master data about the payments, and RBKP, the master table of RSEG, containing the master data about the invoices, are included.

[5] Because their primary key is contained in the primary key of *EKKO* (for EKPO, EKPA, EKET, EKBE), *BKPF*, *RBKP* and *RKPF* respectively.

Defining Object Types. Starting from the choices on the GoR and the identification of the type of tables, it is possible to extract different object types, including *BANFN-BANFN* (purchase requisition), *INFNR-INFNR* (purchasing record), *EBELN-EBELN* (purchase order), *BELNR-RE_BELRN* (the invoice number), *BELNR-BELNR_D* (the payment number), and *AWKEY-AWKEY* (a generic object type containing the ID of the object in SAP).

Defining Activity Concept. The activity concept is defined as follows:

- For the record tables, a unique activity is defined for all the events, that is *Create document (TABNAME)* (where TABNAME is the name of the corresponding record table, so it can be EBAN/EKKO/RKPF).
- For the transaction tables, the activity is given by the transaction code[6]. Mainstream transactions occurring are *Enter incoming invoice*, *Enter incoming payment*, *Enter outgoing payment.*

Connecting Entries. The detail tables are used to enrich the entries extracted from the master tables as follows:

- *BSEG* provides a connection from the payments to the purchase order items.
- *RSEG* connects the invoices to the purchase order items.
- *EKPO* provides a connection of the purchase order items to the corresponding purchase requisition.
- *EKPA* and *EKET* contain detailed information that does not provide meaningful links to other tables in the set. *EKBE* is a peculiar type of detail table, as it contains the information about goods/invoice receipts, so it could be seen as a master table. Still, it also links the purchase order items with the invoices through the goods/invoice receipts.

6 Related Work

This section presents the related work on data extraction from ERP systems for process mining purposes.

Data Extraction and Pre-processing from SAP ERP. In [6], an approach to extract traditional event logs from SAP ERP is proposed. The set of relevant business objects is identified, and the related tables and their relations are identified. A limitation is that the construction of the document flow is manual. In [7], the authors address the pre-processing challenges to extract event logs from SAP ERP by using tools such as EVS Model Builder. In [4], an ontology-driven approach for the extraction of event logs from relational databases is proposed, in which the user can express semantic queries which are then translated to relational queries. In [8], the effects of some decisions on the quality of the resulting event log are analyzed. In particular, the context of event log extraction from ERP system is considered.

[6] Using the description of the transaction contained in the table TSTCT.

Artifact-Centric Models on ERP Systems. In [10], an approach to discover artifact-centric models from ERP systems is proposed. The approach is split into two main parts: 1) identifying a set of artifacts, extracting a traditional event log, and a model of its lifecycle; 2) discovering the interactions between artifacts. The set of tables to extract needs to be decided by the user and the specification of the activity concepts is not described in this work.

In [9], object-centric event logs (in the XOC format) are extracted from the Dollibar ERP system. These logs have been used to generate an object-centric behavioral constraints (OCBC) model. However, OCBC/XOC are not scalable.

OpenSLEX Meta-models. In [11], a meta-model is proposed to ease the extraction of process mining event logs from information systems supported by relational databases. The instances of the OpenSLEX meta-model can be built from different types of database logs (redo logs, SAP change tables). Hence, the meta-model is generic and not tailored to the peculiar features of an SAP ERP system. The main problem is that the extraction of an event log requires a case notion's specification, which leads to convergence/divergence problems.

Enterprise-Grade Connectors. Several commercial vendors of process mining solutions offer enterprise-grade connectors to SAP, that are able to ingest and process millions of events. Notable examples in the current landscape are Celonis[7], Signavio[8], LANA[9], UIPath[10].

7 Conclusion

In this paper, we proposed a generic approach to extract event logs from SAP ERP, which exploits the relationships between tables in SAP to build Graphs of Relations (GoRs) and obtains Object-Centric Event Logs (OCEL) using GoRs. Figure 2 summarizes our approach. By storing extracted event data into OCEL, we permit the specification of multiple case notions, avoiding the convergence/-divergence problems and simplifying the extraction process. An open-source tool implementing the approach and a case study on an educational SAP instance have been presented, showing the feasibility of identifying the relationships between different tables of the P2P process and extracting corresponding OCEL. As future work, we plan to deploy our approach on different instances of SAP systems running in real businesses to explore the connection between GoRs and underlying processes and to discover unknown processes. Moreover, we should further assess how good the extraction of a typical SAP process is in comparison to commercial-grade extractors.

[7] https://www.celonis.com/solutions/systems/sap.

[8] https://www.signavio.com/products/process-intelligence/.

[9] https://lanalabs.com/en/migration-to-sap-s-4-hana-with-lana/.

[10] https://docs.uipath.com/process-mining/docs/introduction-to-sap-connector.

References

1. Aalst, W.M.P.: Object-centric process mining: dealing with divergence and convergence in event data. In: Ölveczky, P.C., Salaün, G. (eds.) SEFM 2019. LNCS, vol. 11724, pp. 3–25. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30446-1_1
2. van der Aalst, W.M., Berti, A.: Discovering object-centric Petri nets. Fundam. Inform. **175**(1–4), 1–40 (2020)
3. Berti, A., van der Aalst, W.: Extracting multiple viewpoint models from relational databases. In: Ceravolo, P., van Keulen, M., Gómez-López, M.T. (eds.) SIMPDA 2018-2019. LNBIP, vol. 379, pp. 24–51. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-46633-6_2
4. Calvanese, D., Montali, M., Syamsiyah, A., van der Aalst, W.M.P.: Ontology-driven extraction of event logs from relational databases. In: Reichert, M., Reijers, H.A. (eds.) BPM 2015. LNBIP, vol. 256, pp. 140–153. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-42887-1_12
5. Ghahfarokhi, A.F., Park, G., Berti, A., van der Aalst, W.M.P.: OCEL: a standard for object-centric event logs. In: Bellatreche, L., et al. (eds.) ADBIS 2021. CCIS, vol. 1450, pp. 169–175. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-85082-1_16
6. van Giessel, M.: Process Mining in SAP R/3: A Method for Applying Process Mining to SAP R/3. Eindhoven University of Technology, Eindhoven, The Netherlands (2004)
7. Ingvaldsen, J.E., Gulla, J.A.: Preprocessing support for large scale process mining of SAP transactions. In: ter Hofstede, A., Benatallah, B., Paik, H.-Y. (eds.) BPM 2007. LNCS, vol. 4928, pp. 30–41. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78238-4_5
8. Jans, M., Soffer, P.: From relational database to event log: decisions with quality impact. In: Teniente, E., Weidlich, M. (eds.) BPM 2017. LNBIP, vol. 308, pp. 588–599. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-74030-0_46
9. Li, G., de Murillas, E.G.L., de Carvalho, R.M., van der Aalst, W.M.P.: Extracting object-centric event logs to support process mining on databases. In: Mendling, J., Mouratidis, H. (eds.) CAiSE 2018. LNBIP, vol. 317, pp. 182–199. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-92901-9_16
10. Lu, X., Nagelkerke, M., Van De Wiel, D., Fahland, D.: Discovering interacting artifacts from ERP systems. IEEE Trans. Serv. Comput. **8**(6), 861–873 (2015)
11. de Murillas, E.G.L., van der Aalst, W.M.P., Reijers, H.A.: Process mining on databases: unearthing historical data from redo logs. In: Motahari-Nezhad, H.R., Recker, J., Weidlich, M. (eds.) BPM 2015. LNCS, vol. 9253, pp. 367–385. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23063-4_25
12. van Eck, M.L., Lu, X., Leemans, S.J.J., van der Aalst, W.M.P.: PM^2: a process mining project methodology. In: Zdravkovic, J., Kirikova, M., Johannesson, P. (eds.) CAiSE 2015. LNCS, vol. 9097, pp. 297–313. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19069-3_19

Towards a Natural Language Conversational Interface for Process Mining

Luciana Barbieri[1](✉), Edmundo Roberto Mauro Madeira[1], Kleber Stroeh[2], and Wil M. P. van der Aalst[3,4]

[1] Institute of Computing, University of Campinas (Unicamp), Campinas, Brazil
{luciana.barbieri,edmundo}@ic.unicamp.br
[2] Everflow Process Mining, Campinas, Brazil
kleber.stroeh@everflow.ai
[3] Fraunhofer Institute for Applied Information Technology FIT, Sankt Augustin, Germany
[4] RWTH Aachen University, Aachen, Germany
wvdaalst@pads.rwth-aachen.de

Abstract. Despite all the recent advances in process mining, making it accessible to non-technical users remains a challenge. In order to democratize this technology and make process mining ubiquitous, we propose a conversational interface that allows non-technical professionals to retrieve relevant information about their processes and operations by simply asking questions in their own language. In this work, we propose a reference architecture to support a conversational, process mining oriented interface to existing process mining tools. We combine classic natural language processing techniques (such as entity recognition and semantic parsing) with an abstract logical representation for process mining queries. We also provide a compilation of real natural language questions (aiming to form a dataset of that sort) and an implementation of the architecture that interfaces to an existing commercial tool: Everflow. Last but not least, we analyze the performance of this implementation and point out directions for future work.

Keywords: Process Mining · Process Querying · Natural Language Interface

1 Introduction

Process Mining (PM) aims to discover, monitor and enhance processes using information extracted from event logs [2]. There exist mature academic and commercial process mining techniques and tools that provide analyses over event log data. The use of these tools, however, requires knowledge of the technology itself and is mostly done by technical teams (process analysts, data scientists and alike).

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 268–280, 2022.
https://doi.org/10.1007/978-3-030-98581-3_20

To make process mining more ubiquitous, i.e., accessible on a daily basis by non-technical teams, we propose a natural language conversational interface. Business level and operations teams, for example, can take great benefit from the insights produced by process mining tools when accessed through such an intuitive conversational interface.

In spite of recent advances in Natural Language Processing (NLP), understanding the semantics of a natural language question and translating it to a correct corresponding logical query is still a challenging task. Problems such as ambiguity (same natural language expression having multiple interpretations) and variability (many different expressions having the same meaning) are yet difficult to handle. Context awareness brings yet another level of complexity to the task, as the meaning of a natural language question may depend on previous questions and responses.

The main objective of this ongoing research is to propose, implement and evaluate an architecture for a process mining natural language conversational interface that takes questions in natural language and translates them to logical queries that can be run against existing process mining tools. The contributions presented in this paper are:

- Introduce a reference architecture for a process mining natural language conversational interface
- Propose an abstract logical representation for process mining queries that is, on the one hand, independent of the underlying concrete process mining tool and, on the other, mappable to its API calls
- An initial collection and categorization of natural language process mining questions aiming to create a public dataset
- A proof of concept of the proposed architecture, including integration to a commercial tool (Everflow Process Mining[1]) through its (RESTful) API

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 introduces the proposed architecture. Section 4 describes the PM question dataset under construction. Section 5 presents the conducted proof of concept. Section 6 concludes this paper and points out future work and directions.

2 Related Work

Natural Language Interfaces to Databases. From the many existing NLP applications, the ones that are mostly related to this research are the so called Natural Language Interfaces to Databases (NLIDB). The main objective of NLIDB is to enable users who are not familiar with complex query languages such as SQL to easily formulate queries over information stored in databases using natural language.

Even though NLIDB is not a new research topic, recent advances in natural language processing have raised its importance and popularity during the last

[1] https://everflow.ai/.

decade [3]. Current methods differ in the use of natural language itself (from queries restrictedly written according to specific grammatical constraints to full natural language sentences), as well as in the technical approaches used to parse and convert them to a machine-readable format such as SQL or SPARQL. Most common parsing techniques are based on rule matching or machine learning. In either case, the types of queries that can be handled by the system are limited either by the set of rules, in the first case, or by the training data in the second.

While most of the existing NLIDB methods are designed to handle queries over any domain (metadata and/or domain ontologies are usually taken as input to map domain terminology to database entities), using specific process mining domain knowledge yields context to the design of a potentially more robust natural language interface.

Natural Language Processing Applications in Business Process Management and Process Mining. One of the most important applications of NLP techniques to the Business Process Management (BPM) domain is the extraction of process models from natural language text [4]. Other existing applications of NLP to BPM include the automatic generation of textual descriptions from process models [6] and the comparison of process models to textual descriptions [9]. In [1], the authors discuss future challenges for NLP applications in the BPM field, including the use of conversational systems to support the execution of business processes.

Most related to our research is the work presented in [5], where the authors propose a method to answer natural language queries over process automation event logs. The method extends the ATHENA NLIDB system [8] to translate natural language queries to queries over process execution data (event logs) stored in Elasticsearch.

Existing process mining techniques and tools can provide automatic analysis over event log data, which can be used to answer high-level user questions. To the best of our knowledge, this is the first research work aiming to automatically understand and answer natural language questions over process mining data and analyses.

3 Proposed Method

Our proposed method can be best described by the architecture depicted in Fig. 1. In broad terms, it can be viewed as a pipeline moving from top to bottom. The input is a question in regular natural language (in our case, English). Questions can be provided as text or speech - planned future work includes an Automatic Speech Recognition module, which will provide “speech-to-text” functionality.

To close the pipeline, we envision Response Generation and Text to Speech modules to provide a conversational response to the user. In the scope of this work, this response was simplified and corresponds to a piece of information directly derived from the call to the PM Tool’s API. The following sections detail the modules responsible for understanding the input natural language question and mapping it to an API call.

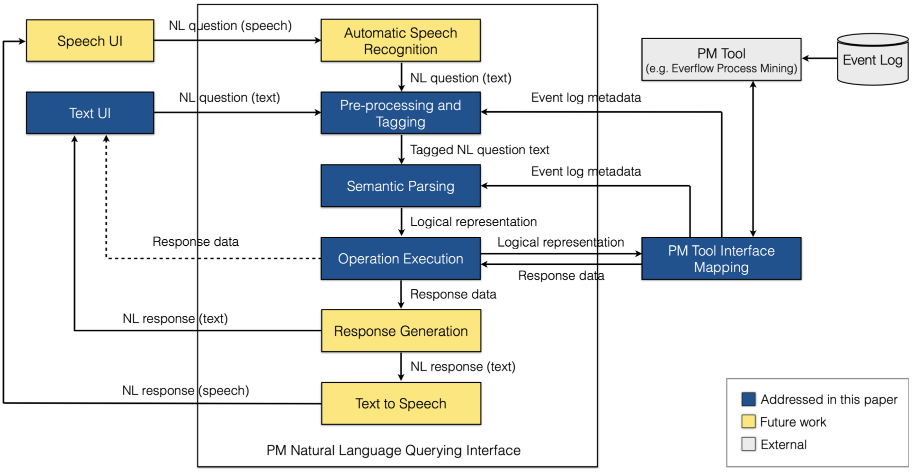


Fig. 1. Process Mining Natural Language Querying Interface Architecture Overview

3.1 Pre-processing and Tagging

The input text passes initially through a Pre-processing and Tagging step, where the following processing occurs:

- Tokenization, which is the splitting of text into tokens. Separation is based on whitespaces, punctuation marks and apostrophes, among others.
- Part-of-Speech (POS) Tagging, which performs morphological analysis over the text, marking tokens with tags such as PRON (pronoun), ADJ (adjective) and VERB.
- Dependency Parsing, which provides semi-syntactic analysis and marks tokens with grammatical structure information and dependency relations between them.
- Lemmatization, which finds the base (non-inflected) form of words.
- Entity Recognition, which identifies and tags real-world entities in the text, as detailed below.

Entity Recognition identifies general entities from pre-defined categories, such as names of people and organizations, geographic locations, time and quantities, among others. In addition to that, a natural language interface for process mining must be able to recognize the process mining entities present in sentences. Terms such as event, case, activity, resource and variant (along with its synonyms) must be recognized and tagged appropriately. The resulting tags are a crucial input for the next task in the processing pipeline (semantic parsing). Figure 2 depicts the process mining data model that underlies the recognition of such terms. Although this model is based in [2], one should notice that, for the purpose of this work, the term “event refers to both event and activity instance.

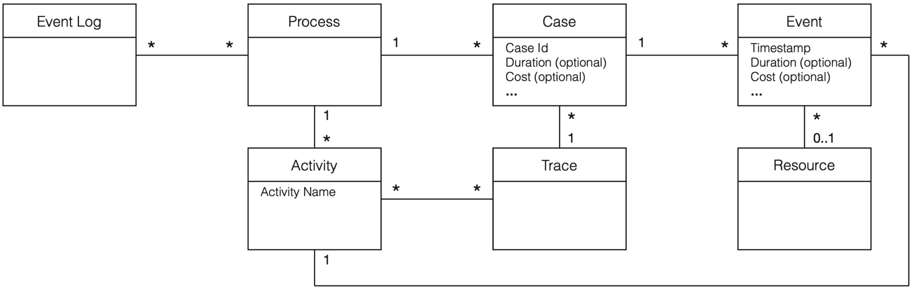
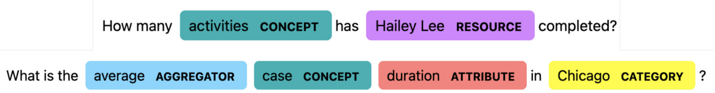


Fig. 2. Process Mining Data Model Underlying Entity Recognition

Besides dealing with general process mining terms, the system must be able to recognize domain-specific terms. This includes the names of non-standard attributes present in the event log along with possible categorical values, among others. To be able to recognize such terms, this module uses event log metadata (names, types and possible values) of these attributes. As the proposed natural language interface does not deal directly with the event log, the PM Tool Interface Mapping layer takes the responsibility of interfacing with the PM Tool to gather these metadata. Figure 3 shows examples of questions tagged with recognized entities. Notice that "Hailey Lee" and "Chicago" are categorical attribute values gathered from event log metadata and used to tag these terms during entity recognition.


How many activities CONCEPT has Hailey Lee RESOURCE completed?

What is the average AGGREGATOR case CONCEPT duration ATTRIBUTE in Chicago CATEGORY ?

Fig. 3. Entity Recognition Examples

3.2 Semantic Parsing

Semantic parsing aims to understand the meaning of a natural language sentence and map it to a logical (machine-readable) representation. The most common methods used for semantic parsing are rule-based and neural approaches. While rule-based methods are usually more appropriate to build domain specific systems targeted to understand a finite set of sentences, neural systems are more suitable to handle complex scenarios at the cost of requiring large training corpora. Logical representations usually take the form of lambda calculus, query languages such as SQL and SPARQL or executable programs, among others.

Rule Matching. As, to the best of our knowledge, there is no process mining question dataset that could be annotated and used to train traditional machine learning or neural models, we have initially adopted a rule matching approach for semantic parsing. Besides requiring no training data, the method has the advantage of achieving high accuracy in answering predictable questions.

In our proof of concept, we used the spaCy open-source natural language processing library[2]. Its Rule Matcher component allows the definition of rules that match sequences of tokens. Rules are based on tags filled in the previous steps in the pipeline (part-of-speech tags, dependency parsing results, entity recognition labels), together with actual words or expressions (in our case, words or expressions used to express the sort of process mining relationship/analysis being queried). Figure 4 illustrates the matching of the question "What activities have been assigned to Hailey Lee?" to a rule pattern.

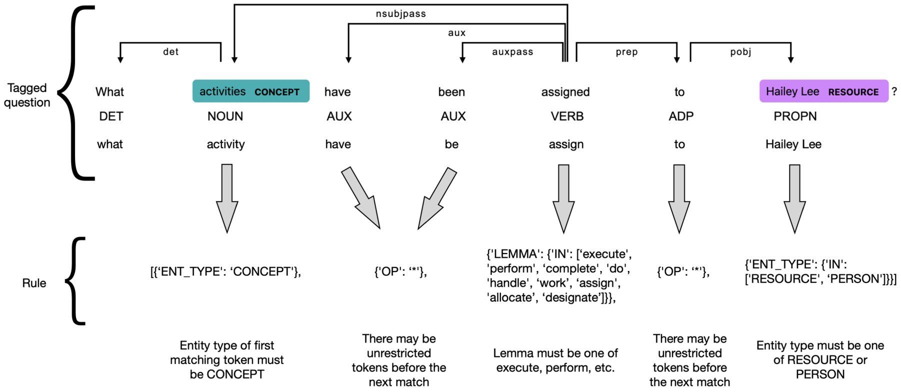

nsubjpass
aux
det
auxpass
prep
pobj
Tagged question
What
activities CONCEPT
have
been
assigned
to
Hailey Lee RESOURCE
?
DET
NOUN
AUX
AUX
VERB
ADP
PROPN
what
activity
have
be
assign
to
Hailey Lee
Rule
[{'ENT_TYPE': 'CONCEPT'},
{'OP': '*'},
{'LEMMA': {'IN': ['execute', 'perform', 'complete', 'do', 'handle', 'work', 'assign', 'allocate', 'designate']}},
{'OP': '*'},
{'ENT_TYPE': {'IN': ['RESOURCE', 'PERSON']}}]
Entity type of first matching token must be CONCEPT
There may be unrestricted tokens before the next match
Lemma must be one of execute, perform, etc.
There may be unrestricted tokens before the next match
Entity type must be one of RESOURCE or PERSON

Fig. 4. Rule matching example

In this case, the matched pattern leads the system to the conclusion that the user wants the list of activity instances associated to a particular resource (Hailey Lee).

Logical Representation. After the semantics of a question is understood (i.e. after it matches a rule), it must be converted to a corresponding logical (PM tool independent) representation. The Question Decomposition Meaning Representation (QDMR) proposed in [11] and inspired by SQL has been used for this purpose with some extensions.

In QDMR questions are represented by a sequence of steps where each step corresponds to an operator. Each operator (except for `select`) is applied to the results of a previous step in the sequence. Additional parameters may be given to logical operators depending on the entities (concepts, attributes, aggregations, etc.) recognized in the natural language question. Table 1 presents the most relevant QDMR operators used in this research work to compose the logical representation of PM queries. For the complete set, please refer to [11].

[2] https://spacy.io/.

Table 1. Some QDMR operators used for PM question logical representation

Operator	Description	Example	Logical Form
`select`	Return all instances of the given concept.	Show me all cases	`select case`

`filter`	Return the referenced instances for which the given condition holds.	Show me all cases from Chicago	`select case`
			`filter city Chicago #1`

`project`	Return the given attribute/relation for the referenced instances.	How long does each process instance take to execute?	`select case`
			`project duration #1`

`aggregate`	Apply the given aggregation to the referenced values.	What is the average case duration?	`select case`
			`project duration #1`
			`aggregate average #2`

`group`	Apply the given aggregation to each subset of values corresponding to each key	What is the average cost of each activity?	`select event`
			`project cost #1`
			`project activity #1`
			`group average #2 #3`

`superlative`	Return the referenced instances for which the given value is the highest/lowest.	What was the slowest case?	`select case`
			`project duration #1`
			`superlative max #1 #2`

Notice that hash tags are used to refer to the results of a previous logical operation in the sequence, which may be a set of event or case instances or their attribute values. For example, in the following sequence, `#1` refers to the results of `select case`, which are all case instances and `#2` refers to the values of the `duration` attribute for `#1`.

```
select case
project duration #1
aggregate average #2
```

The original set of QDMR operators was extended by this work to allow querying the behavioral aspects of process execution. Inspired by and initially based on the set of predicates defined by the Process Query Language (PQL) [7], the `predicate` operator was introduced to logically represent questions over behavioral relations between executed activities. Supported predicates can be applied over cases or traces and are presented in Table 2.

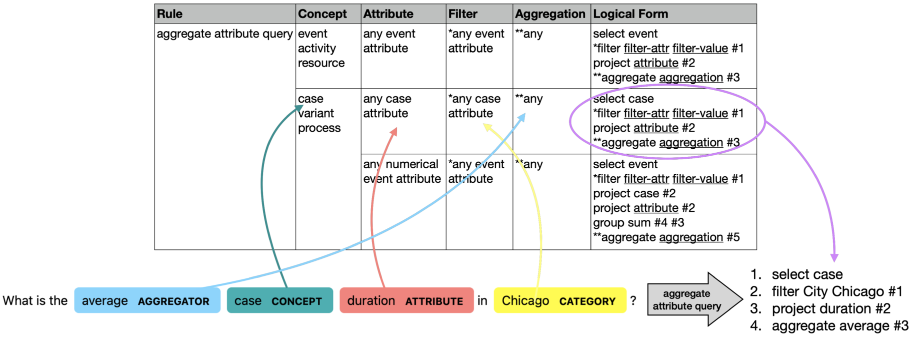
Rule to Logical Representation Mapping. As one of the architectural goals of the proposed method is to allow integration to any Process Mining tool, it makes as few assumptions as possible on how the integrated Process Mining tool models the event log data. As a result, a minimal process mining data model based in the XES standard event log format [10] drives the mapping of matched rules to logical representation. Some of the entities tagged and handled as concepts during entity recognition and rule matching (activity, resource, trace) are, at this point, mapped to attributes of event and case, which are the only

Table 2. Predicates used for PM question logical representation

Predicate	**Parameters**	**Description**
occurs	activity	Return the referenced cases or traces that execute the given activity.
cooccur	activity1, activity2	Return the referenced cases or traces that execute both activity1 and activity2 or none.
conflict	activity1, activity2	Return the referenced cases or traces that execute either activity1, activity2 or none.
causal	activity1, activity2	Return the referenced cases or traces where any occurrence of activity1 precedes any occurrence of activity2.
concurrent	activity1, activity2	Return the referenced cases or traces where some occurrence of activity1 occurs at the same time as some occurrence of activity2.
activity-count	-	Return the number of activities executed by each referenced case or trace, including repetitions.
distinct-activity-count	-	Return the number of distinct activities executed by each referenced case or trace.
occurence-count	activity	Return the number of times the given activity is executed for each referenced case or trace.

selectable concepts (assuming that processes are queried one at a time). Non-standard attributes contained in the event log are mapped based on the metadata obtained from the PM tool.

Once a rule fires, a corresponding logical representation must be put together. This depends not only on what rule has been matched, but also on the entities (concepts, attributes, etc.) recognized in the sentence. As an example, Fig. 5 depicts the possible logical representations to be created when the “aggregate attribute query” rule is matched.



Rule	Concept	Attribute	Filter	Aggregation	Logical Form
aggregate attribute query	event activity resource	any event attribute	*any event attribute	**any	select event *filter filter-attr filter-value #1 project attribute #2 **aggregate aggregation #3
	case variant process	any case attribute	*any case attribute	**any	select case *filter filter-attr filter-value #1 project attribute #2 **aggregate aggregation #3
		any numerical event attribute	*any event attribute	**any	select event *filter filter-attr filter-value #1 project case #2 project attribute #2 group sum #4 #3 **aggregate aggregation #5

1. select case
2. filter City Chicago #1
3. project duration #2
4. aggregate average #3

Fig. 5. Logical forms for attribute query rules

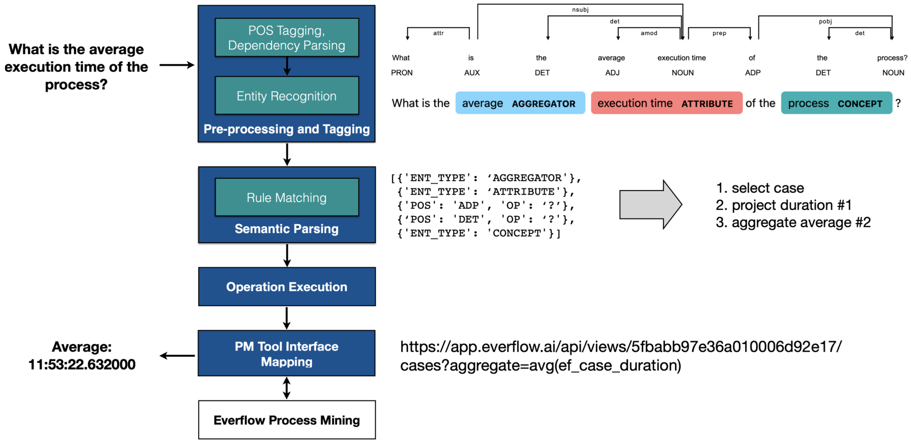
The matched rule indexes the first column in the table, while the entities tagged in the sentence index the next four (concept, attribute, filter and aggregation). The last column corresponds to the logical representation that will be used to drive the calls to the PM Tool API detailed in the following subsection. Asterisks indicate optional entities and the corresponding logical operations that are added to the sequence when they are present. The complete set of correspondences between rules and logical representations is available at https://ic.unicamp.br/~luciana.barbieri/ruletological.pdf.

3.3 PM Tool Interface Mapping

The final step in the question processing pipeline is to map the logical representation of the query into a real API call provided by a process mining tool.

In this work, we integrated the architecture into Everflow's RESTful API. This API presents endpoints that mimic process mining main concepts and naturally maps into the PM data model used to create logical representations.

Using Everflow's "cases" and "events" endpoints, altogether with their associated parameters (such as "filter" and "aggregate"), it is straightforward to map the logical representation into actual API calls. Figure 6 illustrates the end-to-end mapping of a natural language question to a final API call.



```
[{'ENT_TYPE': 'AGGREGATOR'},
 {'ENT_TYPE': 'ATTRIBUTE'},
 {'POS': 'ADP', 'OP': '?'},
 {'POS': 'DET', 'OP': '?'},
 {'ENT_TYPE': 'CONCEPT'}]
```

1. select case
2. project duration #1
3. aggregate average #2

https://app.everflow.ai/api/views/5fbabb97e36a010006d92e17/cases?aggregate=avg(ef_case_duration)

Fig. 6. End to end mapping of question to API call

The integration of a new PM tool currently requires a different instantiation of the PM Tool Interface Mapping component. Planned future work includes the definition of a standard API to replace this component and allow PM tools to easily integrate our natural language conversational interface.

4 Sample Questions

An initial set of natural language questions was collected from graduate students with beginner to intermediate level of expertise in Process Mining, resulting in

250 general (not specific to any existing event log) questions originally written in Portuguese. Free translation was performed by 3 volunteers resulting in 794 questions in English (multiple translations were done by the volunteers for some of the questions).

Questions were then categorized into 4 groups: event log data questions (questions over case/event instances, attributes and counts), process model questions (model structure, behavior and quality), process mining analysis questions (conformance checking, bottleneck analysis, transitions, root cause analysis, social network, etc.) and advanced analysis questions (prediction, prescription and simulation). Table 3 summarizes these categories.

Variability in both language and contents of questions can be improved in the future by collecting them directly in English and eliminating the translation step. Nonetheless, as no public dataset of PM questions is currently available, the samples collected so far played an important role in setting the ground for this research and being the initial input for building the rules used for semantic parsing. The complete set of 794 questions, with their corresponding classifications, is available at https://ic.unicamp.br/~luciana.barbieri/pmquestions.csv.

Table 3. Question categories

Category	# Samples	Example
Event log data	327	Which activity has the highest mean resolution time?
Process model	107	What are the possible start and end activities in my log?
Analysis	240	What are the most frequent non-conformances in my process?
Advanced analysis	120	What is the predicted completion time for case X?

5 Proof of Concept

In order to verify the applicability of the proposed method, we implemented a subset of the architecture (blue colored components) presented in Fig. 1. For this proof of concept, we used the spaCy open-source natural language processing library, targeting questions from the "Event log data" category of the original collected set. The library's Rule Matcher component was used and fed with 34 semantic rules covering event log attribute querying, instance querying and counting, aggregations and superlatives ("most", "least"), among others.

In order to test the implementation, the set of questions from the "Event log data" category was further refined by removing compound questions (questions containing multiple embedded subquestions) and time-bound questions (questions containing time-related filters as in "What is the average duration of cases completed in the first quarter of 2020?"), as these constructions were not covered by the implemented set of semantic rules. This led to a testing dataset of 203 questions.

This testing set was executed against a Work Force Management based event log that was uploaded into the Everflow Process Mining tool. However, any process mining event log could be used, as the collected questions are not context-specific (not bounded to any particular event log).

From the 203 testing questions, 163 (80.3%) were correctly answered, 22 (10.8%) were not responded because the system was not able to match any semantic rule, and 18 (8.9%) were incorrectly answered, because they fired the wrong semantic rule or because they were falsely tagged during the Pre-processing and Tagging phase.

Examples of successfully answered questions are "What is the most common flow of activities?", "Which activity has the highest mean resolution time?" and "What are the 10% slower cases?" Unmatched questions include "What resources execute what activities?" and "Which resources are the most agile in task execution?". Likewise, examples of questions that fired the wrong semantic rule are "What resources take the longest to execute activity A?" and "What are the resources assigned to the fastest cases?". Actually, all these failed tests illustrate the shortcomings of a rule-based approach, where the final result is sensitively connected to the rules in use. This means that they could be fixed by a more crafted, possibly longer, rule set, which is hard to achieve and difficult to maintain.

On the other hand, properly answered questions such as "What is the average execution time of the process in Chicago?" illustrate the ability of the system to use terms that are specific to the event log. In this example, "Chicago" is a value under the attribute "City" in the event log, and could be used in the question due to the capacity to handle metadata coming from the process mining tool. This question was, of course, not present in the original testing dataset.

Overall, in spite of the limited size and variation of the testing questions and rules, the 80.3% accuracy seems promising as a first result. As expected, a rule-based approach has limitations in treating questions that stray too much away from the structures implemented in the rules. In general, this method presents high precision, but low generalization.

6 Conclusions and Future Work

Implementing the proposed reference architecture and testing it against the aforementioned sample question dataset has led to some interesting conclusions. Rule-based semantic parsing was an appropriate choice for bootstrapping a natural language interface for PM as no training data set of any kind or size is currently available to train any supervised or semi-supervised machine learning technique.

Furthermore, as the PM general ontology is small (few entities and relations), it was possible to answer questions for a selected, pre-defined, set with high accuracy using a relatively small number of rules. However, this approach does come with limitations. Rule-based semantic parsing does not generalize well, with new rules being required for most new/unpredicted questions.

In order to overcome this generalization limitation and to evolve the study towards a fully functional architecture, we envision the following future work:

- Use machine learning for semantic parsing by using the developed rule matching parser to create an annotated training dataset.
- Increase our experiment by working with questions in other categories (process model, analysis, advanced analysis).
- Extend the response mechanism to include natural language response generation, making responses more natural and user-friendly.
- Extend the training dataset and make it public. This implies collecting additional questions, if possible, directly in English and associated with a selected event log, so that questions can be more context-based and closer to what real business users would ask in a specific domain.

Acknowledgements. We would like to thank Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, for providing the financial support for this work. Likewise, we would like to thank Professors Sarajane Marques Peres and Marcelo Fantinato for helping us collect questions from their students for our dataset.

References

1. van der Aa, H., Carmona Vargas, J., Leopold, H., Mendling, J., Padró, L.: Challenges and opportunities of applying natural language processing in business process management. In: International Conference on Computational Linguistics, pp. 2791–2801. Association for Computational Linguistics (2018)
2. van der Aalst, W.M.P.: Process Mining: Data Science in Action. Springer, Cham (2016). https://doi.org/10.1007/978-3-662-49851-4
3. Affolter, K., Stockinger, K., Bernstein, A.: A comparative survey of recent natural language interfaces for databases. VLDB J. **28**(5), 793–819 (2019). https://doi.org/10.1007/s00778-019-00567-8
4. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 482–496. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21640-4_36
5. Han, X., Hu, L., Sen, J., Dang, Y., Gao, B., Isahagian, V., Lei, C., Efthymiou, V., Özcan, F., Quamar, A., Huang, Z., Muthusamy, V.: Bootstrapping natural language querying on process automation data. In: 2020 IEEE International Conference on Services Computing (SCC), pp. 170–177 (2020)
6. Leopold, H., Mendling, J., Polyvyanyy, A.: Generating natural language texts from business process models. In: Ralyté, J., Franch, X., Brinkkemper, S., Wrycza, S. (eds.) CAiSE 2012. LNCS, vol. 7328, pp. 64–79. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31095-9_5
7. Polyvyanyy, A., ter Hofstede, A.H., La Rosa, M., Ouyang, C., Pika, A.: Process query language: Design, implementation, and evaluation (2019). arXiv preprint: arXiv:1909.09543
8. Saha, D., Floratou, A., Sankaranarayanan, K., Minhas, U.F., Mittal, A.R., Özcan, F.: Athena: an ontology-driven system for natural language querying over relational data stores. Proc. VLDB Endow. **9**(12), 1209–1220 (2016)

9. Sànchez-Ferreres, J., Carmona, J., Padró, L.: aligning textual and graphical descriptions of processes through ILP techniques. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 413–427. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_26

10. Verbeek, H.M.W., Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: XES, XESame, and ProM 6. In: Soffer, P., Proper, E. (eds.) CAiSE Forum 2010. LNBIP, vol. 72, pp. 60–75. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-17722-4_5

11. Wolfson, T., Geva, M., Gupta, A., Gardner, M., Goldberg, Y., Deutch, D., Berant, J.: Break it down: a question understanding benchmark. Trans. Assoc. Comput. Linguist. **8**, 183–198 (2020)

On the Performance Analysis of the Adversarial System Variant Approximation Method to Quantify Process Model Generalization

Julian Theis, Ilia Mokhtarian, and Houshang Darabi(✉)

Department of Mechanical and Industrial Engineering, University of Illinois
at Chicago, 842 West Taylor Street, Chicago, IL 60607, USA
{jtheis3,imokht2,hdarabi}@uic.edu

Abstract. Process mining algorithms discover a process model from an event log. The resulting process model is supposed to describe all possible event sequences of the underlying system. Generalization is a process model quality dimension of interest. A generalization metric should quantify the extent to which a process model represents the observed event sequences contained in the event log and the unobserved event sequences of the system. Most of the available metrics in the literature cannot properly quantify the generalization of a process model. A recently published method called Adversarial System Variant Approximation leverages Generative Adversarial Networks to approximate the underlying event sequence distribution of a system from an event log. While this method demonstrated performance gains over existing methods in measuring the generalization of process models, its experimental evaluations have been performed under ideal conditions. This paper experimentally investigates the performance of Adversarial System Variant Approximation under non-ideal conditions such as biased and limited event logs. Moreover, experiments are performed to investigate the originally proposed sampling parameter value of the method on its performance to measure the generalization. The results confirm the need to raise awareness about the working conditions of the Adversarial System Variant Approximation method and serve to initiate future research directions.

Keywords: Process Mining · Conformance Checking · Generalization · Generative Adversarial Networks

1 Introduction

Significant research effort has been spent on the automated discovery of process models from event logs and the quality assessment of such models, i.e., *conformance checking*. While the focus of conformance checking has been mainly on measuring how well a discovered process model reflects event sequences that are recorded in an event log, measuring the extent to which a process model

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 281–293, 2022.
https://doi.org/10.1007/978-3-030-98581-3_21

generalizes the possible event sequences of the system from which the event log originates, is less explored. The origin of such event logs is usually real-world systems in domains such as business [2], manufacturing [19,23], or healthcare [9,11,21]. Studies have shown that measuring the generalization of discovered process models is of importance [16] and that only a few methods focus on this objective. Meanwhile, the research community is aware that existing methods do not fully address requirements and present individual shortcomings [12,18].

Adversarial System Variant Approximation (AVATAR) is a method [20] to overcome some of the known issues in measuring generalization. This method leverages a Generative Adversarial Network (GAN) that is trained on the same event log that is used to discover a process model. AVATAR is based on the fact that GANs successfully demonstrated the ability to unveil underlying data distributions, including discrete sequences, and transfers the approach to the context of measuring the generalization of process models. By sampling from the GAN, a baseline of supposedly generalizing event sequences is obtained. Experimental evaluations have been performed using ground truth systems which have shown that the GAN of AVATAR can model observed event sequences of the event log, and unobserved event sequences of the ground truth system accurately.

Whereas the experimental evaluation of AVATAR demonstrated that GANs are suitable and promising neural network architectures that can be used to measure the generalization of a process model, further research is required to understand the working conditions of those GANs in depth. This paper contributes to this objective by conducting performance analyses on the GANs of AVATAR using the same ground truth systems that were used in the original publication. First, the performance analysis includes an experimental evaluation of the proposed sampling parameter k value of $10,000$ of the AVATAR GAN. Second, experiments are performed on limited event log sizes. The original publication used a constant 70% split ratio of the event sequences of the ground truth systems that were used as the event log for process discovery and AVATAR. Under real-world conditions, such a constant 70% split ratio is usually infeasible. Hence, it is necessary to investigate the GAN performance of AVATAR using different split ratios. Third, an experimental evaluation is performed on the robustness of AVATAR towards bias. Specifically, this paper investigates if event logs that are biased affect the ability of the GAN to unveil unobserved event sequences of the ground truth system. The results of the experiments are used to draw conclusions and to raise awareness about the working conditions of the GANs of AVATAR. The results and source codes are available on Github[1].

2 Related Work

2.1 Generalization Metric

Generalization describes that a process model, such as a Petri net (PN), models ideally all possible event sequences of a system that can realistically occur.

[1] https://github.com/ProminentLab/AVATAR.

This means that a process model should allow for the event sequences that are recorded in an event log when observing a system under investigation. These event sequences are usually used to automatically discover a process model using a process discovery algorithm. Additionally, the process model should not allow for unrealistic event sequences beyond the observed ones. It is obvious that the difficulty of measuring the generalization of a process model reduces to classifying if given unobserved event sequences are either realistic or unrealistic in the context of the system under investigation.

A significant amount of research has been spent on measuring how well a process model allows for event sequences contained in an event log (i.e., measuring the *fitness*) and how well a process model restricts to allow for event sequences beyond the ones contained in an event log (i.e., measuring the *precision*). However, research on measuring the generalization of process models is scarce due to the difficulty of deriving realistic and unobserved event sequences from an event log. Nonetheless, the process mining research community is aware that the quality dimension of generalization is of importance [7,12,18].

Historically, one of the first approaches to quantify the extent to which a process model generalizes event sequences beyond the ones contained in an event log has been introduced by Buijs et al. [8]. The proposed approach is based on quantifying the trustworthiness of the precision of a process model using alignments. Highly frequent used areas of a process model are considered well generalizing whereas low frequent parts of the model are less generalizing.

Van der Aalst et al. [1] built a measurement to quantify that a process model does not overfit on a given event log. Specifically, their approach is based on the probability of observing a new event in any given state of the model based on the observations contained in the event log. If the likelihood of observing a new event in a given state is small, then the generalization is good.

Vanden Broucke et al. [6] introduced a method to measure the generalization of a process model using weighted artificial negative events. In comparison to an actual event, an artificial negative event prevents the occurrence of a specific event at a given time. This concept enables to derive allowed and disallowed generalized event sequences.

A method proposed by van Dongen et al. [10] is based on anti-alignments which are event sequences that are disparate from a set of given event sequences. This notion is used to measure the generalization by relating the state space of a process model. A generalizing process model has therefore a maximally different set of anti-alignments without introducing unseen states.

A comparative study by Janssenswillen et al. [13] led to the conclusion that metrics that quantify the generalization with respect to a given event log do usually not assess the quality of a process model concerning the underlying system correctly. Hence, generalization metrics need to be developed that do not solely relate modeled event sequences to the ones contained in an event log. Such metrics should be evaluated using ground truth systems.

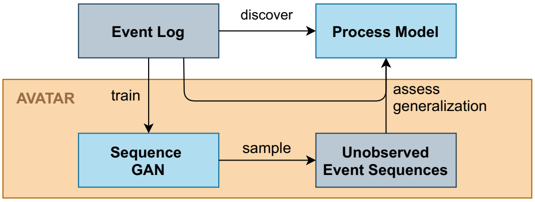


Fig. 1. Flow chat of the AVATAR methodology, derived from [20]

2.2 Adversarial System Variant Approximation

AVATAR is a recently proposed approach to quantify the extent to which a process model generalizes [20]. The idea of this method is to unveil realistic but unobserved event sequences of a system using Generative Adversarial Networks (GANs). If it is possible to confidently model unobserved event sequences using GANs, then measuring the generalization reduces to measuring the fitness and precision of a process model using the observed event log in combination with the unobserved event sequences that are modeled by the GAN. This is motivated by the generalization capabilities of GANs [3]. A flow chart of the methodology is provided in Fig. 1. A given set of event sequences that is used for automated process discovery is also used to train a Sequence GAN (SGAN). AVATAR leverages a RelGAN [15] architecture that is enhanced with an additional standard discriminator neural network. A major hyperparameter of this SGAN architecture is the temperature control β of the RelGAN that controls the tradeoff between sample diversity and quality. The trained SGAN is then used to sample unobserved event sequences. AVATAR proposes therefore two sampling methodologies. The first is *naive sampling* controlled by the parameter k which means that k samples are drawn from the generator of the SGAN. The intuition is that the number of unique event sequences converges with an increasing number of sampling iterations. This also means that the relative frequency of an event sequence indicates the modeling confidence of this particular event sequence. The second sampling methodology uses the *Metropolis-Hastings algorithm* [14] and is inspired by the work of Turner et al. [22]. It is assumed that by sampling from the SGAN, the unobserved event sequences of a system can be unveiled. Here, quantifying the generalization of a process model reduces to measuring the fitness and precision of the process model with respect to the set of observed and approximated unobserved event sequences from the GAN.

The AVATAR methodology has been statistically evaluated using the finite set of event sequences of 15 ground truth PNs. These PNs were created artificially as part of a comparative study of process discovery quality measures [13] and are publicly available[2]. Each of the 15 PNs has different, but realistic characteristics. 10 of the PNs can be classified as *moderately complex* with a small number of transitions and comparatively few parallelisms whereas 5 PNs

[2] https://github.com/gertjanssenswillen/processquality/.

are *highly complex* with a larger number of transitions and parallel structures. The *highly complex* PNs are supposed to reflect the complexity of real-world systems. For each ground truth PN, a random and unbiased 70% random split of the modeled unique event sequences was considered as an event log. These event logs were used to discover process models using two process discovery algorithms [4,5]. The remaining 30% were withheld as the set of unobserved event sequences that the GAN should be able to model.

The results of the experimental evaluation showed that SGANs are well suited to obtain realistic unobserved event sequences with a relatively small number of unrealistic event sequences. Moreover, the AVATAR generalization scores were compared to existing generalization metrics on the discovered process models. The obtained AVATAR scores on those models were perceived more appropriate than the scores of existing generalization measures based on the ground truth event sequence information. All experimental results were obtained under ideal working conditions.

3 Notations

The notations that are used throughout this paper are based on and consistent with the ones of the original AVATAR publication. The reader is referred to [20] for comprehensive introductions.

A system is denoted by S. An event $a \in \mathcal{A}$ describes an instantaneous change of the state of S where $\mathcal{A}$ is the finite set of all possible events. The cardinality of a set is denoted by $|\cdot|$. An event instance E is a vector and describes the occurrence of a specific a along with its occurrence timestamp and optional additional information. A trace is a finite and chronologically ordered sequence of event instances. A *variant* $v \in \mathcal{V}$ is a sequence of events where $\mathcal{V}$ is the infinite set of all variants. A trace maps to exactly one variant. Whereas an event log is a set of traces, denoted by $\mathcal{L}$, a variant log is a sample of variants denoted by $\mathcal{L}^*$. A *unique variant log* is denoted by $\mathcal{L}^+$ and equals to the set of $\mathcal{L}^*$. The set of all variants that can be observed during the runtime of S is denoted by $\mathcal{V}_S$. The functions $\mu(\mathcal{V})$ and $mean(\mathcal{V})$ return the maximum and mean variant lengths of a given set of variants, respectively.

Following the AVATAR methodology, a SGAN architecture is trained on $\mathcal{L}^+$ with a hyperparameter β, i.e., GAN_β. The SGAN can be used to naively sample variants. The number of sampling iterations from GAN_β is denoted by k.

When training a GAN, all variants of $\mathcal{L}^+ \subseteq \mathcal{V}_S$ are considered. A subset of variants $\mathcal{V}_u$ might exist such that $\mathcal{V}_S = (\mathcal{L}^+ \cup \mathcal{V}_u)$ and $(\mathcal{L}^+ \cap \mathcal{V}_u) = \emptyset$. $\mathcal{V}_u$ is intuitively the set of unobserved behavior. Ideally, when sampling k times from GAN_β, it is desired to obtain an estimated set of system variants, i.e., $\hat{\mathcal{V}}_S$ that equals to $\mathcal{V}_S$. How well the GAN performs to reach this goal is quantified using the true positive ratios $tp = \frac{|\hat{\mathcal{V}}_S \cap \mathcal{V}_S|}{|\mathcal{V}_S|}$ and $tp_u = \frac{|\hat{\mathcal{V}}_S \cap \mathcal{V}_u|}{|\mathcal{V}_u|}$. tp describes the proportion of realistic variants sampled using GAN_β over all possible system variants. tp_u describes the ratio of sampled variants using GAN_β over all unobserved variants. Moreover, the number of unique sampled variants is recorded. Ideally,

tp and tp_u should be equal to 1 while the number of unique sampled variants should equal $|\mathcal{V}_S|$. The score function $s(tp, tp_u) = \frac{tp+tp_u}{\sqrt{2}}$ is used, as proposed and reasoned in [20], to quantify how well the GAN of AVATAR performs.

4 Problem Statement

The AVATAR methodology [20] demonstrated that SGANs can model $\mathcal{V}_u$ which builds a foundation to measure the generalization of process models. The evaluation setup of AVATAR consisted of a 70/30 split ratio of $\mathcal{V}_S$ to obtain $\mathcal{L}^+$ and $\mathcal{V}_u$ and a sampling parameter k value $10,000$ for each of the ground truth systems. This setup raises multiple research questions, including the following.

RQ1: *Is the parameter k with a value of* $10,000$ *optimally defined and is there a relationship between k and the GAN performance of AVATAR?* The parameter k describes the number of variants that are drawn naively from the trained SGAN without leveraging the Metropolis-Hastings algorithm. Whereas [20] states that preliminary results showed that setting k to the value of $10,000$ is a good choice, a proven justification for this value is missing. Moreover, it remains unclear if a relationship between k and the performance of the GAN of AVATAR exists. This paper experimentally assesses the performance of the GANs with multiple values for k to validate the statement made in the original publication and investigates the relationship between S, k, and the GAN to model $\mathcal{V}_S$.

RQ2: *How does the size of $\mathcal{L}^+$ relate to the performance of modeling $\mathcal{V}_S$?* The AVATAR methodology has been evaluated using a 70/30 split ratio of $\mathcal{V}_S$ to obtain $\mathcal{L}^+$ and $\mathcal{V}_u$ across all used ground truth systems. However, it remains unclear how the GAN of AVATAR performs if less information of a system is given. In real-world scenarios, an exact 70% split of all possible variants of a system is usually unrealistic. The ratio of variants contained in $\mathcal{L}^+$ to all variants in $\mathcal{V}_S$ can be guessed at its best. Hence, this paper experimentally assesses the performance of the GANs of AVATAR at different split ratios to investigate the working conditions of AVATAR when the given event log size is limited.

RQ3: *Are the GANs of AVATAR sensitive to biased variant logs?* The GANs of AVATAR have been evaluated using a random and unbiased split of $\mathcal{V}_S$. In real-world scenarios though, $\mathcal{L}^+$ might be biased due to a limited observation duration of the system or adverse environmental situations. Whereas research has been conducted on the impact of biased event logs on process discovery algorithms [17], it remains unclear how the GANs of AVATAR perform when being trained on a biased set of variants. Bias can be expressed, e.g., in terms of variant lengths. In this paper, preliminary experiments are performed to investigate if the performance of the GANs are affected when being trained on specific biased variant logs.

5 Experimental Setup

5.1 Sampling Parameter

To investigate the relationship between k and the performance of the SGANs (*RQ1*), multiple values for k are investigated. Specifically, k is set to $1,000$, $2,000, 4,000, 6,000$, up to $20,000$, with an increment of $2,000$ each. This includes the originally proposed $k = 10,000$ value. These specific values are chosen such that performance changes can be observed when increasing and decreasing the proposed value of k. It is expected that the performance of the GANs decreases with a very small value, such as $k = 1,000$, but it remains unclear if the performance increases with an increased value of k. It is not expected that a granularity finer than $2,000$ will unveil significant differences.

Training and sampling of the SGANs is performed on the five highly complex PNs that were also used to evaluate the AVATAR methodology according to the original publication. These systems are denoted as S_{11-15} and correspond to Systems 11–15 in [20]. For each of the five systems, two SGANs are trained with $\beta = 100$ and $\beta = 1000$, respectively. These GANs are trained using a random 70% split of $\mathcal{V}_S$ which corresponds to $\mathcal{L}^+$. The remaining 30% results in $\mathcal{V}_u$ and are used to evaluate the performance of the SGAN to approximate the unobserved system variants, as in the original publication. This is called a *70/30 split ratio*. The setup results in ten different SGAN models and, due to 11 different values for k, in a total of 110 observation values for evaluation.

5.2 Variant Log Size

To investigate the performance of the GANs of AVATAR when limited variant log sizes are given (*RQ2*), two SGANs per system are trained with different split ratios compared to the 70/30 ratio of the original evaluation. In this setup, the 70/30 split ratio is used as a baseline for comparison. Moreover, experiments are performed using 10/90, 20/80, 30/70, 40/60, 50/50, and 60/40 split ratios. It is expected that the performance of the GANs in modeling $\mathcal{V}_u$ decreases with smaller $|\mathcal{L}^+|$ values. As before, the systems S_{11-15} are used for experimental evaluation due to their realistic complexity. The SGANs are trained with $\beta = 100$ and $\beta = 1000$ to be consistent with the original AVATAR work. This results in 70 SGANs for evaluation. Variants are generated from the SGANs using the originally proposed $k = 10,000$ value.

5.3 Biased Variant Logs

This experiment investigates the performance of the GANs of AVATAR in detecting $\mathcal{V}_u$ when being trained on a biased $\mathcal{L}^+$ to provide an answer to *RQ3*. Bias is expressed using the length of variants. The baseline is obtained using a random and unbiased 70/30 split ratio on $\mathcal{V}_S$ such that $mean(\mathcal{L}^+)$ and $mean(\mathcal{V}_u)$ are almost equal. Four bias setups are defined and denoted by $b1$ to $b4$.

The first bias setup $b1$ is defined such that $\mathcal{L}^+$ contains the shortest 70% of $\mathcal{V}_s$ and $\mathcal{V}_u$ contains the remaining 30%. This means that a SGAN is trained on short variants, but is supposed to generalize to long variants. The setup $b2$ is defined such that $\mathcal{L}^+$ contains the longest 70% of $\mathcal{V}_s$ and $\mathcal{V}_u$ contains the remaining variants. In this case, a SGAN is trained on long variants and is supposed to generalize short variants. The setups $b3$ and $b4$ are leaky variations of $b1$ and $b2$, respectively. For both setups, 20% of the variants in $\mathcal{V}_u$ are randomly exchanged with a randomly chosen variant from $\mathcal{L}^+$. This means that the corresponding SGAN is not trained on strictly short or strictly long variants. However, bias in terms of the lengths of variants contained in $\mathcal{L}^+$ and $\mathcal{V}_u$ persists.

For all setups $b1$ to $b4$, the longest possible variant of a corresponding system is contained in $\mathcal{L}^+$ rather than $\mathcal{V}_u$. This is required to satisfy the assumption that the maximum possible system variant length is known to train an SGAN [20]. Therefore, at least one variant with a length equal to $\mu(\mathcal{V}_S)$ must be known. Like before, two SGANs are trained with $\beta = 100$ and $\beta = 1000$, respectively, for each of the systems S_{11-15} and each setup plus the baseline setup. Consequently, the total number of SGAN models under investigation equals 50.

6 Results

6.1 Sampling Parameter Results

For S_{11} and GAN_{100}, the number of approximated system variants increases with the value of k. This GAN setup is closest to the desired $|\mathcal{V}_S|$ value when using $k = 8,000$. In the meantime, the tp ratio decreases with an increasing value of k. With an increasing value of k, the tp_u ratio converges to 0.6. Similar behavior is observed for the SGANs for S_{12}. However, with $k = 2,000$, $\hat{\mathcal{V}}_S$ already exceeds the desired value of $\mathcal{V}_S$. Accordingly, tp decreases and tp_u converges with increasing k to about 0.8. The overestimation of variants can be explained by the complexity of the underlying system. The second most complex system is S_{14} with a much smaller maximum variant length. Accordingly, the SGANs of S_{14} are better in approximating $\mathcal{V}_S$ compared to the ones of S_{12}. Systems S_{13-15} perform similarly to S_{11} with an optimal variant number approximation around $k = 10000$. The tp_u ratios seem to converge around 0.7 and 0.9.

The results look similar for GANs with $\beta = 1,000$. In general, $\hat{\mathcal{V}}_S$ is overestimated with an increasing value of k and when $k > 10,000$. Only for S_{11}, the corresponding SGAN underestimates $|\mathcal{V}_S|$ when using any of the considered values for k. However, for $k = 20,000$, GAN_{1000} almost perfectly estimates $|\mathcal{V}_S|$ with a decently high tp and tp_u ratio. Generally, the tp ratio reduces with a more gentle slope compared to GAN_{100} while tp_u converges to a fixed value similar to GAN_{100}. The tp_u convergence value lies between 0.75 and 1.0.

Since it can be observed that the performance of the GANs on more complex systems, such as S_{12}, can be weaker, a linear regression model is fit using the features k, $\mu(\mathcal{V}_S)$, and $|\mathcal{V}_S|$ to model the resulting scoring value for s. With linear features, this leads to an R^2 value of 1.4% indicating a bad fit. With the corresponding quadratic features, the R^2 score improves to 40%. The quadratic relationship could be an initial step to develop a rule-of-thumb to select an individual and optimized value for k. The required values for $|\mathcal{V}_S|$, $\mu(\mathcal{V}_S)$ and a desired minimum score value s can be guessed by using expert knowledge.

The median value of k that corresponds to the best obtained scores for the SGAN models under consideration, equals $10,000$. This validates the general suitability of $k = 10,000$ as proposed in [20] and answers *RQ1*.

6.2 Variant Log Size Results

For the GANs that were trained using $\beta = 100$, it can be generally noted that fewer unique variants are sampled with decreasing sizes of $\mathcal{L}^+$. At the same time, it can also be observed that tp and tp_u generally tend to decrease. A similar, but less significant behavior can be observed for the SGANs that are trained using $\beta = 1000$. This confirms the expectations.

The same trend can be observed when visualizing the 90% confidence intervals (CIs) of the obtained scores s for each SGAN and variant log size setup over all systems, as visualized in Fig. 2. Whereas this visualization cannot provide statistical proof due to the small sample size, it shows the decreasing trend satisfyingly. Since the CIs for a 10/90 split ratio and the baseline 70/30 split ratio for both SGAN setups are non-overlapping, it can be concluded that a 10/90 split ratio performs statistically poorer than a 70/30 split ratio with 90% confidence.

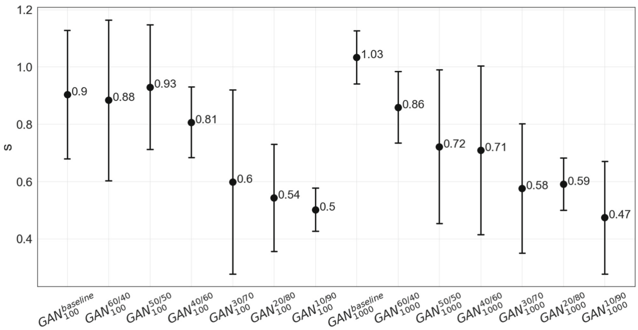


Fig. 2. 90% CIs of the mean scores s for each SGAN setup of different $\mathcal{L}^+$ sizes over all systems S_{11-15}

To provide an answer to *RQ2*, the GAN performance decreases with less variants contained in $\mathcal{L}^+$ with respect to $|\mathcal{V}_S|$. These experiments prove that the SGANs of AVATAR trained with a 70/30 split ratio perform statistically significantly better compared to a 10/90 split ratio. For GAN_{1000}, the experiments show that a 70/30 split ratio leads to statistically significantly better performance compared to 30/70, 20/80, and 10/90 split ratios. Further experiments with a larger sample size are required to provide statistical proof.

6.3 Biased Variant Log Results

For all systems, the SGAN using $\beta = 100$ on the biased setup $b1$ performs poorly. However, when training using $\beta = 1000$, the performance seems to be increasing. The β parameter indicates an impact on the performance when $\mathcal{L}^+$ is biased. However, the details of the impact remain unclear. Overall, the SGANs trained with $\beta = 1000$ seem to perform better in general.

Furthermore, the performance seems to increase when $\mathcal{L}^+$ is less restrictively biased, i.e., with the setups $b3$ and $b4$ compared to $b1$ and $b2$, respectively. This indicates that less bias leads to better performance. Additionally, $b2$ seems to perform better than $b1$, and $b4$ performs better than $b3$. The same can be observed when visualizing the 90% CIs of the scores s per SGAN setup over all systems in Fig. 3. Comments on the statistical significance of each CI cannot be made due to the small sample size. However, the CI mean values indicate the observed trend. The baseline SGANs are the best-performing models. When introducing leaky bias with $b3$ and $b4$, the performance reduces on average. Strict bias, such as with setups $b1$ and $b2$, leads to a further decrease of performance in unveiling $\mathcal{V}_S$. The large CIs for the SGANs trained using $\beta = 1000$ and using the setups $b3$ and $b4$ can be either a randomness artifact or a sign that the β hyperparameter can accommodate for non-strict bias in specific situations.

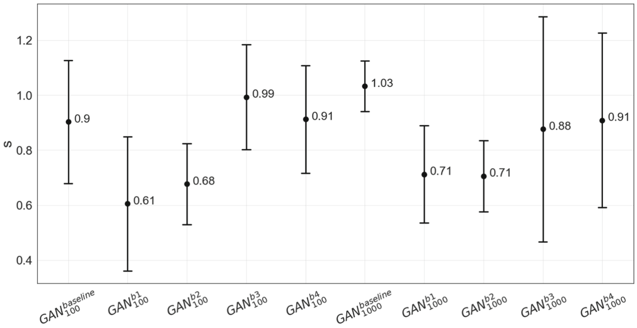


Fig. 3. 90% CIs of the mean scores s for each biased $\mathcal{L}^+$ and baseline SGAN setup over all systems S_{11-15}

To answer *RQ3*, the GANs are sensitive to bias and perform with a s value that decreases proportionally to the significance of present variant length bias in $\mathcal{L}^+$. Further experiments with larger sample sizes of ground truth systems are anticipated to provide statistical evidence and insights on the potential impact of β to accommodate for bias.

7 Conclusion

Regarding *RQ1*, the experiments have shown that $k = 10,000$ is generally a good choice. However, an individual value for k is required depending on the underlying system complexity to fine-tune the GAN performance. Linear regression with quadratic features indicated a good fit to estimate an optimized value for k given the desired performance score s, the total number of system variants, and the maximum variant length of the underlying system. For *RQ2*, the GAN performance in modeling $\mathcal{V}_S$ generally tends to decrease when fewer variants of the system are contained in $\mathcal{L}^+$. Finally, the GANs of AVATAR seem to be sensitive towards biased variant logs, as an answer to *RQ3*. The performance of the underlying SGANs decreases the more significant the bias in $\mathcal{L}^+$ is. Moreover, the experimental results show the potential that the SGAN hyperparameter β might be able to accommodate for bias in specific situations.

While the experimental results unequivocally highlight certain conditions of the GANs that need to be considered when applying AVATAR, detailed statistical evidence remains mostly missing due to limited sample sizes. Hence, the results of this paper should raise awareness to the research community and provide the following three research directions. First, the results of the parameter k investigations motivate future experimental evaluations to derive a rule-of-thumb to select an optimal value k. This requires an experimental evaluation using a large set of different ground truth systems to derive a robust rule-of-thumb. Second, a larger set of experiments need to be conducted to investigate the required variant log size to train a converging GAN such that AVATAR can be applied confidently. Third, the bias sensitivity of the GANs of AVATAR needs to be investigated with a larger set of ground truth systems and with different β hyperparameter values to unveil a potential relationship between β and the GAN sensitivity towards bias.

References

1. van der Aalst, W., Adriansyah, A., van Dongen, B.: Replaying history on process models for conformance checking and performance analysis. Wiley Interdisc. Rev. Data Mining Knowl. Disc. **2**(2), 182–192 (2012). https://doi.org/10.1002/widm.1045
2. van der Aalst, W.M., et al.: Business process mining: an industrial application. Inf. Syst. **32**(5), 713–732 (2007)
3. Arora, S., Ge, R., Liang, Y., Ma, T., Zhang, Y.: Generalization and equilibrium in generative adversarial nets (GANs). In: Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 224–232. JMLR. org (2017)
4. Augusto, A., Conforti, R., Dumas, M., La Rosa, M., Polyvyanyy, A.: Split miner: automated discovery of accurate and simple business process models from event logs. Knowl. Inf. Syst. **59**(2), 251–284 (2018). https://doi.org/10.1007/s10115-018-1214-x
5. vanden Broucke, S.K., De Weerdt, J.: Fodina: a robust and flexible heuristic process discovery technique. Decis. Support. Syst. **100**, 109–118 (2017)
6. vanden Broucke, S.K., De Weerdt, J., Vanthienen, J., Baesens, B.: Determining process model precision and generalization with weighted artificial negative events. IEEE Trans. Knowl. Data. Eng. **26**(8), 1877–1889 (2013)
7. Buijs, J.C., Van Dongen, B.F., Van Der Aalst, W.M.: Quality dimensions in process discovery: the importance of fitness, precision, generalization and simplicity. Int. J. Cooper. Inf. Syst. **23**(1), 1440001 (2014). https://doi.org/10.1142/S0218843014400012

8. Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: On the role of fitness, precision, generalization and simplicity in process discovery. In: Meersman, R., et al. (eds.) OTM 2012. LNCS, vol. 7565, pp. 305–322. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33606-5_19
9. Darabi, H., Galanter, W.L., Lin, J.Y., Buy, U., Sampath, R.: Modeling and integration of hospital information systems with Petri nets. In: 2009 IEEE/INFORMS International Conference on Service Operations, Logistics and Informatics, pp. 190–195, July 2009. https://doi.org/10.1109/SOLI.2009.5203928
10. van Dongen, B.F., Carmona, J., Chatain, T.: A unified approach for measuring precision and generalization based on anti-alignments. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM 2016. LNCS, vol. 9850, pp. 39–56. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45348-4_3
11. Ghasemi, M., Amyot, D.: Process mining in healthcare: a systematised literature review. Int. J. Electron. Healthcare **9**(1), 60–88 (2016)
12. Janssenswillen, G., Depaire, B.: Towards confirmatory process discovery: making assertions about the underlying system. Bus. Inf. Syst. Eng. **61**(6), 713–728 (2018). https://doi.org/10.1007/s12599-018-0567-8
13. Janssenswillen, G., Donders, N., Jouck, T., Depaire, B.: A comparative study of existing quality measures for process discovery. Inf. Syst. **71**, 1–15 (2017). https://doi.org/10.1016/j.is.2017.06.002
14. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculations by fast computing machines. J. Chem. Phys. **21**(6), 1087–1092 (1953)
15. Nie, W., Narodytska, N., Patel, A.: RelGAN: relational generative adversarial networks for text generation. In: International Conference on Learning Representations (2018)
16. Rehse, J.-R., Fettke, P., Loos, P.: Process mining and the black swan: an empirical analysis of the influence of unobserved behavior on the quality of mined process models. In: Teniente, E., Weidlich, M. (eds.) BPM 2017. LNBIP, vol. 308, pp. 256–268. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-74030-0_19
17. Fani Sani, M., van Zelst, S.J., van der Aalst, W.M.P.: The impact of biased sampling of event logs on the performance of process discovery. Computing **103**(6), 1085–1104 (2021). https://doi.org/10.1007/s00607-021-00910-4
18. Syring, A.F., Tax, N., van der Aalst, W.M.P.: Evaluating conformance measures in process mining using conformance propositions. In: Koutny, M., Pomello, L., Kristensen, L.M. (eds.) Transactions on Petri Nets and Other Models of Concurrency XIV. LNCS, vol. 11790, pp. 192–221. Springer, Heidelberg (2019). https://doi.org/10.1007/978-3-662-60651-3_8
19. Theis, J., Mokhtarian, I., Darabi, H.: Process mining of programmable logic controllers: input/output event logs. In: 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE), pp. 216–221, August 2019. https://doi.org/10.1109/COASE.2019.8842900
20. Theis, J., Darabi, H.: Adversarial system variant approximation to quantify process model generalization. IEEE Access **8**, 194410–194427 (2020)
21. Theis, J., Galanter, W., Boyd, A., Darabi, H.: Improving the in-hospital mortality prediction of diabetes ICU patients using a process mining/deep learning architecture. IEEE J. Biomed. Health Inf. **26**(1), 388–399 (2022). https://doi.org/10.1109/JBHI.2021.3092969
22. Turner, R., Hung, J., Frank, E., Saatchi, Y., Yosinski, J.: Metropolis-Hastings Generative Adversarial Networks. In: International Conference on Machine Learning, pp. 6345–6353 (2019)

23. Yang, H., Park, M., Cho, M., Song, M., Kim, S.: A system architecture for manufacturing process analysis based on big data and process mining techniques. In: 2014 IEEE International Conference on Big Data (Big Data), pp. 1024–1029. IEEE (2014)

PODS4H 2021: 4th International Workshop on Process-Oriented Data Science for Healthcare

Fourth International Workshop on Process-Oriented Data Science for Healthcare (PODS4H)

The world's most valuable resource is no longer oil, but data. The ultimate goal of data science techniques is not to collect more data, but to extract knowledge and valuable insights from existing data in various forms. To analyze and improve processes, event data is the main source of information. In recent years, a new discipline has emerged combining traditional process analysis and data-centric analysis: Process-Oriented Data Science (PODS). The interdisciplinary nature of this new research area has resulted in its application to analyze processes in a wide range of different domains such as education, finance, and especially healthcare.

The International Workshop on Process-Oriented Data Science for Healthcare 2021 (PODS4H21) aims at providing a high-quality forum for interdisciplinary researchers and practitioners (both data/process analysts and a medical audience) to exchange research findings and ideas on healthcare process analysis techniques and practices. PODS4H research includes a wide range of topics ranging from process mining techniques adapted for healthcare processes to practical issues related to the implementation of PODS methodologies in healthcare organizations. For more information, we would like to refer the reader to our website pods4h.com.

The fourth edition of the workshop has been organized in conjunction with the International Conference on Process Mining in Eindhoven (the Netherlands). We have received 10 full paper submissions, of which 5 full papers have been accepted after being reviewed by three experts from our Program Committee. The papers focused on a wide range of topics: heterogeneous treatment effect modeling, clinical guideline compliance, patient discharge classification, interactive process mining, and care pathway discovery. Besides the presentation of the full papers included in these proceedings, the workshop program also contained a poster session and a panel discussion.

This edition of the workshop also included a Best Paper Award. The Best Paper Award of PODS4H21 was given to Sam Verboven and Niels Martin with their paper "Combining the Clinical and Operational Perspectives in Heterogeneous Treatment Effect Inference in Healthcare Processes".

The PODS4H workshop is an initiative of the Process-Oriented Data Science for Healthcare Alliance (PODS4H Alliance) within the IEEE Task Force on Process Mining. The goal of the PODS4H Alliance is to promote awareness, research, development, and education regarding process-oriented data science in healthcare. For more information about the activities of the alliance, we refer you to the webpage pods4h.com/alliance.

The organizers would like to thank all the Program Committee members for their valuable work in reviewing the papers, as well as the ICPM 2021 workshop chairs and local organizers for supporting this successful event.

November 2021

Niels Martin
Carlos Fernandez-Llatas
Owen Johnson
Marcos Sepúlveda
Emmanuel Helm

Organization

Workshop Chairs

Carlos Fernandez-Llatas	Universitat Politècnica de València
Niels Martin	Hasselt University
Owen Johnson	University of Leeds
Marcos Sepúlveda	Pontificia Universidad Católica de Chile
Emmanuel Helm	University of Applied Sciences Upper Austria

Program Committee

Davide Aloini	Università di Pisa
Robert Andrews	Queensland University of Technology
Iris Beerepoot	Utrecht University
Elisabetta Benevento	Università di Pisa
Andrea Burattin	Technical University of Denmark
Daniel Capurro, M. D.	University of Melbourne
Marco Comuzzi	Ulsan National Institute of Science and Technology
Benjamin Dalmas	École des Mines de Saint-Étienne
Carlos Fernandez-Llatas	Universitat Politècnica de Valencia
René de la Fuente, M. D.	Pontificia Universidad Católica de Chile
Claudio di Ciccio	Sapienza University of Rome
Onur Dogan	Izmir University Bakircay
Roberto Gatta	Università Cattolica del Sacro Cuore
Emmanuel Helm	University of Applied Sciences Upper Austria
Owen Johnson	University of Leeds
Felix Mannhardt	Eindhoven University of Technology
Ronny Mans	Philips Research
Niels Martin	Hasselt University
Mar Marcos	Universitat Jaume I
Renata Medeiros de Carvalho	Eindhoven University of Technology
Jorge Munoz-Gama	Pontificia Universidad Católica de Chile
Marco Pegoraro	RWTH Aachen University
Simon Poon	University of Sydney
Luise Pufahl	Technische Universität Berlin
Ricardo Quintano	Philips Research
Hajo Reijers	Utrecht University
David Riaño	Universitat Rovira i Virgili
Stefanie Rinderle-Ma	Technical University of Munich

Eric Rojas	Pontificia Universidad Católica de Chile
Lucia Sacchi	University of Pavia
Fernando Seoane	Karolinska Institutet
Marcos Sepúlveda	Pontificia Universidad Católica de Chile
Minseok Song	Pohang University of Science and Technology
Alessandro Stefanini	Università di Pisa
Emilio Sulis	Università di Torino
Pieter Toussaint	Norwegian University of Science and Technology
Vicente Traver	Universitat Politècnica de Valencia
Wil van der Aalst	RWTH Aachen University
Rob Vanwersch	Maastricht University Medical Center
Mathias Weske	HPI – University of Potsdam
Moe Wynn	Queensland University of Technology

Verifying Guideline Compliance in Clinical Treatment Using Multi-perspective Conformance Checking: A Case Study

Joscha Grüger[1,2](✉), Tobias Geyer[2], Martin Kuhn[2], Stephan A. Braun[3,4], and Ralph Bergmann[1,2]

[1] Business Information Systems II, University of Trier, Trier, Germany
grueger@uni-trier.de

[2] German Research Center for Artificial Intelligence (DFKI), Trier, Germany

[3] Department of Dermatology, University Hospital Münster, Münster, Germany

[4] Department of Dermatology, Medical Faculty, Heinrich Heine University, Düsseldorf, Germany

Abstract. Clinical guidelines support physicians in the evidence-based treatment of patients. The technical verification of guideline compliance is not trivial, since guideline knowledge is usually represented textually and none of the approaches to computer-interpretable guideline representation has yet been able to establish itself. Due to the procedural nature of treatment sequences, this case study examines the applicability of a guideline process model to real hospital data for verification of guideline compliance. For this purpose, the limitations and challenges in the transformation of clinical data into an event log and in the application of conformance checking to align the data with the guideline reference model are investigated. As a data set, we use treatment data of skin tumor patients from a cancer registry enriched by hospital information system data. The results show the difficulty of applying process mining to medically complex and heterogeneous data and the need for complex preprocessing. The variability of clinical processes makes the application of global conformance checking algorithms challenging. In addition, the work shows the semantic weakness of the alignments and the need for new semantically sensitive approaches.

Keywords: Process mining · Multi-perspective Conformance checking · Clinical guidelines · Guideline compliance

1 Introduction

Evidence-based medicine states that patient-centered medical treatment decisions should be based on empirically proven effectiveness whenever possible [23].

The healthcAIre project is funded by the ministry of science and health of the German state Rhineland-Palatinate and the Pre-OnkoCase project is funded by the National Care Conference on Skin Cancer (NVKH) e.V.

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 301–313, 2022.
https://doi.org/10.1007/978-3-030-98581-3_22

This knowledge is documented in clinical guidelines [25]. The degree to which clinical treatment processes in practice are guideline-compliant and thereby evidence-based is unknown [9,14]. The verification of guideline compliance is relevant, e.g., for the certification of oncology centers[1], the development of clinical decision support systems [2,15] and medical research [3,12]. An approach to check the compliance of treatment processes against guidelines is to interpret individual treatment processes as process instances and the guideline as a reference process model [10]. This would enable the use of conformance checking, a process mining technique, which raises two challenges though. First, the transformation of the guideline knowledge into a process model. Second, the provision of clinical data as event logs and their preprocessing. Due to the lack of standardization of clinical data storage and the associated structure heterogeneity, naming and data quality, a preprocessing of this data is necessary [13]. Furthermore, clinical processes are characterized as highly variable, ad-hoc, multidisciplinary and vary from hospital to hospital [22].

As part of the Pre-OnkoCase[2] project, a process model was developed for a section of the malignant melanoma guideline. In this case study, we investigate to what extent the model can be applied to real clinical data, what preprocessing is necessary and what limitations exist. The case study was conducted in collaboration with the skin tumor center of the Münster University Hospital[3].

The remainder of the paper is organized as follows. Section 2 provides background information about the medical context, the process reference model and conformance checking details. Section 3 describes the research method and shows how the event log is created. Section 4 describes the implementation of the conformance checking and the required preprocessing. In Sect. 5, the results are discussed and Sect. 6 concludes the paper.

2 Background

Within the Pre-OnkoCase research project, we investigated how a clinical guideline can be represented procedurally. Since guidelines assume tacit knowledge, they provide an incomplete representation of the treatment processes. Therefore, missing information had to be supplemented by experts' knowledge. In workshops with domain experts of the skin tumor center in Münster, a conceptual model of a section of the evidence-based guideline for the treatment of malignant melanoma (skin cancer) [6] was created. Due to the size and complexity of the model Fig. 1 shows just a sketch of the fundamental treatment process. Each element of the sketch is a representative of a treatment section in the treatment courses of patients who have been diagnosed with melanoma and consists of a set of activities. If malignant melanoma is diagnosed during the clinical and histopathological examination, which is part of the *Diagnosis of Melanoma* section, then several treatment options are available to the patient.

[1] https://www.krebsgesellschaft.de/deutsche-krebsgesellschaft/zertifizierung.html.

[2] https://nvkh.de/projekte/pre-oncocase.

[3] https://www.ukm.de/index.php?id=hauttumorzentrum.

- *Re-Excision*: Repeated excision ensures that no tumor residues remain.
- *Sentinel Lymph Node Biopsy*: The patient can receive a lymph node sonography and receives a re-excision together with the sentinel lymph node biopsy.
- *Other Diagnostic Measures*: The patient can receive diagnostics to confirm metastases or further examinations with imaging techniques.
- *Staging up to IIB & Staging IIC and III*: Depending on prior examinations, the patient can come to staging, which provides patients and physicians the critical benchmark for defining prognosis and for determining the best treatment approach [8].
- *Lymphadenectomy*: The patient receives a lymphadenectomy, can receive radiotherapy afterwards and receives a drug therapy.
- *Adjuvant Therapy*: The patient receives additional cancer treatment after initial treatment to reduce the risk of recurrence.

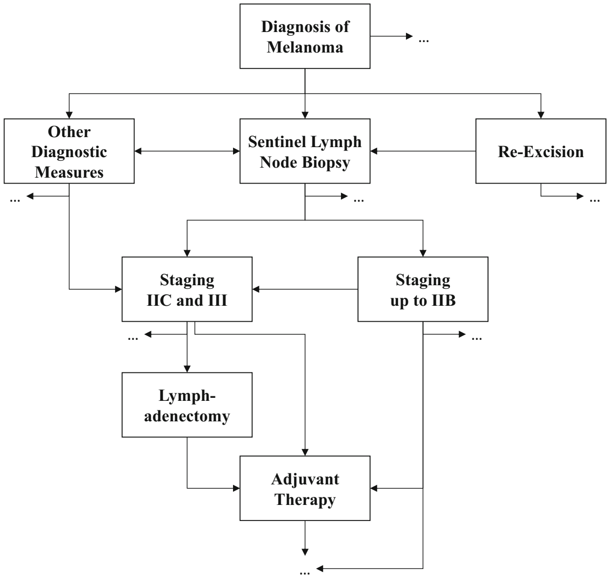


Fig. 1. Reduced overview of the considered clinical guideline section for the treatment of malignant melanoma patients. Paths marked with "..." are treatment areas, which are not covered in the model, e.g., the treatment of stage IV patients or follow-up care.

The final conceptual model was then transferred into a Data Petri Net (DPN) by Geyer [11]. A DPN is an extended Petri net that can map data and time information [16]. The modeled DPN consists of 50 places and 76 transitions (see Fig. 2). Due to the many decisions made in treatment based on examination results, there are 52 transitions with a guard. The resulting model represents all conditions and recommendations of the selected guideline section.

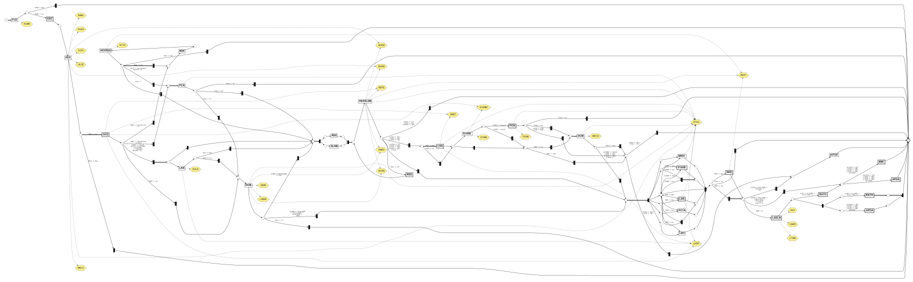


Fig. 2. Overview of the DPN reference model

In the following, the basic terminology in the context of multi-perspective conformance checking is explained. Multi-perspective conformance checking describes the process of identifying discrepancies between the desired behavior of the process, represented by the process model, and the actual behavior involving multiple perspectives such as the data perspective or the time perspective. Most approaches use alignments for this purpose, which are a mapping of the process instance to the process model. In the context of alignments, a *log move* is executed by the alignment algorithm for events that are recorded in the event log but do not occur in the process model. A *model move* is executed if events occur in the process model but do not occur in the event log. If the event from the event log matches the activity in the process model, but the values of the variables do not match, this is called *incorrect synchronous move*. If everything matches, the move is defined as *correct synchronous move* [16].

Most of the process mining algorithms which are capable of calculating multi-perspective alignments are using the Alpha* algorithm [5] in combination with MILP (Mixed Integer Linear Programming) [24]. The state-of-the-art approach is from Mannhardt [17] where DPNs are used for calculating multi-perspective optimal alignments [7]. It is also possible to calculate multi-perspective alignments by using MP-Declare a multi-perspective version of Declare [21]. This method was developed by Mawoko [19] and utilized a similar approach as Mannhardt.

3 Research Method

The exemplary data set used in this project represents the treatment of a total of five real patients diagnosed with malignant melanoma from Münster University Hospital. For data privacy reasons, the data were anonymized. The treatment data are provided in the format of the ADT/GEKID basic data set[4]. The uniform oncological ADT/GEKID basic data set describes a common coding scheme for the documentation of oncological treatments in Germany in the form of an XML schema. A major advantage of using data in the format of the basic data set is that it is used by all German cancer registries and results are thus transferable

[4] https://www.gekid.de/adt-gekid-basisdatensatz.

and comparable. The basic data set includes among others patient master data, diagnostic data, histology data, cancer classification data, surgical data, therapy data and tumor conference data.

Each entry in the basic data set is provided with a timestamp and a treating resource and uniquely assignable to a patient and a treatment case. The structure of the basic data set is based on the obligation of hospitals in Germany to report the course of cancer cases to cancer registries. Accordingly, the data on individual treatment activities are assigned to reporting elements in the XML format and enriched with treatment-specific information. In order to apply conformance checking, the data are transferred into the XES event log format [1]. For this purpose, a generic XML to XES converter was implemented in Python and configured to convert ADT/GEKID data to XES.

The resulting process log covers many areas important for determining guideline compliance, such as surgeries and diagnoses. Also, additional information on follow-up examinations, medical therapies and tumor conferences are contained. However, it lacks information on, e.g., histological examinations, certain tumor markers, or lymphadenectomy. The resulting event log contains 24 different events while considering different medical procedures as different events.

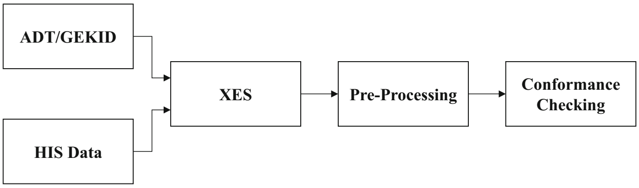


Fig. 3. Overview of the process steps up to conformance checking

In order to be able to take these data into account in conformance checking, the log was enriched with treatment data from the hospital information system (HIS). For this purpose, data from the HIS were exported as CSV and imported into the XES file. Most of the entries could be transferred automatically, since they are structured and timestamped. However, individual details of the treatment process had to be extracted manually from the free text of the diagnostic findings and doctor's letters. The final event log contains 179 different events and a total of 1114 events, an average of 222 events per patient.

4 Implementation

The following describes the adjustments that were necessary to perform conformance checking. An overview of the individual procedures in the project are shown in Fig. 3.

4.1 Preprocessing

The final event log contains 179 different events, while the guideline reference model has only 20 different events. The difference results from the fact that the event log contains events of other medical domains such as nursing and psychosocial care and from the fact that the granularity in which events are represented is inconsistent. In addition, it is also due to the fact that there are deviations from the guideline. In order to perform an alignment between the event log and the guideline reference model, extensive preprocessing had to be performed: removal of explicitly irrelevant events, reduction of therapy events to the respective initial therapy event, harmonization of granularity, event aggregation and event and variable name matching.

In the first step, events that were explicitly irrelevant for conformance checking were removed. These include events from perspectives not considered by the guideline, such as the nursing and psychosocial domains, events such as tumor conferences, which neither establish new diagnoses nor provide direct treatment, and events such as follow-up care, which are outside the selected guideline section. The events were identified using the event names and a HIS-internal ID. Subsequently, in the second step, the therapy sequences of the same therapy were reduced to the respective starting event. This is necessary because cancer therapies are usually performed several times and the reference model of the guideline, however, only addresses whether a patient with certain diagnoses receives a certain therapy and then implies that this therapy is subsequently performed correctly. The granularity of the event log in terms of the event description is in many ways finer than in the reference model. While the reference model refers to “excision”, the ICPM (International Classification of Procedures in Medicine) classification used in the data set defines over 30 different excisions. Therefore, the data set is harmonized in terms of granularity. For this purpose, the ICPM code is abstracted in the hierarchically structured coding scheme to such an extent that the description matches the identifiers of the reference model. This results in partial events with identical designation and identical timestamp, which originally described, e.g., surgeries with several similar individual events are aggregated to one event. It is essential that the names of the same variables and events in the guideline reference model and in the event log are identical. For this purpose, a comparison of the identifiers of the event log with identifiers of the model was performed. This was particularly time-consuming because identifiers were not consistent and unique. This is on the one hand due to the fact that the data set is based on data from two systems and on the other hand due to the fact that the treatment documentation is partially in free text and identifiers were accordingly heterogeneous. The resulting event log forms the basis for the conformance check. After applying the described preprocessing steps, the event log only contains 40 different events directly related to treatment instead of the initial 179. The discrepancy between 20 activities in the model and 40 events in the log was deliberately accepted in order to have complete traces and a comparison between guideline specifications and reality.

4.2 Conformance Checking

The in the following presented conformance checking approach is considered as a *global conformance checking* technique, which views the process reference model as an accurate representation of the overall process behavior. It is assumed that the whole process is modeled and can therefore be checked. This method enables not only the identification of the deviations but also the identification of the exact source causing the problem [17]. We have chosen a global conformance checking approach as this corresponds to the medical practice of considering entire treatment processes.

Therefore, ProM [26] was used with the Multi-perspective Process Explorer (MPE) [18], which uses the multi-perspective alignment algorithm developed by Mannhardt [17]. A fundamental feature of the conformance checking algorithm is the definition of a cost function. The cost function should be defined in a way that the calculated alignments are semantically correct. We define semantically correct alignments as a meaningful and logical alignment concerning a process instance with deviations. A semantically correct alignment does not need to be an optimal alignment but should be correct in a sense that a domain expert would consider this alignment as meaningful.

First, the standard cost function is used. This cost function defines the cost as 3 for log move (delete), 2 for model move (insert), and 1 for incorrect synchronous move (data write). The standard cost function results in semantically incorrect alignments, because the alignment algorithm changes the attribute values of events to create an optimal alignment. In the medical context, this is semantically incorrect, as the data collected by the doctor represents reality and should be immutable for the algorithm. In this case, the standard cost function generates unusable alignments.

To achieve the desired result, the cost for data writes is increased such that it is higher than for the other two operations. Also, the delete cost for events that are not part of the staging process is reduced to 0, since in the course of the medical examination it is possible that multiple additional examinations are undertaken, that are needed to perform but are not depicted by the process model. Thus, costs were defined as 1 for log move (delete), and model move (insert), 0 for log move (for events not defined in the model) and incorrect synchronous move (non-data write) and 2 for incorrect synchronous move (data write).

Moreover, it is important to mention that the cost for the non-data writes was set to 0, because this allows the alignment algorithm to make insert operations that are associated with attribute values, without paying the cost for the data writes. The calculated fitness value itself was not considered, since the focus of the use case is on the calculated deviations on the event level. Due to privacy regulations it is not possible to show the resulting alignments, thus the results will be explained in a qualitative way. Two of the alignments are semantically correct. These traces correspond perfectly to the guideline but also contain medical examinations, which are not depicted in the process model, but were needed to perform. The additional undertaken examinations are deleted by the alignment algorithm, which is semantically correct since these examinations are a positive deviation from the guideline, which have the cost of 0. For the other three traces a semantically correct alignment was not possible. This is mainly due to the occurrence of events that are depicted of the process model but are occurring at other positions as expected. In this case, the alignment algorithm seeks the shortest or least expensive path through the process model and deletes correct events or inserts new events, which already have been executed. Here, the least expensive path allowed by the process model is not semantically correct in every case, as our alignments show. As stated by [17] the algorithm only shows one alignment and this is the optimal one in terms of alignment costs. However, there are also possible alignments that might be better in terms of semantics but worse in terms of alignment costs.

In summary, it was not possible to define a cost function which leads to semantically correct alignments for all traces. Nevertheless, it was possible to identify the medical examinations that are not part of the guideline but were executed by the physician.

5 Discussion

The following section discusses the results of the project and the associated problems and limitations identified. Although the domain experts attempted to provide the most heterogeneous and complex patient data possible for this case study, it should be noted that additional challenges and issues may arise as additional patient cases are examined.

Several problems, partially typical for medical data, were found in the data set used. The following issues and characteristics were identified: high variability of treatment processes, time delays, incomplete data, none-activity-data and mapping ambiguities between reference activities. The treatment histories have a **high degree of variability** typical for medical data. Patient treatment data have shown that there are activities in treatment that can occur at any time and any number of times. Thus, such activities occur more frequently than described in the guideline. These treatment activities pose a challenge in guideline compliance checking because they are explicitly mentioned in the guideline only at specific points in treatment. Consequently, guideline-compliant modeling does not represent all contingencies of medical treatment, leading to the identification of activities as deviations where they do occur additionally. Another important aspect at this point is that some of these activities may play a crucial role in the further course of treatment. For this reason, the activities must be able to occur at any time in the model and they must have paths to all possible subsequent treatments. However, based on the data collected so far, it is apparent that mapping all options would increase the complexity of the model and thus the effort to maintain it is no longer manageable.

A similar problem occurs due to **time delays** in treatment. For example, in the treatment of patients, surgical procedures are followed by histological laboratory examinations in which, e.g., tissue or lymph nodes are examined. Consequently, the obvious modeling approach is to place the laboratory testing after surgery. In practice, treatment data have shown that some time elapses between surgery and laboratory examination, and patients continue to receive treatment in the meantime. This leads to the issue of valid activities being identified as a deviation or violation. The data from the systems are **incomplete** as they only represent the clinically documented course and parts of the out-of-hospital treatment and diagnosis are missing. This is particularly evident in the data for events at the beginning of the treatment process. Although it is evident from a medical view that all patients should have passed through the same diagnostic steps, patients start with different events. This is due to the fact that parts of the treatment such as excision, histological examination, initial clinical examination, etc. were performed out-of-hospital. Parts of the ADT/GEKID data are **none-activity-data** and thus cannot be assigned to an event or timestamp. For this data, it is neither possible to determine when nor in the course of which activity it was collected. This affects the master data, which also includes attributes such as age, which are crucial for guideline recommendations. The same applies to the diagnostic data, which only reflects the current status and not the procedural progression over time. Therefore, it is not possible to track staging over the progression of treatment with ADT/GEKID. In the context of the reference model, **mapping ambiguities between reference activities** occurs in the data. Thus, there are events in the event log which could imply the execution of certain activities by numerous attributes. However, the collection of the value does not necessarily imply the use of the value and thus the execution of the activity in the process model. Standard laboratory tests, e.g., involve the collection of numerous values, including tumor markers. However, the documentation of the values does not allow any conclusion to be drawn about the observation, analysis and usage of the tumor markers. Thus, at no point in the process can it be determined whether a particular tumor marker was considered or not.

During **conformance checking**, process mining specific problems were identified in addition to the data set related ones. The following problems have been identified: semantically inappropriate control-flow alignments, semantically inappropriate data alignments and definition of cost function. The **semantically inappropriate control-flow alignments** describe a conflict between the goal of the algorithm and the medical intent. By default, the algorithm uses a cost function where aligning data values is cheaper than aligning events. As a result, patient examination values are modified during alignments, such as changing the staging value, to restore conformance. The examination values are of utmost importance for the course of treatment, but should only be modifiable by new diagnoses of the physicians and not by the algorithm. Accordingly, to produce the desired behavior, in the configuration, aligning data values is more expensive than aligning events. As a result, the sequence of events is aligned, but not in the desired way. Consequently, situations arise where the alignment changes only a single data value, e.g., making it the most favorable path for the alignment. This approach ends the patient's path as fast as possible and implies, e.g., that no melanoma was found during the initial clinical examination and the patient is discharged from the hospital. Therefore, from a medical point of view, it becomes apparent that the most favorable path represents not the best possible course of treatment. Based on this finding, further efforts should be made to examine whether the current conformance checking approach is suitable for checking medical treatment processes for guideline compliance. Since treatment courses are highly dynamic, a potentially more appropriate approach would be to examine whether a possible path of the process model can be reconstructed via sequence segments of the corresponding treatment course. Since guideline specifications only partially describe steps or sequences anyway, an alignment of sequence segments would provide a means for medical conformance checking. A suitable approach could be a *local conformance checking* technique, which describes the process of checking the conformance by using a set of independent rules regarding the process. Therefore, only specific parts of the process are checked not the process as a whole. These rules are often defined in LTL or in declarative modeling languages like Declare [4]. Furthermore, **semantically inappropriate data alignments** could be identified when performing conformance checking. These occurred when a guard was violated by an improper value. For example, a patient may receive radiotherapy after a lymphadenectomy if they have a count of three or more lymph nodes affected with cancer. In an alignment, the value was generically set to 1000, which satisfies the condition but creates semantic incorrectness. At this point, it becomes evident once again that data values should not be adaptable across the board in medical conformance checking. The medical context is highly relevant in and between treatment steps, which is why simple value alignments to satisfy guards are not sufficient. If a conformance checking algorithm should indeed have the authority to make data alignments, then semantic technologies must be used in order to draw proper conclusions and achieve meaningful results. Another problem became apparent in the attempt to define a generally valid **cost function** for the patients. Thus, although desired alignments could be achieved sporadically by changing costs, they could only ever be achieved for an individual patient. Since the medical conditions for a patient in treatment are highly dynamic and individual, it is not possible to achieve globally desired results by defining costs.

6 Conclusion

In this work, we focused on the applicability of conformance checking to determine clinical guideline compliance on clinical data. For our case study, we used real data of non-trivial treatment and diagnosis of malignant melanoma provided by Münster University Hospital and a procedural guideline representation created in collaboration with medical professionals. The data used were in the format of the ADT/GEKID data set, which is used by the German cancer registries, and enriched with data from a HIS where necessary.

We showed that it is possible to use conformance checking to verify clinical guideline conformance of real-world clinical data. Unfortunately, there are a number of application problems, mostly rooted in the data, but also in the conformance checking algorithm and the process model. In particular, the characteristically high variability of clinical treatment processes is a challenge. Both the execution and the order of execution of activities in clinical treatment processes are subject to a variety of factors, including co-morbidities, time delays in the process and patient preferences, resulting in highly variable processes. In addition, incomplete processes, e.g., when data from treatments in other organizations are not available, need to be handled. Moreover, alignments by writing attribute values or deleting activities partially resulted in semantically incorrect alignments. Further challenges lie in the preprocessing of the data, as they were inconsistent in granularity, contained activities irrelevant to conformance checking, and most importantly were documented heterogeneously and partially unstructured, requiring a complex preprocessing process.

We plan to extend the evaluation to other guidelines, including time-constraints such as follow-up care. We are also working on fitness functions based on sub-processes and an analogy-based alignment approach. In this context, we plan to further investigate the clinical data and define similarity measures for treatment-relevant parameters with medical experts. Also, we want to test other approaches such as deep-align [20] and investigate how they address the identified problems. Many problems are due to semantic violations of the alignment. Here, we are working on an ontology-supported hybrid alignment procedure that detects semantically incorrect alignments and tries to prevent them.

References

1. IEEE Standard for eXtensible Event Stream (XES) for Achieving Interoperability in Event Logs and Event Streams. IEEE Std 1849–2016, pp. 1–50 (2016)
2. Alharbi, R.F., Berri, J., El-Masri, S.: Ontology based clinical decision support system for diabetes diagnostic. In: Science and Information Conference (SAI), pp. 597–602. IEEE (2015)
3. American Society of Clinical Oncology: Good clinical practice research guidelines reviewed, emphasis given to responsibilities of investigators: second article in a series. J. Oncol. Pract. **4**(5), 233–235 (2008)
4. Caron, F.: Business process analytics for enterprise risk management and auditing (2013)
5. Dechter, R., Pearl, J.: Generalized best-first search strategies and the optimality of a*. J. ACM **32**(3), 505–536 (1985)
6. Deutsche Krebsgesellschaft, Deutsche Krebshilfe, AWMF: Leitlinienprogramm Onkologie: Diagnostik, Therapie und Nachsorge des Melanoms, Kurzversion 3.3, 2020f (30042018). https://www.leitlinienprogramm-onkologie.de/leitlinien/melanom/. Accessed 28 May 2021
7. Dunzer, S., Stierle, M., Matzner, M., Baier, S.: Conformance checking: a state-of-the-art literature review. In: Proceedings of the 11th International Conference on Subject-Oriented Business Process Management, pp. 1–10. ACM (2019)
8. Edge, S.B.: AJCC Cancer Staging Manual, 7th edn. Springer, New York (2010). https://doi.org/10.1007/978-1-4757-3656-4
9. Forsner, T., Hansson, J., Brommels, M., Wistedt, A.A., Forsell, Y.: Implementing clinical guidelines in psychiatry: a qualitative study of perceived facilitators and barriers. BMC Psychiatry **10**, 8 (2010)

10. Gatta, R., et al.: Clinical guidelines: a crossroad of many research areas. Challenges and opportunities in process mining for healthcare. In: Di Francescomarino, C., Dijkman, R., Zdun, U. (eds.) BPM 2019. LNBIP, vol. 362, pp. 545–556. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-37453-2_44
11. Geyer, T.: Process mining on CIG process model representations for clinical guidelines. Master thesis, University of Trier (2021)
12. Grimshaw, J.M., Russell, I.T.: Effect of clinical guidelines on medical practice: a systematic review of rigorous evaluations. Lancet **342**(8883), 1317–1322 (1993)
13. Holzinger, A., Dehmer, M., Jurisica, I.: Knowledge discovery and interactive data mining in bioinformatics-state-of-the-art, future challenges and research directions. BMC Bioinform. **15**(Suppl 6), I1 (2014)
14. Landfeldt, E.: Compliance to care guidelines for Duchenne muscular dystrophy. J. Neuromuscul. Dis. **2**(1), 63–72 (2015)
15. Lenz, R., Reichert, M.: It support for healthcare processes - premises, challenges, perspectives. Data Knowl. Eng. **61**(1), 39–58 (2007)
16. de Leoni, M., van der Aalst, W.M.P.: Aligning event logs and process models for multi-perspective conformance checking: an approach based on integer linear programming. In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 113–129. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40176-3_10
17. Mannhardt, F.: Multi-perspective process mining. Ph.D. thesis, Technische Universiteit Eindhoven (2018)
18. Mannhardt, F., De Leoni, M., Reijers, H.A.: The multi-perspective process explorer. BPM (Demos) **1418**, 130–134 (2015)
19. Mawoko, C.T.: Aligning data-aware declarative process models and event logs. Master's thesis, University Tartu (2019)
20. Nolle, T., Seeliger, A., Thoma, N., Mühlhäuser, M.: DeepAlign: alignment-based process anomaly correction using recurrent neural networks. In: Dustdar, S., Yu, E., Salinesi, C., Rieu, D., Pant, V. (eds.) CAiSE 2020. LNCS, vol. 12127, pp. 319–333. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49435-3_20
21. Pesic, M., Schonenberg, H., van der Aalst, W.M.: DECLARE: full support for loosely-structured processes. In: 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), p. 287 (2007)
22. Rebuge, Á., Ferreira, D.R.: Business process analysis in healthcare environments: a methodology based on process mining. Inf. Syst. **37**(2), 99–116 (2012)
23. Sackett, D.L.: Evidence-based medicine. Semin. Perinatol. **21**(1), 3–5 (1997)
24. Schrijver, A.: Theory of Linear and Integer Programming. Wiley, Chichester (1998)
25. Tobe, S.W., Hua, D., Twohig, P.: Clinical practice guidelines. In: Schiffrin, E.L., Touyz, R.M. (eds.) Hypertension, pp. 238–251. The Future Science Group eBook Collection, Future Medicine Ltd, London (2013)
26. van Dongen, B.F., de Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The ProM framework: a new era in process mining tool support. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 444–454. Springer, Heidelberg (2005). https://doi.org/10.1007/11494744_25

Patient Discharge Classification Based on the Hospital Treatment Process

Jonas Cremerius(✉), Maximilian König, Christian Warmuth, and Mathias Weske

Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
{Jonas.Cremerius,Mathias.Weske}@hpi.de,
{Maximilian.Koenig,Christian.Warmuth}@student.hpi.de

Abstract. Heart failure is one of the leading causes of hospitalization and rehospitalization in American hospitals, leading to high expenditures and increased medical risk for patients. The discharge location has a strong association with the risk of rehospitalization and mortality, which makes determining the most suitable discharge location for a patient a crucial task. So far, work regarding patient discharge classification is limited to the state of the patients at the end of the treatment, including statistical analysis and machine learning. However, the treatment process has not been considered yet. In this contribution, the methods of process outcome prediction are utilized to predict the discharge location for patients with heart failure by incorporating the patient's department visits and measurements during the treatment process. This paper shows that, with the help of convolutional neural networks, an accuracy of 77% can be achieved for the hospital discharge classification of heart failure patients. The model has been trained and evaluated on the MIMIC-IV real-world dataset on hospitalizations in the US.

Keywords: Discharge Classification · Process Outcome Prediction · Machine Learning · Heart Failure

1 Introduction

With a rehospitalization rate of up to 45% within six months of discharge, heart failure is the leading cause of rehospitalization and a significant cause of hospitalization for patients over the age of 65 in American hospitals [2,4]. This constitutes a high medical risk for the patients and leads to high expenditures and workload for hospitals and other treatment facilities patients are discharged to after treatment in the hospital. However, as the rehospitalization rate varies depending on the discharge location, the decision on the most suitable discharge location is of high importance [7].

Up to now, determining characteristics and actual prediction models for the discharge location are primarily based on statistical methods, which mainly look at the patient's state at the end of hospitalization [2,12]. The idea of this contribution is to incorporate the treatment process of heart failure patients to make

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 314–326, 2022.
https://doi.org/10.1007/978-3-030-98581-3_23

the final decision about the discharge location based on the whole treatment process. Therefore, this paper applies process outcome prediction, a business process management technique, using machine learning to predict hospital discharge locations. In practice, this approach can serve as a decision support system to choose the appropriate discharge location more accurately, as we consider the treatment process instead of merely looking at the patient's state at the end of treatment.

The remainder of the paper is structured as follows: Sect. 2 lays the theoretical foundation on heart failure, process outcome prediction, and convolutional neural networks, followed by an overview of related work. The specifics of the MIMIC-IV dataset, which we used as a foundation for the subsequent work, are covered in Sect. 3. In Sect. 4, we describe our approach and elaborate on the discharge location prediction using convolutional neural networks trained on the MIMIC-IV dataset. Results and a discussion are part of the evaluation in Sect. 5. Section 6 summarizes our contribution and outlines future work.

2 Preliminaries and Related Work

This section provides an overview of the domain of heart failure and introduces the concepts used in the remainder of this paper. Additionally, we present related work regarding patient discharge classification and process outcome prediction.

2.1 Heart Failure

Following the American Heart Association (AHA)/American College of Cardiology guidelines [10], Roger defines heart failure as "a complex clinical syndrome that can result from any structural or functional cardiac disorder that impairs the ability of the ventricle to fill or eject blood" [17]. Heart failure was chosen as the application area as this is the leading cause of rehospitalization for people older than 65 years with a rehospitalization rate within six months of up to 45% [2,4,18]. According to Howie et al., the rehospitalization risk strongly varies depending on the discharge location [2,7]. In their study, heart failure patients discharged to home or home health care had a 2.6 times higher risk of rehospitalization than those discharged to skilled nursing facilities (SNF), emphasizing the importance of the decision on the discharge location.

2.2 Process Outcome Prediction

"Business Process Management (BPM) includes concepts, methods, and techniques to support the design, administration, configuration, enactment, and analysis of business processes" [21]. The area of business process monitoring as a branch of business process management provides means to analyze events occurring during process executions, allowing for insights on the overall process and how to improve it. A subfield of business process monitoring, predictive business process monitoring, aims at making predictions about future states of current process executions based on the activities performed so far and other previously executed process instances.

One technique emerging from that field in recent years is process outcome prediction. According to Teinemaa et al., it can be defined as “classifying each ongoing case of a process according to a given set of possible categorical outcomes” with *case* in this context referring to a single process execution [19]. The advantages of this technique are better predictability and the potential to improve the decision-making during process executions [22]. Approaches to process outcome prediction are settled in the fields of statistics and supervised machine learning as a classification problem [8,19].

2.3 CNN

Convolutional Neural Networks (CNNs) are deep neural networks that are commonly used for sequence classification tasks and process outcome prediction [14,20,22]. Originally, CNNs became popular for pattern recognition in, for example, computer vision tasks, i.e., the analysis of images. A CNN architecture comprises three elements: convolutional layers, pooling layers, and fully-connected layers. Convolutional layers perform convolutions using kernels of different sizes to extract relevant high-level features from the input data, reducing dimensionality. Pooling layers are used to perform down-sampling to reduce the complexity for subsequent layers. In fully-connected layers, each node has a direct connection to every node in the next layer up to the final layer, that finally produces the output [1].

2.4 Related Work

Research has been conducted on determining factors leading to patients being discharged to different discharge locations using statistical approaches. In [12], Kobewka et al. performed a systematic review to identify models and variables with predictive power for discharge location decisions after stays in intensive care units. Their results show that age, impaired physical function, and the absence of an informal caregiver are of high importance. Similarly, Allen et al. conducted an observational analysis of heart failure patients at the age of 65 or above to determine the most relevant aspects of patients and hospitals associated with discharge to SNF [2]. Their most influential predictors are the total length of stay, patient age, different comorbidities, and gender. Apart from statistical analysis, machine learning has been applied to classify the discharge location of patients by incorporating the patient’s temperature, blood pressure, comfort, and more at the end of the treatment process [6].

In the field of process outcome prediction, Teinemaa et al. present a systematic review and taxonomy of process outcome prediction methods together with a comparative experimental evaluation of a subset of these methods [19]. The approaches taken into account primarily focus on features that are not changing throughout the process. In contrast, Le et al. introduced an approach they call *Markov sequence alignment*, which focuses on temporal features. Their method is an extension of Markov models that uses temporal categorical features extracted from past process executions to predict the following steps during process execution, as well as the process outcomes [13]. Leontjeva et al. present a multi-class sequence classification approach to incorporate constant and temporal features where a hidden Markov model or Long Short-Term Memory (LSTM) is trained on the temporal features followed by a random forest model trained on the constant features enriched by the temporal model's results [14]. Recent research has evaluated the application of CNNs in the field of process outcome prediction. In their comparison of CNN with LSTM architectures, Weytjens et al. conclude that "CNNs deliver the same results as the state-of-the-art LSTMs at a fraction of the time and can therefore be recommended as the first choice for practitioners" [22].

Applying process outcome prediction to the discharge location classification of heart failure patients allows for early resource allocation for the discharge facilities due to the improved predictability during a patient stay. Patients currently undergoing treatment in a hospital can be assigned the most probable discharge location, allowing treatment facilities and services, such as SNF, to predict their workload better and adjust their resource planning and staffing accordingly. Furthermore, process outcome prediction enables a process oriented decision-making by making the decision not only based on the patient's state at the end of the treatment but also on the development of the patient's state during the treatment process.

3 Dataset

We use the Medical Information Mart for Intensive Care (MIMIC)-IV database [11] as a data foundation for the discharge location prediction. The database is publicly available on PhysioNet [5] (authorized access due to privacy regulations - see license[1]) and contains information on over 40,000 patients admitted to the Beth Israel Deaconess Medical Center in Boston, Massachusetts, from 2008 to 2019. The data is stored in a relational database format. All information was de-identified by obfuscating the exact time of events while retaining their chronological order, which allows for the application of process mining and process outcome prediction.

The MIMIC-IV dataset consists of 35 tables in which, amongst other information, the following patient data is stored: Demographic information on patients, such as their age and marital status, transfers between departments during their stay, as well as the medications they received in each of them. Furthermore, various information on diagnoses is provided, e.g., International Classification of Diseases (ICD) codes, Diagnosis-related Group (DRG) codes, and laboratory values resulting from laboratory tests for patients, e.g., hemoglobin, creatinine, and urea nitrogen values.

[1] https://physionet.org/content/mimiciv/view-license/0.4/.

4 Contribution

The contribution is presented in three steps. First, we describe the process of selecting the cohort. Second, the steps of feature selection and preprocessing are explained. Lastly, we describe the architecture of the prediction model.

4.1 Cohort Selection

The cohort of patients was selected based on the diagnosis, and the DRG of the hospital stay to identify patients where heart failure was treated. The dataset stores diagnoses as so-called ICD codes. Thus, we selected all patients who had a heart failure related diagnosis as their primary diagnosis. Today, two ICD coding systems co-exist in hospitals, which are ICD-9 and ICD-10. For ICD-9, the codes starting with 428 are related to heart failure, whereas for ICD-10, the codes starting with I50 are heart failure related.

Second, we used DRG codes to identify only those patients whose primary reason for hospitalization was heart failure. DRG codes correspond to the main reason for a patient's stay at the hospital. All cardiac related DRG codes[2] were considered, which can be seen in the script for data extraction from the MIMIC-IV database[3].

With the combination of a heart failure primary diagnosis and a cardiac-related DRG, it is known that the patients suffered from heart failure and that this was the primary reason for their hospitalization. Filtering for these characteristics, the dataset provides a total of 12,306 stays of 7,693 patients.

The discharge location is stored for each patient stay, with 13 different discharge locations available. The three most frequent discharge locations for heart failure patients found in the MIMIC-IV dataset are home (3,430 stays, 27.9%), home health care (4,982 stays, 40.5%), and SNF (2,323 stays, 18.9%). As the other discharge locations have a frequency of less than 4%, we decided to focus on the discharge locations listed above to have a sufficient sample size for each class for model training and testing.

Since the discharge to SNF is associated with high costs and workload for medical personnel, we also decided to make predicting discharge to SNF the primary goal of our classification models. Additionally, there is a need to better characterize the patient population being discharged to SNF [2]. This reduces the complexity to binary classification where we predict SNF vs. others (representing discharge to home or home health care).

Considering only heart failure patients discharged to the three most frequent discharge locations, the resulting number of patient stays serving as data points for model training, validation, and testing is 10,725.

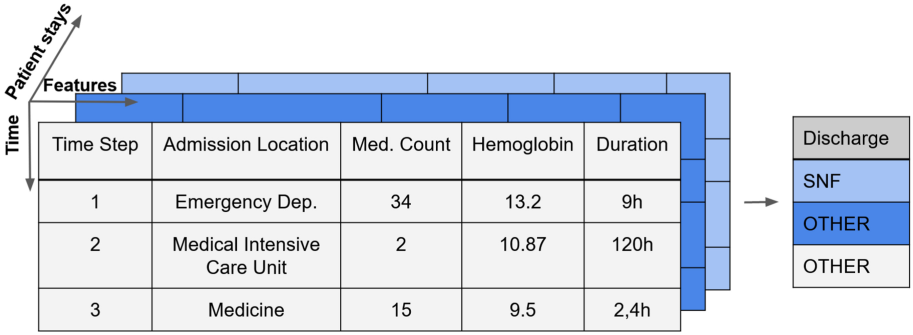
[2] https://www.hcup-us.ahrq.gov/db/nation/nis/APR-DRGsV20MethodologyOverviewandBibliography.pdf.

[3] https://github.com/christianwarmuth/treatment-based-patient-discharge-classification.

4.2 Feature Selection and Data Preprocessing

In order to represent the process behind the data, i.e., the order of departments each patient visits during a stay, the data is required to be in a three-dimensional shape. We consider the different features across multiple time steps for each patient stay individually. A visualization of the data's shape is shown in Fig. 1. This allows us to combine features that do not change during a patient's stay, such as age and gender, with features that may be different for every department visit, such as the length of stay there and laboratory values measured in a department.

Initially, the features taken into account were selected based on the literature presented in Sect. 2.4. This selection included the total length of stay in the hospital, patient age, and gender, which do not change during a stay. Furthermore, the selection of variables included the stay duration, the med count representing the number of medications received, and the lab count representing the number of laboratory values resulting from analyses conducted, which are different for each stay in a department. Starting with these features, we tested and compared multiple combinations with additional features and their impact on the predictive performance, resulting in the final feature selection shown in Table 1. In addition to the aforementioned features, incorporating information on the patient's insurance situation, marital status, ethnicity, and the number of ICD codes associated with them, meaning the number of different diagnosed disease patterns, lead to improved predictive performances. Also, taking laboratory values such as creatinine, hemoglobin, red blood cells, glucose, and urea nitrogen into account resulted in higher accuracy.



Time Step	Admission Location	Med. Count	Hemoglobin	Duration
1	Emergency Dep.	34	13.2	9h
2	Medical Intensive Care Unit	2	10.87	120h
3	Medicine	15	9.5	2,4h

Discharge
SNF
OTHER
OTHER

Fig. 1. Visualization of the data shape

Given the set of features, the raw data had to be preprocessed to fulfill the shape and data type requirements of the models to train. Categorical features such as the department visited, gender, and marital status were one-hot encoded to represent them as numerical values that can serve as input for machine learning analyses. This also prevents the introduction of non-existent ordering between the items [16]. To avoid potential biases from different value ranges, all numerical features were standardized by scaling them to zero mean and unit variance. In addition, many model architectures require each sequence of departments to have the same length. Therefore, each sequence shorter than a specified length was padded with null values while longer sequences were cut off. The value of the sequence length was derived from the distribution of the number of departments visited by patients during a single stay.

Due to the disparity in the number of samples between patient stays resulting in a discharge to SNF and those resulting in other discharge locations, we also decided to use balancing techniques. On the one hand, we applied over- and downsampling, which were found to be effective methods in dealing with class imbalance [9]. That means we randomly duplicated patient stays where the patient was discharged to SNF and randomly removed patient stays resulting in another discharge location until both cases were represented equally. On the other hand, we introduced class weighting to model training. Thereby, instances of the underrepresented class, i.e., discharge to SNF, are multiplied with a weighting factor in the loss function, increasing the penalty for misprediction. While both techniques improved the predictive performance of our model, especially with regard to the confusion matrix, class weighting yielded better results in our case, which is why we chose this technique for our final model.

Table 1. Final selection of features incorporated in the CNN model

Demographic Information	Lab Values	Stay Information
• Patient age	• Creatinine	• Department
• Gender	• Hemoglobin	• Admission Location
• Insurance	• Urea Nitrogen	• Transfer duration
• Marital Status	• Glucose	• No. of medications received
• Ethnicity	• Red Blood Cells	• No. of lab values measured
		• No. of ICD codes

4.3 Model Selection and Training

We chose CNNs as our model architecture and trained all models on the preprocessed data for patient discharge classification. In order to get the best model, we then applied hyperparameter tuning. We defined multiple hyperparameters such as the kernel sizes of the convolutional layers, the size of the fully connected layers and pooling layers, and the intermediate activation functions. Each of these hyperparameters was assigned a range of possible values. Multiple models were trained with the hyperparameter optimization approach tree-structured Parzen estimator [3]. The best model parameters were chosen based on the F1-score on the validation part of the dataset. Afterwards, the models were analyzed using accuracy, precision, recall, and confusion matrices.

The final model consists of two 1D-convolutional layers followed by a dropout and a max-pooling layer. The result is flattened and then serves as input for a sequence of four fully connected layers and the output layer using the sigmoid activation function.

5 Evaluation and Discussion

The evaluation starts with the results of model training and validation, followed by a discussion about the resulting model's feature importance and limitations.

5.1 Results

The code to reproduce our results, including the result of the hyperparameter search, can be found on GitHub[4]. Please note that due to data privacy restrictions for the MIMIC-IV database, you will have to get access to the database. The solution is implemented in Python, and the README of the linked repository provides instructions on how to run the experiments.

We split our preprocessed dataset into a train and a test set, with the latter accounting for 25% of the dataset (2684 patient stays). The metrics used to compare the models resulting from our hyperparameter tuning were calculated on a validation set consisting of 10% of the train set after training on the remaining part of the train set. They comprise the following: The *accuracy* represents the proportion of data points assigned to the correct discharge location. *Precision* reflects the fraction of correct predictions of discharge to SNF over all predictions of discharge to SNF. In contrast, *recall* shows the percentage of how many of the patient stays that resulted in discharge to SNF were predicted as such. The *F1-score* then is the harmonic mean of precision and recall. Another metric, the *Area Under ROC Curve (AUROC)* is the probability of a randomly chosen positive data point (discharged to SNF) being ranked higher by the model than a randomly chosen negative data point. *Confusion matrices* show for each true label on the y-axis the distribution of the correctly or incorrectly predicted labels on the x-axis. If the model predicted everything correctly, the diagonal from upper left to bottom right would contain only values of 1.0.

The final model reaches an accuracy of 77% with a weighted precision of 81% and a weighted recall of 77%, respectively. The F1-score is 0.78, and the AUROC is 0.73. As shown in the confusion matrix in Fig. 2, there is a discrepancy of about 14% between the accuracy of predicting SNF as discharge location on the one hand and the accuracy of predicting other discharge locations on the other hand.

Figure 3 shows the feature importance of our model as a beeswarm plot using SHAP values (SHapley Additive exPlanations) [15]. The graph was generated using the SHAP library[5]. It shows the impact of the 18 most influential features. Each dot for each feature corresponds to a single patient stay. The x-axis shows

[4] https://github.com/christianwarmuth/treatment-based-patient-discharge-classification.

[5] https://github.com/slundberg/shap.

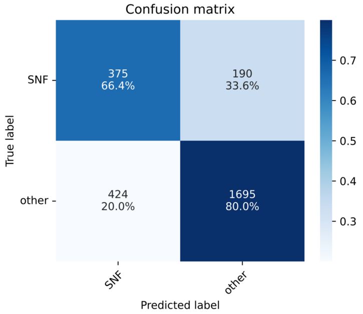


Fig. 2. Confusion Matrix of the final CNN model

how much impact those features had, with high negative values indicating a high impact on the decision to SNF as discharge location, high positive values indicating the opposite. The color of a dot represents the value of the feature, red representing a high value, blue a low value. Since our data had a three-dimensional shape, which could not be represented in this graph, we averaged the SHAP values and the feature values for each patient stay. For example, the distribution of dots for the patient age shows that a higher age often serves as a predictor for discharge to SNF.

5.2 Discussion

This contribution suggests considering the treatment process in classifying the discharge location of heart failure patients. Looking at the feature importance in Fig. 3, features changing throughout the process have a significant effect on predicting the discharge location. For example, the development of the laboratory values creatinine, hemoglobin, glucose, red blood cells, and urea nitrogen impact the prediction. Furthermore, the number of medications received and the number of laboratory values measured per department are relevant. Interestingly, a higher number of medications indicates a discharge to SNF, whereas a higher number of laboratory values indicates a discharge to other locations.

The departments visited and the admission locations also affected the outcome, as the admission from SNF resulted in a higher probability of being discharged to SNF. Being referred by a physician to the hospital impacts the discharge decision to home/home health care. A visit to the Medicine/Cardiology department has only a slight influence on the prediction. The transfer duration, representing the length of stay in each department, helps to predict the discharge location.

Additionally, we were able to confirm relevant factors as proposed in current literature, which includes age, insurance, length of stay, gender, and laboratory values (creatinine, urea nitrogen, and hemoglobin). Information about the availability of an informal caregiver is provided in the form of the marital status in

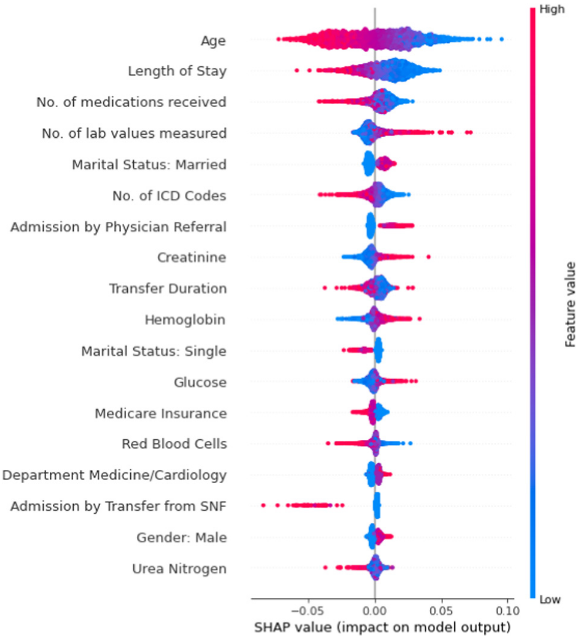


Fig. 3. Beeswarm plot for the final CNN model

the MIMIC database, which constitutes a relevant factor [2]. However, marital status is only an indicator and does not represent the guaranteed availability.

With our process oriented approach, we emphasize to incorporate the development of the patient's state throughout the treatment process in the decision-making. As we identified process characteristics in the different cohorts, a more precise discharge classification can be achieved by incorporating the treatment process. It should be noted that we identified patient characteristics regarding discharge classification based on decisions made by healthcare professionals in the past. Thus, we only reproduce the decision-making of healthcare professionals. Nevertheless, the identified characteristics can be further investigated to improve the decision-making, for example, why patients with increased creatinine get discharged to home/home health care and not to SNF.

Looking at the results in Fig. 2, our model is better in predicting discharge to other locations (80%) than to SNF (66%), resulting in an overall accuracy of 77%. We assume that better predictive performance could have been achieved with a larger sample size, as a sufficiently large sample size can significantly impact the predictive performance of machine learning models [22]. We performed training on other models, such as LSTM and XGBoost, while CNNs turned out to provide the best results. Comparing our results to recent research is difficult, as the discharge locations are different among the datasets. To our knowledge, there is no respective model using the MIMIC dataset yet. However, we could confirm the already identified patient characteristics as described above.

Furthermore, a more detailed view of the process could improve the results of our model, as we did not incorporate a comprehensive view of the patient's diagnoses, medications, laboratory values, and the procedures performed on them. Additionally, the mental status and further sociodemographic information could help to improve the model's performance.

6 Conclusion and Future Work

This paper discusses the approach of predicting the discharge location for heart failure patients by incorporating the treatment process.

We have shown that the development of the patient's state during the process and the respective visits in the hospital departments have a considerable impact on the discharge location prediction. Therefore, taking into account the treatment process instead of merely looking at the patient's state at the point of discharge can serve as a decision support for healthcare professionals.

An accuracy of 77% could be achieved in this contribution, which is a promising result, but still leaves room for improvement. Therefore, future work could be conducted by combining the MIMIC-IV dataset with other datasets such as the HiRID[6] database to increase the sample size and improve the prediction results. Furthermore, a more comprehensive representation of the treatment process might help to increase the accuracy by adding detailed information on medications received or procedures performed. Besides, it would be worthwhile to consult domain experts who could point towards additional features not yet considered.

References

1. Albawi, S., Mohammed, T.A., Al-Zawi, S.: Understanding of a convolutional neural network. In: 2017 International Conference on Engineering and Technology (ICET), pp. 1–6 (2017). https://doi.org/10.1109/ICEngTechnol.2017.8308186
2. Allen, L.A., et al.: Discharge to a skilled nursing facility and subsequent clinical outcomes among older patients hospitalized for heart failure. Circ. Heart Fail. **4**(3), 293–300 (2011)
3. Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. In: Proceedings of the 24th International Conference on Neural Information Processing Systems, NIPS 2011, pp. 2546–2554 (2011)
4. Desai, A.S., Stevenson, L.W.: Rehospitalization for heart failure: predict or prevent? Circulation **126**(4), 501–506 (2012)
5. Goldberger, A., et al.: PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. Circulation **101**(23), e215–e220 (2000)
6. Gramaje, A., Thabtah, F., Abdelhamid, N., Ray, S.: Patient discharge classification using machine learning techniques, June 2019
7. Howie-Esquivel, J., Spicer, J.G.: Association of partner status and disposition with rehospitalization in heart failure patients. Am. J. Crit. Care **21**(3), e65–e73 (2012)

[6] https://physionet.org/content/hirid/1.1.1/.

8. Huang, Z., Dong, W., Ji, L., Duan, H.: Outcome prediction in clinical treatment processes. J. Med. Syst. **40**(1), 1–13 (2016)
9. Japkowicz, N., et al.: Learning from imbalanced data sets: a comparison of various strategies. In: AAAI Workshop on Learning from Imbalanced Data Sets, vol. 68, pp. 10–15. AAAI Press, Menlo Park (2000)
10. Jessup, M., et al.: 2009 focused update: ACCF/AHA guidelines for the diagnosis and management of heart failure in adults: a report of the American College of Cardiology Foundation/American Heart Association Task Force on Practice Guidelines: developed in collaboration with the International Society for Heart and Lung Transplantation. Circulation **119**(14), 1977–2016 (2009)
11. Johnson, A., Bulgarelli, L., Pollard, T., Horng, S., Celi, L.A., Mark, R.: MIMIC IV (version 0.4). PhysioNet (2020)
12. Kobewka, D.M., et al.: Predicting the need for supportive services after discharged from hospital: a systematic review. BMC Health Serv. Res. **20**(1), 1–10 (2020)
13. Le, M., Gabrys, B., Nauck, D.: A hybrid model for business process event and outcome prediction. Expert. Syst. **34**(5), e12079 (2017)
14. Leontjeva, A., Kuzovkin, I.: Combining static and dynamic features for multivariate sequence classification, pp. 21–30, October 2016
15. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: Guyon, I., et al. (eds.) Advances in Neural Information Processing Systems, vol. 30, pp. 4765–4774. Curran Associates, Inc. (2017)
16. Potdar, K., Pardawala, T., Pai, C.: A comparative study of categorical variable encoding techniques for neural network classifiers. Int. J. Comput. Appl. **175**, 7–9 (2017). https://doi.org/10.5120/ijca2017915495
17. Roger, V.L.: Epidemiology of heart failure. Circ. Res. **113**(6), 646–659 (2013)
18. Ross, J.S., et al.: Statistical models and patient predictors of readmission for heart failure: a systematic review. Arch. Intern. Med. **168**(13), 1371–1386 (2008)
19. Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M.: Outcome-oriented predictive process monitoring: review and benchmark. ACM Tran. Knowl. Discov. Data (TKDD) **13**(2), 1–57 (2019)
20. Weinzierl, S., et al.: An empirical comparison of deep-neural-network architectures for next activity prediction using context-enriched process event logs (2020)
21. Weske, M.: Business Process Management: Concepts, Languages, Architectures, 3rd edn. Springer, Heidelberg (2019). https://doi.org/10.1007/978-3-642-28616-2
22. Weytjens, H., De Weerdt, J.: Process outcome prediction: CNN vs. LSTM (with Attention). In: Del Río Ortega, A., Leopold, H., Santoro, F.M. (eds.) BPM 2020. LNBIP, vol. 397, pp. 321–333. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-66498-5_24

Combining the Clinical and Operational Perspectives in Heterogeneous Treatment Effect Inference in Healthcare Processes

Sam Verboven[1](✉) and Niels Martin[2,3](✉)

[1] Data Analytics Laboratory, Vrije Universiteit Brussel, Pleinlaan 2,
1050 Brussels, Belgium
sam.verboven@vub.be

[2] Research Group Business Informatics, Hasselt University, Martelarenlaan 42,
3500 Hasselt, Belgium
niels.martin@uhasselt.be

[3] Research Foundation Flanders (FWO), Egmontstraat 5, 1000 Brussels, Belgium

Abstract. Recent developments in causal machine learning open perspectives for new approaches that support decision-making in healthcare processes using causal models. In particular, Heterogeneous Treatment Effect (HTE) inference enables the estimation of causal treatment effects for individual cases, offering great potential in a process mining context. At the same time, HTE literature typically focuses on clinical outcome measures, disregarding process efficiency. This paper shows the potential of jointly considering the clinical and operational effects of treatments in the context of healthcare processes. Moreover, we present a simple pipeline that makes existing HTE machine learning techniques directly applicable to event logs. Besides these conceptual contributions, a proof-of-concept application starting from the publicly available sepsis event log is outlined, forming the basis for a critical reflection regarding HTE estimation in a process mining context.

Keywords: Heterogeneous Treatment Effect · Process Mining · Machine Learning · Event Log

1 Introduction

Process mining techniques aim to extract valuable insights from process execution data captured in an event log [1]. As it starts from data entries representing real-life behaviour, instead of the assumed or ideal behaviour [1], process mining offers evidence-based insights in processes [20]. Within the healthcare domain, process mining techniques have been used for various use cases, such as automatically discovering the order of activities, assessing whether clinical guidelines have been followed, or identifying bottlenecks in a healthcare process [24].

While process mining in healthcare often focuses on conveying process insights to practitioners based on historical data, there is increasing awareness

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 327–339, 2022.
https://doi.org/10.1007/978-3-030-98581-3_24

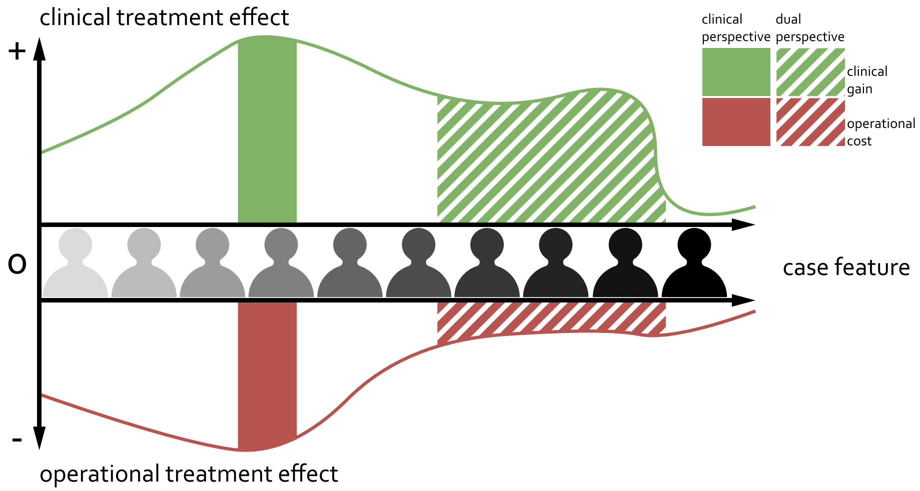


Fig. 1. The top graph y-axis depicts the positive clinical treatment effect. The bottom graph y-axis depicts the negative operational treatment effect (i.e., the operational cost). Along the shared x-axis, a case feature value is varied. Green areas below the treatment effect curve represent the clinical gain for a given policy. Red areas represent the operational cost. Filled areas represent the policy if we only take the clinical treatment effect into account. Dashed areas represent the policy when case-level process efficiency effects are also taken into account. Both clinical and operational effects can be estimated from event logs. In this example, taking individual operational efficiency effects into account more than doubles the total clinical effect. (Color figure online)

of the need for a complementary set of proactive techniques that can instigate actions in active processes [20]. This awareness, combined with recent developments in causal machine learning, opens perspectives for new approaches that support decision-making in healthcare processes using causal models.

Causal approaches in healthcare processes are confronted with three challenges. First, the effect of the same process intervention (e.g., the execution of a particular activity) can vary widely across patients. Nonetheless, current intervention guidelines are often developed at the population level and, hence, tuned to the *average* case. However, the goal in healthcare process management is evolving towards determining the optimal intervention for any case. Secondly, when causal models consider treatment effects at the patient level, there is a predominant focus on clinical outcome measures, with no regard for process efficiency. In practice, clinical and operational measures are not independent from each other. For instance, while a process intervention may be desirable from a clinical perspective (e.g., reduced likelihood for a particular adverse event), it might have negative implications from an operational point of view (e.g., in terms of ICU length-of-stay). Moreover, increased operational efficiency can also lead to improved clinical outcomes, as more patients receive treatment. Finally, causal models require assumptions to be made based on a priori domain knowledge [22]. In other words, for models to have a causal interpretation, causal theory needs to be taken into account, preferably before data gathering.

Against the background of these three challenges, this paper explores the potential of Heterogeneous Treatment Effect (HTE) inference within the context of healthcare processes. Recent advances in causal machine learning enable the estimation of the causal treatment effect at the level of an individual using observational data. Consequently, event logs qualify as input for Heterogeneous Treatment Effect (HTE) modelling. In process mining, a treatment can represent any intervention within a healthcare process such as admitting a drug, executing selected activities in a specific order, or letting a particular resource perform an activity. Typical clinical outcomes include general life expectancy measures (e.g., expected days of survival [4]), and disease-specific parameters (e.g., tumor size [6]). Besides being suitable for HTE modelling, event logs also include important clues regarding the operational efficiency of a healthcare process (e.g., the length of stay or the resource involvement). This paper introduces a joint perspective on clinical and operational efficiency. The importance of adopting this joint perspective is illustrated conceptually in Fig. 1. The estimated operational and clinical treatment effects support crucial decisions within resource-constrained healthcare processes. This way, using HTE estimation provides detailed insights into the potential trade-offs between objectives are provided at the case level. A proof-of-concept application is presented using the publicly available sepsis event log [18].

The remainder of this paper is structured as follows. Section 2 introduces HTE inference and discusses the related work. Section 3 presents how HTE inference can be used in healthcare processes. In Sect. 4 a proof-of-concept is presented within the context of the sepsis event log. The paper ends with a discussion in Sect. 5 and a conclusion in Sect. 6.

2 Background

2.1 Heterogeneous Treatment Effects

The goal of HTE estimation is the estimation of the causal effect of a treatment $W \in \{0, 1\}$ on an outcome $Y \in \mathbb{R}$ for an individual i characterised by features $\mathbf{X} \in \mathcal{X} \subset \mathbb{R}^n$, where $\mathcal{X}$ denotes the n-dimensional universe of features. We adopt the standard causal effect formulation in line with the standard Rubin/Neyman Potential Outcomes Framework [25]. In the binary setting, there are two potential outcomes (POs), Y_0 and Y_1, that signify the outcomes when $W = 0$ and $W = 1$, respectively. The HTE can then be specified as:

$$\tau(\mathbf{x}) := \mathbb{E}[Y_1 \mid \mathbf{x}] - \mathbb{E}[Y_0 \mid \mathbf{x}] = \mathbb{E}[Y_1 - Y_0 \mid \mathbf{x}]. \tag{1}$$

From hereon, we will refer to $\tau(\mathbf{x})$ as the HTE.

Methods for HTE Estimation with Observational Data. From a machine learning point of view, two central elements distinguish HTE estimation (sometimes referred to as CATE/ITE) from a standard supervised learning problem. First, the HTE is unobservable for any individual, also referred to as the fundamental problem of causal inference [12]. For instance, when we execute an extra activity (treatment) in the process for an individual, we only observe the throughput time (outcome) with the extra step. For an individual, when $W = 1$ we observe Y_1, when $W = 0$ we observe Y_0, never both. Effectively, HTE models estimate something that cannot be observed directly. To still estimate $\tau(\mathbf{x})$, the dominant estimation strategy involves joint modelling of both POs in a multitask neural network with one output per potential outcome. An estimate of the HTE is then constructed as the difference between PO estimates [26].

Second, standard supervised learning methods cannot handle treatment assignment policies that are not uniformly random, i.e., datasets with assignment bias. Assignment bias thus arises in an observational dataset when the propensity to receive treatment depends on the characteristics of individuals. In reality, this is almost always the case. For example, people with a more advanced stage of cancer will have a higher propensity to receive more radical treatment options. As such, treatment assignment bias induces the treated and untreated distributions to differ. In machine learning literature, this is called covariate shift [27]. Most algorithms for HTE estimation from observational data include some component to counteract such covariate shift. Examples of such components include inverse propensity weighting, propensity score matching [16], PPM [26].

Assumptions for HTE Estimation with Observational Data. Even though machine learning methods have been designed to tackle both aforementioned challenges, not all requirements can be validated or learned directly from the data. To guarantee that the treatment effect can be identified in the Rubin-Neyman PO framework, the following standard assumptions are made:

Assumption 1 (Stable Unit Treatment Value (SUTVA)). *First, there cannot be spillover effects between the potential outcomes of individuals in different treatment groups. Second, each unit is assumed to be presented with identical versions of each treatment. Third, we observe through the factual outcome Y the potential outcome associated with the assigned treatment.*

For example, Frank's hospital stay length (outcome) should not depend on whether Sarah received antibiotics (treatment), and the antibiotics both would receive are the same. When Frank is assigned treatment ($W = 1$), we observe potential outcome Y_1. This assumption is usually validated based on expert knowledge.

Assumption 2 (Overlap). *For all individuals $\mathbf{x} \in \mathbb{R}^n$, and all treatments $W \in \{0, 1\}$, the following holds:* $0 < p(W|\mathbf{x}) < 1$.

Overlap implies that for the whole feature support region every instance has a non-zero probability of receiving treatment. Intuitively, if there are no examples of both potential outcomes for some regions of x, we cannot reliably estimate the causal effect for those x.

Assumption 3 (No hidden confounders). *This assumption implies that all variables that impact both treatment assignment and outcome are observed. As such,* $(Y_0, Y_1) \perp\!\!\!\perp W|\mathbf{x}$.

For example, to assess the effect of regular walking on mortality, a straightforward confounder is health status [11]: individuals with poor health walk less – effect on treatment assignment – and have higher chances of dying – effect on the outcome. Not including health status would lead a model to overestimate the causal effect of walking on health. Hence, in this context, it is crucial to collect health status data to avoid confounding bias.

Assumptions 2 and 3 together constitute *strong ignorability* given a set of covariates. When both SUTVA and strong ignorability hold, estimation of causal effects based on the factual outcomes in observational data is possible [25]. The assumptions regarding hidden confounders and SUTVA are fundamentally untestable based on observational data alone [14]. As such, expert knowledge plays a crucial role in HTE inference. The no hidden confounders assumption is the most difficult to satisfy. But as the dimensionality of X increases, the larger the probability that hidden confounders are observed. Consequently, a practical heuristic would be to gather as many features as possible in future event logs. This guideline facilitates causal learning, but stands in contrast with current process mining practices of narrow data gathering, often limited to which activities have been executed for a patient and when they were executed.

2.2 Related Work

HTE Estimation in Healthcare. Causal effect estimation allows us to address questions such as 'how effective is a given treatment in curing this person?' and 'which treatment is more effective for this specific individual?'. Such questions are of critical importance in clinical decision-making. Moreover, recent availability of electronic healthcare records (EHR) and methodological advances have spurred increased interest in HTE inference as a clinical tool [3,4,6,22].

Previous work for healthcare solely considers purely clinical outcomes of actions. However, it has been shown in a business context that taking into account costs greatly improves total profit [5,32]. Similarly, it makes sense to account for overarching operational objectives. While the average treatment effect has been studied for multiple clinical outcomes (e.g., [17]), no existing work to our knowledge combines both operational and clinical effects of the same treatment.

Causality in Process Mining. Within the process mining field, there has been growing interest in the identification of causal patterns from an event log. This interest is exemplified by approaches developed to conduct root-cause analysis [10,28,31], even though they focus on finding characteristics that are correlated

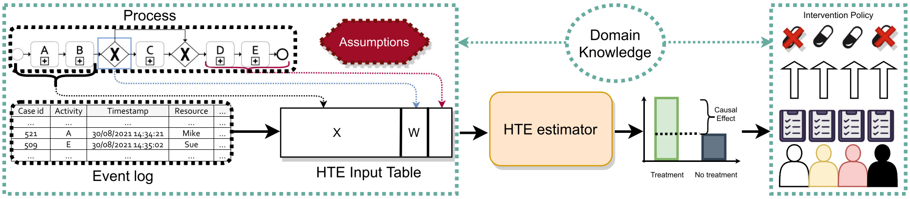


Fig. 2. Basic process flow for using causal models starting from event logs. An HTE input table can be constructed from an event log, allowing the application of standard causal machine learning methods. Domain knowledge plays a vital role in the determination of data collection, the intervention point, the validation of the assumptions, and final policy guidance.

with certain phenomena, without assessing whether the observed correlations are causal in nature. In contrast, Hompes et al. [13] and Narenda et al. [21] identify causal relationships at the process level starting from an event log using the Granger causality test and structural causal models, respectively.

Limited research has considered causal effects at a case level in the process mining field. Qafari and van der Aalst [23] use counterfactual reasoning to detect statements indicating why an undesirable outcome has happened for a particular case. Bozorgi et al. [7] also focus on the case level by proposing a technique that provides case level recommendations of treatments. The technique generates candidate treatments using action rule mining, after which an uplift tree and associated rules are generated for each candidate treatment. They apply their approach within the context of a loan application context [7].

Our work extends existing work on causality in process mining in general and HTE inference in particular, by jointly considering clinical treatment effects and operational treatment effects at the case level. Moreover, we formalise a simple pipeline that makes existing HTE machine learning techniques directly applicable to event logs.

3 Heterogeneous Treatment Effect Inference in Healthcare Processes

Definition 1 (Event, Trace and Event Log). *Let $\mathcal{A}$ represent the universe of attributes. An event $e \in \mathcal{A} \nrightarrow \mathcal{X}$ is an assignment of values to attributes. Let $\mathcal{E} = \mathcal{A} \nrightarrow \mathcal{X}$ represent the universe of events. A trace $t \in \mathcal{E}^*$ is a sequence of events referring to the same case c. Let $\mathcal{T} = \mathcal{E}^*$ represent the universe of traces. An event log L collects the traces of a set of cases, i.e., $L \subset \mathcal{T}$.*

Definition 2 (HTE input table). *Given an event log L, $\mathcal{X}$ represents the universe of features which can be calculated over L. Let $f \in L \rightarrow \mathcal{X}$ be a feature function assigning values x calculated over L. Then, the HTE input table I consists of a set of entries γ, one entry $\forall c \in L$. Each entry $\gamma = (X_1, X_2 \ldots, X_n, W, Y)$, where $X_1, \ldots, X_n \in \mathcal{X}$, $W \in \{0, 1\}$ represents whether the treatment has been assigned, and Y represents the value of the outcome measure.*

Under Assumptions 1 to 3, the HTE is identifiable and can be estimated using causal machine learning algorithms based on the HTE input table. Then, using Definitions 1 and 2, we can featurise the event log such that it translates to a standard set-up that facilitates use of all state-of-the-art machine learning algorithms for HTE estimation. A visual depiction of this pipeline can be found in Fig. 2.

4 Proof-of-Concept: Sepsis Event Log

4.1 Case Description

Healthcare process management benefits from a joint perspective on operational and clinical objectives of interventions. In many real-world healthcare processes, there is an apparent conflict between operational and clinical objectives. For instance, from a purely clinical point of view, extended hospital stay and extensive treatment with close supervision of clinicians is often optimal. From an operational perspective, typical process efficiency measures (e.g., throughput time) are improved with shorter treatment and earlier discharge. In reality, all healthcare processes are resource-constrained to some extent. Even when not explicitly considered, choices are made on the efficiency – effectiveness plane. HTE modelling allows mapping of the effects on both dimensions at the individual level, improving decision-making.

We empirically illustrate the potential of HTE inference in healthcare processes based on event logs, using the publicly available sepsis event log [18]. This event log contains events related to the trajectory of 1050 patients admitted to the emergency department (ED) of a Dutch hospital with sepsis symptoms [19]. The activities included in the event log relate, amongst others, to the moment when registration and triage took place, when laboratory results were recorded in the system, when antibiotics or liquid were administered and when the patient was discharged from the ED. Moreover, several parameters recorded in the triage document are available as event attributes. Finally, the observation that swift treatment with antibiotics is *always* advised according to the clinical guideline, but not applied in almost half the cases, illustrates the relevance of operational efficiency limits in treatment assignment [19].

4.2 Data Setup

Due to the fundamental problem of causal inference, the ground truth HTE is unobservable, and only one of the potential outcomes – i.e., the factual outcome – is ever observed. Consequently, we cannot directly assess HTE generalization performance based on factual data alone (e.g., using MSE, MAE). Furthermore, the factual outcome distribution reflects biased treatment selection. Hence, a biased model will perform better than a model that successfully corrects against assignment bias.

Table 1. Evaluation axes of Heterogeneous Treatment Effect models.

	Model Capacity	Counterfactual Estimation
Data	Original	Semi-synthetic
Metric type	Standard (e.g., MSE)	Specialised (e.g., PEHE)

These observations are reflected in the standard quantitative evaluation strategies for HTE estimators (Table 1). These strategies separately assess (i) the functional capacity to model the underlying response functions, i.e., whether the model can overfit the factual data, and (ii) its counterfactual estimation capabilities, i.e., whether the model correctly handles assignment bias to yield unbiased estimates of the potential outcomes. While (i) can be evaluated on the factual data, (ii) by definition requires a (semi-)synthetic setup. In such a setup the original features and treatment assignment are retained, but the potential outcomes are simulated [2,4,8,14,26,29]. This way, the original assignment bias and feature structure in the dataset stay intact, while allowing quantitative evaluation of the HTE model with the Precision in Estimation of Heterogeneous Effects (PEHE) measure.

$$PEHE = \frac{1}{N}\sum_{i=1}^{N}(HTE_i - H\hat{T}E_i)^2 \tag{2}$$

In line with Alaa and van der Schaar [2], the data generating model for the clinical potential outcomes is specified by: $f_{c_0}(x) = \epsilon + \exp\left(\left(x + \frac{1}{2}\right)\varOmega\right)$, and $f_{c_1}(x) = \epsilon + \varOmega x - \omega$, for no treatment and treatment, respectively. The regression coefficients are comprised by $\varOmega$, and sampled uniformly from [0, 0.1, 0.2, 0.3, 0.4]. $\epsilon \sim \mathcal{N}(0,1)$ adds i.i.d. sampled zero-centered additive Gaussian Noise to the potential outcomes. Finally, ω is selected such that the average treatment effect of the simulated clinical outcome matches the original sepsis event log data.

Next, we use the same functional form for the data generating model for the potential outcomes of the operational model, f_{o_0} and f_{o_1}. We sample regression coefficients comprised by $\varOmega$ from [0, −0.1, 0.2, −0.3, 0.4]. Furthermore, ω is selected such that the operational cost of treating is always positive. After all, doing nothing should be cheapest. The synthetic operational efficiency effect has a mean of 0.69, a standard deviation of 0.17, and ranges from 0.22–1.28. Note that the original feature structure and treatment assignment bias from the sepsis log are once more retained.

4.3 Model Setup

We use `cfrnet` [26], a popular neural network-based HTE estimator that uses an integral probability metric in its loss function to explicitly balance the covariate distribution of the treated and untreated group within a learned shared representation. After deletion of observations with missing values, 642 patient observations are retained. We hold out 100 observations for validation, 200 for testing, and use the rest for training. `Cfrnet` is run twice, once for the clinical outcome, and once for the operational outcome, yielding estimates of $\hat{HTE}_C$ and $\hat{HTE}_O$, respectively. We use the same hyperparameters as reported in Shalit et al. [26].

We validated that `cfrnet` performs well for both model capacity and counterfactual estimation by assessing validation set MSE and PEHE on the factual and synthetic data, respectively. Note that outside of proof-of-concept demonstrations, one should rigorously select an appropriate functional class among multiple benchmarks to avoid model misspecification [30].

As a benchmark treatment assignment policy, we rank all individuals by their clinical effect $\hat{HTE}_C$. This is the standard assignment policy in HTE literature. Specific to our setting is that we introduce an operational cost that, if exceeded, prohibits further treatment of individuals. Our synthetic setup reflects that treating each individual has a unique cost that depends on its features x. For the second policy, reflecting the joint clinical-operational perspective, we take into account the impact on the operational budget and treat based on the estimated clinical effect per unit of operational effect, or $\hat{HTE}_C/\hat{HTE}_O$.

4.4 Results

On the test set, taking into account the joint perspective, we treat 170% more patients and achieve a total clinical effect increase of 57.83%, compared to the clinical-only baseline, using the same operational resources. The results highlight the synergy between the process and clinical views. Even with maximisation of clinical effect in mind, it is thus helpful to adopt a joint perspective. Post-deployment, a model can be further evaluated by assessing whether following the model's recommendations has improved patient outcomes.

5 Discussion

The interplay between causal learning and process mining is a promising frontier for the management of healthcare processes. However, to empirically validate this promise, awareness of the practical requirements of HTE inference is required. Based on the conducted analyses, we enumerate three main lessons. Moreover, we reflect upon two broader perspectives on HTE inference for process mining.

Lesson 1: The HTE input table enables the use of state-of-the-art causal machine learning algorithms. The transformation of an event log to the HTE input table, the standard HTE modelling setup, can be performed with minimal overhead. Hence, state-of-the-art machine learning methods are available to the process mining community to develop causal models using event logs. We believe this simple formalization significantly lowers the threshold for coalescence between the HTE inference and process mining communities.

Lesson 2: To take more effective actions in healthcare, effects on both clinical and operational outcomes need to be modeled. Currently, many process mining works do not give explicit consideration to clinical process outcomes, while the machine learning for healthcare community does not model operational outcomes. In healthcare, the impact of an action on process efficiency is often not uniformly distributed across every case or intervention. We have intuitively illustrated how total clinical gain can be achieved when jointly modelling the clinical and operational effect of actions. However, to apply this in a real-life setting, we need the right combination of data and domain knowledge.

Lesson 3: A paradigm-shift for event log building is needed to fully capitalise on causal learning. To facilitate causal process interventions based on HTE inference, more a priori planning is required than is the current practice in the process mining field. First, the causal assumptions (Sect. 2.1) need to be validated together with domain experts. Second, these assumptions also translate to explicit data requirements. Currently, event logs mainly highlight when particular activities were executed on a patient. However, for HTE inference, confounders also need to be included, which will often require broader data extraction when building an event log. Finally, to enable jointly modelling operational and clinical outcomes, representative outcome measures need to be defined for the application at hand.

Perspective 1: Methodological extensions towards methods that can learn directly from event logs and capitalise on time dependencies are on the horizon. While the definition of the HTE input table offers a simple solution to enable causal learning using event logs, featurising the event log comes at the cost of losing information. For example, the time-series nature of the data is often lost when translating to a tabular data structure. Hence, methods that can learn from the original event logs offer opportunities to learn from richer data and the time dependencies in the log. Possible solutions could originate from time-series compatible models, such as RNN, LSTM or Transformer-based architectures.

Perspective 2: More discussion is needed to establish a consensus on policy standards and ethics. While opportunities arise due to novel technologies based on observational data, the adoption of decision support systems in healthcare needs to be soundly motivated. Adoption standards have not yet been established for learning HTEs from observational data of healthcare processes. Existing evaluation standards have mainly evolved from the machine learning field and not from a consensus of requirements from governing bodies (e.g., EMA, FDA) and healthcare organizations. Although stronger theoretical underpinnings can increase trust in HTE model predictions, uncertainty estimates offer an explicit measure of model confidence. Ultimately, the level of uncertainty also influences healthcare process decision making [15]. Finally, we refer to Eichler et al. [9] for a detailed discussion on the requirements of algorithmic decision-support in healthcare based on observational data.

6 Conclusion

In this paper, we introduce a joint approach to HTE inference, combining the clinical and operational perspective of healthcare processes. Despite its potential, careful consideration is required to incorporate HTE inference in the toolbox of healthcare organisations. When the prevailing assumptions are not accounted for when building the event logs, estimates of causal effects will not be identifiable and, hence, biased. Most importantly, strong cooperation with domain experts is needed to check for hidden confounders as violations of this assumption cannot be deduced from the data itself. To the best of our knowledge, no publicly available event logs have been collected with the HTE assumptions in mind, which hampers the development and testability of causal learning for process mining.

References

1. van der Aalst, W.M.P.: Process Mining - Data Science in Action. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
2. Alaa, A.M., van der Schaar, M.: Bayesian inference of individualized treatment effects using multi-task Gaussian processes. arXiv preprint arXiv:1704.02801 (2017)
3. Berrevoets, J., Alaa, A., Qian, Z., Jordon, J., Gimson, A.E., Van Der Schaar, M.: Learning queueing policies for organ transplantation allocation using interpretable counterfactual survival analysis. In: Proceedings of the International Conference on Machine Learning, pp. 792–802 (2021)
4. Berrevoets, J., Jordon, J., Bica, I., Gimson, A., van der Schaar, M.: OrganITE: optimal transplant donor organ offering using an individual treatment effect. In: Proceedings of the 2020 Annual Conference on Neural Information Processing Systems (2020)
5. Berrevoets, J., Verboven, S., Verbeke, W.: Optimising individual-treatment-effect using bandits. arXiv preprint arXiv:1910.07265 (2019)
6. Bica, I., Alaa, A.M., Lambert, C., Van Der Schaar, M.: From real-world patient data to individualized treatment effects using machine learning: current and future methods to address underlying challenges. Clin. Pharmacol. Theor. **109**(1), 87–100 (2021)
7. Bozorgi, Z.D., Teinemaa, I., Dumas, M., Rosa, M.L., Polyvyanyy, A.: Process mining meets causal machine learning: Discovering causal rules from event logs. In: van Dongen, B.F., Montali, M., Wynn, M.T. (eds.) Proceedings of the 2nd International Conference on Process Mining, pp. 129–136 (2020)
8. Curth, A., van der Schaar, M.: Doing great at estimating CATE? On the neglected assumptions in benchmark comparisons of treatment effect estimators. arXiv preprint arXiv:2107.13346 (2021)
9. Eichler, H.G., et al.: Are novel, nonrandomized analytic methods fit for decision making? the need for prospective, controlled, and transparent validation. Clin. Pharmacol. Ther. **107**(4), 773–779 (2020)
10. Gupta, N., Anand, K., Sureka, A.: Pariket: mining business process logs for root cause analysis of anomalous incidents. In: Chu, W., Kikuchi, S., Bhalla, S. (eds.) DNIS 2015. LNCS, vol. 8999, pp. 244–263. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16313-0_19

11. Hakim, A.A., et al.: Effects of walking on mortality among nonsmoking retired men. New Engl. J. Med. **338**(2), 94–99 (1998)
12. Holland, P.W.: Statistics and causal inference. J. Am. Stat. Assoc. **81**(396), 945–960 (1986)
13. Hompes, B.F.A., Maaradji, A., La Rosa, M., Dumas, M., Buijs, J.C.A.M., van der Aalst, W.M.P.: Discovering causal factors explaining business process performance variation. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 177–192. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_12
14. Johansson, F.D., Shalit, U., Kallus, N., Sontag, D.: Generalization bounds and representation learning for estimation of potential outcomes and causal effects. arXiv preprint arXiv:2001.07426 (2020)
15. Kochenderfer, M.J.: Decision Making Under Uncertainty: Theory and Application. MIT Press, Cambridge (2015)
16. Kurth, T., et al.: Results of multivariable logistic regression, propensity matching, propensity adjustment, and propensity-based weighting under conditions of nonuniform effect. Am. J. Epidemiol. **163**(3), 262–270 (2006)
17. Li, D., et al.: Assessment of treatment effect with multiple outcomes in 2 clinical trials of patients with Duchenne muscular dystrophy. JAMA Network Open **3**(2), e1921306 (2020)
18. Mannhardt, F.: Sepsis cases - event log, 4TU.ResearchData. dataset (2016). https://doi.org/10.4121/uuid:915d2bfb-7e84-49ad-a286-dc35f063a460
19. Mannhardt, F., Blinde, D.: Analyzing the trajectories of patients with sepsis using process mining. In: CEUR Workshop Proceedings, vol. 1859, pp. 72–80 (2017)
20. Martin, N., et al.: Recommendations for enhancing the usability and understandability of process mining in healthcare. Artif. Intell. Med. **109**, 101962 (2020)
21. Narendra, T., Agarwal, P., Gupta, M., Dechu, S.: Counterfactual reasoning for process optimization using structural causal models. In: Hildebrandt, T., van Dongen, B.F., Röglinger, M., Mendling, J. (eds.) BPM 2019. LNBIP, vol. 360, pp. 91–106. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26643-1_6
22. Prosperi, M., et al.: Causal inference and counterfactual prediction in machine learning for actionable healthcare. Nat. Mach. Intell. **2**(7), 369–375 (2020)
23. Qafari, M.S., van der Aalst, W.M.P.: Case level counterfactual reasoning in process mining. In: Nurcan, S., Korthaus, A. (eds.) CAiSE 2021. LNBIP, vol. 424, pp. 55–63. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-79108-7_7
24. Rojas, E., Munoz-Gama, J., Sepúlveda, M., Capurro, D.: Process mining in healthcare: a literature review. J. Biomed. Inform. **61**, 224–236 (2016)
25. Rubin, D.B.: Causal inference using potential outcomes: design, modeling, decisions. J. Am. Stat. Assoc. **100**(469), 322–331 (2005)
26. Shalit, U., Johansson, F.D., Sontag, D.: Estimating individual treatment effect: generalization bounds and algorithms. In: International Conference on Machine Learning, pp. 3076–3085 (2017)
27. Sugiyama, M., Krauledat, M., Müller, K.R.: Covariate shift adaptation by importance weighted cross validation. J. Mach. Learn. Res. **8**(5), 985–1005 (2007)
28. Suriadi, S., Ouyang, C., van der Aalst, W.M.P., ter Hofstede, A.H.M.: Root cause analysis with enriched process logs. In: La Rosa, M., Soffer, P. (eds.) BPM 2012. LNBIP, vol. 132, pp. 174–186. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36285-9_18
29. Tian, Y., Schuemie, M.J., Suchard, M.A.: Evaluating large-scale propensity score performance through real-world and synthetic data experiments. Int. J. Epidemiol. **47**(6), 2005–2014 (2018)

30. Vansteelandt, S., Bekaert, M., Claeskens, G.: On model selection and model misspecification in causal inference. Stat. Methods Med. Res. **21**(1), 7–30 (2012)
31. Vasilyev, E., Ferreira, D.R., Iijima, J.: Using inductive reasoning to find the cause of process delays. In: Proceedings of the 15th Conference on Business Informatics, pp. 242–249 (2013)
32. Verbeke, W., Olaya, D., Berrevoets, J., Verboven, S., Maldonado, S.: The foundations of cost-sensitive causal classification. arXiv preprint arXiv:2007.12582 (2020)

Interactive Process Mining Applied in a Cardiology Outpatient Department

Juan José Lull[1](✉), Adrián Cid-Menéndez[2], Gema Ibanez-Sanchez[1], Pedro Luis Sanchez[2], Jose Luis Bayo-Monton[1,2,3], Vicente Traver[1], and Carlos Fernandez-Llatas[1,3]

[1] Instituto Universitario de Tecnologías de la Información y de las Comunicaciones (ITACA), Universitat Politècnica de València, Valencia, Spain
{jualulno,geibsan,jobamon,vtraver,cfllatas}@itaca.upv.es

[2] Cardiology Department, University Hospital of Salamanca, Instituto de Investigación Biomédica de Salamanca (IBSAL), Salamanca, Spain

[3] Department of Clinical Science, Intervention and Technology (CLINTEC), Karolinska Institutet, Stockholm, Sweden
carlos.fernandezllatas@ki.se

Abstract. Cardiology departments receive many outpatients from primary care services and it is necessary to differentiate which patients need special attention. One-stop clinics were deployed in a hospital in Salamanca (Spain) to triage such patients, separating those who needed further examination and those who were discharged.

Data (covering December 2018—August 2020) was explored and there was an iterative process in which clinicians, process miners and technical staff at the hospital interacted in special interviews or Data Rodeos. Interactive Process Indicators (IPIs) were generated. During Data Rodeos data quality problems arose and were tackled, input data was cleaned and preconditioned, process activities were discovered and modelled.

The original assumption that the iterative implementation of the IPI would allow clinicians and managers to have a deeper understanding of the one-stop cardiology clinics process, was evaluated and validated by them. After each iteration, they found that the IPI was more useful and near to the reality they see everyday.

The final IPI was easy to interpret by the clinicians. In the end, many key indicators were extracted, but most importantly, clinicians had a comprehensive tool that they could use by themselves, without technical assistance, to extract and interpret different indicators at any time, providing a high-quality source of information to improve patient-centered daily medical care.

Keywords: Process mining · Cardiology · Interactive Process Mining · Healthcare system · Outpatient care

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 340–351, 2022.
https://doi.org/10.1007/978-3-030-98581-3_25

1 Introduction

Hospitals have limited resources and they need to constantly evolve to attend patients in time and differentiate between patients that need extra attention and diagnosed patients and healthy subjects. Patients that need a heart diagnosis are usually referred from primary care to the cardiology service. In the normal pathway, when the patient arrives to the service, the doctor has a consultation and may require further tests in e.g., a month. Afterwards, the patient goes back to the cardiologist some weeks later and can be diagnosed as healthy and thus discharged. They can also be diagnosed with a specific known condition, and a treatment starts. Finally, the patient may also need extra tests because of a more complex malady. This process takes a long time and the number of consultations could be reduced with other approaches.

The model of one-stop clinic reduces the number of patients referred from primary care through a pathway that aims to provide a higher efficiency in the diagnosis of patients. Contrary to the normal pathway, the one-stop clinic aims to create more thorough consultations, in which the doctor has support from nursing staff and full access to specific extra test requests within the cardiology department. Many times, a basic echocardiogram (ECHO) or electrocardiogram is performed during the consultation and most patients are discharged, since they can be directly diagnosed and do not need any more tests. If extra tests or interventions need to be performed, they are carried out on a different (near) scheduled date. Tests generate a routine follow-up review. The way one-stop clinics are conducted makes it possible to quickly rule out most cardiac syndromes and, fundamentally, to discharge a great proportion of referred patients who do not meet the criteria for specific follow-up to primary care, saving resources for the sicker cohort of patients that benefit from a closer surveillance [2], and reducing the length of stay. One-stop clinics have been extended to European cardiology services in recent years as a solution for the management of first visits [7].

In the context of a European funded project, a cardiology department that had implemented one-stop clinics wanted to visualize and take decisions based on the processes related to these one-stop clinics, with the help of Process Mining (PM). The goal in the PM methodology is to provide solutions to the experts (in this case, the clinicians), that help them understand the behavior of the processes [11]. In the healthcare domain, process indicators need to be extracted from the data by analysts with the other stakeholders' help: Managers, clinicians, and Information Technology (IT) professionals.

PM has been applied to different Cardiology use cases. In [8], it was applied in a Pakistan Cardiology Hospital. It did not count on real event logs but was rather based on reports from the physicians. The aim of the research was to prove that PM could be applied to enhance the medical system in the country. In [1], Interactive Process Mining (IPM) [4] was applied to investigate how the time it takes to transfer the patient with myocardial infarction from their home to the percutaneous intervention center affects the survival rate. There is also a literature review about cardiovascular diseases studied with PM [9]. In that publication, they focused on what specific disease each paper had studied and which PM method had been used. A deep inspection of the process and the way it was achieved was not apparent in the referred works.

Key performance indicators (KPIs) are commonly used in clinical settings. IPM, however, lets the stakeholders define Interactive Process Indicators (IPI) from the data and the questions that the clinicians and managers have. KPIs are single-dimension variables, e.g., the ratio of patients discharged per hospitalized patients, per week. Differently, IPIs contain a full visual description of the process, and also contain metadata about process traces and events. KPIs are just numbers, so errors in the data will not be easily caught, while IPIs show the process and errors are easily detected. Timing is included in the IPI and can be visualized, used for differentiating groups, etc. The reader is encouraged to consult the section Interactive Process Indicators by Example in [5]. IPM is a methodology that puts the healthcare professionals first, facilitating the understanding and easiness of exploration of the process indicators.

The structure of the rest of the paper is as follows: The application of IPM in the cardiology service at Salamanca Hospital, Spain, is explained in the Materials and Methods section. Afterwards, the results of applying IPM are shown in Results. Finally, next steps, limitations, and a comment on the COVID pandemics are shown in the Conclusion and Discussion section.

2 Materials and Methods

2.1 Data of Origin

For the study, EHR data collected from the Hospital Information Service (HIS) in the Cardiology Department was used.

Clinical data had been manually introduced in MediConnect®(Fleischhacker, Schwerte, Germany), a clinical process management software tool. The timespan for the data analysis was from December 2018 until August 2020. The records that were included are defined in Table 1.

Table 1. Records in the initial data provided.

Anonymized patient ID	Patient identifier
Mediconnect activity ID	Activity identifier
Activity Name	i.e. Nuclear medicine (NM) test, Magnetic Resonance (MR) test, Holter test, Computerized Tomography (CT) test, Structural intervention, Outpatient visit, etc.
Agenda	Sub-type of activity (e.g. Kind of outpatient visit: One-stop clinic, general hospital consultation).
Activity code	It identified the kind of test in a more specific way, e.g., for the ECHO test, the code referring to trans-thoracic test.

(*continued*)

Table 1. (*continued*)

Order Status	The action could be in one of the following statuses: Planned, Delayed, Running, Confirmed, Finished, Canceled
Order	Date and time when the clinician, nurse or administrative asked for a new activity.
Order scheduled Start	Date and time when the appointment was scheduled to start (the real start time was not available).
Order scheduled Finish	Date and time when the appointment was scheduled to finish.
Follow up request	It could be one of: – Rehabilitation – NM test – Holter – Hemodynamic test – Structural intervention – Spirometry – ECHO – Consultation – Implant – MR – CT
Reason or symptom(s) for the request	Logical values for one or more fields among the following: Asymptomatic, Dyspnea, Dizziness, Palpitations, Murmur, Syncope.
Diagnostic from the activity	Logical values for the following fields: Cardiac arrhythmia, Congenital cardiopathy, Ischemia, Valvular heart disease, Infectious Endocarditis, Aorta illness, Pericardiac illness, Structural damage, Lung hypertension, Heart failure, Myocardiopathy, Sudden death, Pulmonary embolism, Syncope.
Patient plan	The possible values were: Return to primary care, interconsultation, monographic consultation, request for tests, request for intervention, request for tests and results, etc.

Extra fields were also available but were already discarded at an initial data quality assessment stage, such as the logical field Patient discharge (in that case, Patient plan: return to primary care was more accurate).

2.2 Variables of Interest for the Clinician

The clinicians had in mind some aspects of the process that they wanted to measure and dive into. One aspect was the wait lists: They wanted to detect where and when bottlenecks occurred, along with time from primary care request for the cardiology department till the patient was attended. Another need was to discover long time-to-diagnosis and long time-to-treatment of patients with a cardiac disease.

Clinicians also needed to detect low level of coordination with the general practitioner for derivation and follow-up of patients. Another question was whether there were differences in clinical decision making between junior and senior doctors, specially through the number of requested tests. Gender and age inequities in diagnosis and requested tests were of importance, along with the impact of Covid-19 pandemic in the number of consultations over the worst months of the first wave.

2.3 Methods in Data Rodeos

Data rodeos are sessions with all the actors involved (managers, PM analysts, clinicians, technical staff), where an interactive analysis of the data and latest process indicators is performed. Each derived IPI helps better understand, quantify, and qualify the process that is being studied. Duration of data rodeos can range from hours up to one month. The result of each data rodeo, the IPI, must be validated by healthcare professionals. The process indicator should allow the clinician to check the representation against the HIS. This leads to an increase of confidence in the model by the professional, e.g., if privacy allows it, the doctor can see any patient identification and see that the patient follows the process as observable in the IPI.

Data rodeos are separated into three stages: Shake down, Research and Production [3]. The initial stage requires aggressive data cleaning and fast PM discovery algorithms: It corresponds to the initial interviews between all the parties and the iterations in the process model must keep doctors attentive. The second stage, Research, can introduce long processing research tools and must introduce a more respectful data cleaning strategy: In the medical domain, *outliers* may be related to patients that follow a different path that is especially important to the doctors. The production stage is carried out after the research stage and the identified IPI is deployed in a live environment and it incorporates the creation of a dashboard for the hospital, load tests and security and privacy implementations. In this study, Shake down and Research stages are presented.

Anonymized data files were created by the IT staff at the hospital containing the information that was described in the Subsect. 2.1. For each IPI, the data files were ingested, generating a PM log. The log was filtered and processed. Afterwards PM discovery was applied, and a model was discovered. The model was processed and, after enhancement and conformance, the IPI was generated. With each generated IPI, a report with invalid traces and other data quality problems was created. This let all the actors find any problem in the data or the IPI. Apart from the soundness of the IPI, it must also be compelling and easy-to-understand and interact with, by the health professional [10]. Different ways of achieving an augmented model with metadata have been described, creating maps where color, node or transition representation size, tags, transparency, etc. change to show information about the process. During this study color was used to represent duration in each activity (represented in nodes) and number of traces (coded in the transitions), both with gradients with value ranges represented in the legend in each figure.

3 Results

3.1 First Data Rodeo

During the first Data Rodeo, an initial IPI was generated, as a base to work on. The main nodes were: 1. Request for One-stop clinic 2. First consultation 3. Consultation (any further consultations, after the first one). 4. ECHO: an echocardiography had been performed for the patient. 5. Discharge 6. Exitus (deceased)

Since the time spent in each node was unknown, this could not be introduced in the IPI, but in the case of the time from request to scheduled consultation.

Figure 1 shows a representation of the IPI where node color implies time spent at the node and transition colors show the number of executions that go from node to node (see Heatmap legend).

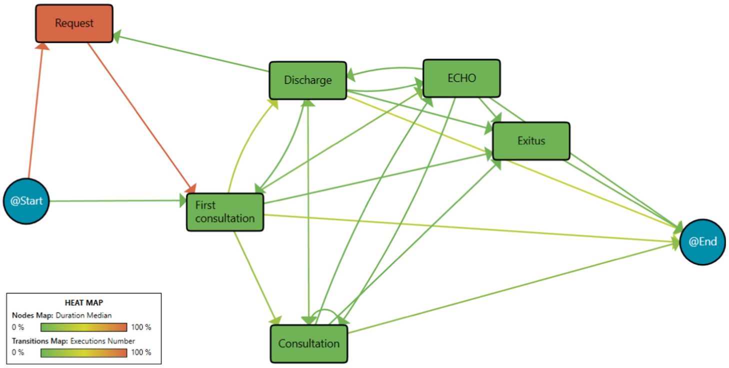


Fig. 1. Initial IPI from First Data Rodeo.

As mentioned earlier, the process at this stage needed further work, e.g. there are transitions that start at First Consultation and that cannot happen in reality. However, those details would be polished later.

In the IPI, most of the information remained in the model but was not visible at first glance, as in Fig. 1. The following extra data was introduced with filters: symptoms and diagnostics for each patient.

This data rodeo let us find that, as observable in Fig. 1, there is a high number of patients who do not need extra tests and are directly discharged. This was observed in the transitions First consultation → @End, and First consultation → Discharge, and it was an approximate measure of efficiency with around 50% patients discharged after the initial consultation.

Doctors were very interested in watching wait times and other variables depending on the symptoms. However, they were discarded since most consultation did not have the information. A second data rodeo was appointed to further explore the data and obtain a better IPI.

3.2 Second Data Rodeo

The clinicians wanted to measure the ECHOs that had been solicited to be performed during the consultation, since the number of ECHOs impacted the efficiency of the clinic. A node was created for the case of non-request of an ECHO during the consultation. A view of the process can be seen in Fig. 2.

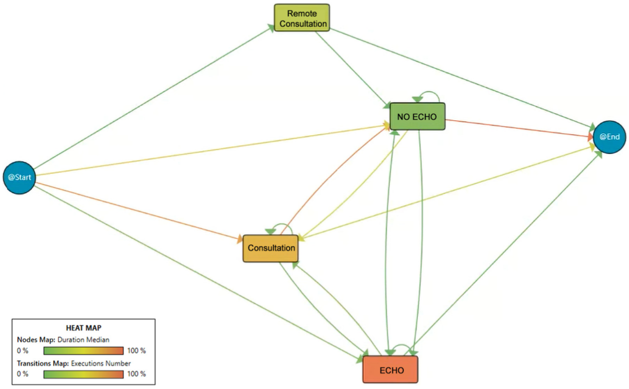


Fig. 2. Second IPI with Consultation and two nodes for the extra ECHO test and the absence of it.

Through the creation of trace metadata groups for doctor category (registrars, consultants), the proportion of the during-consultation ECHOs could be easily seen.

The data about clinicians who had attended the patients in the one-stop had been introduced, since one question was if registrars (junior clinicians) asked for more tests than consultant doctors (seniors). The IT team provided the data, with the following fields, among others: Mediconnect activity ID, Clinician name. The rest of the fields were discarded since they were not finally used to create the IPI. With this field, the clinicians involved in the generation of the IPI classified doctors between consultants and registrars. This information was included in each trace as Type of doctor.

The clinicians involved could see the percentage of patients that underwent an ECHO inside the consultation. However, it was estimated later that a relevant percentage of doctors did not fulfill the ECHO forms on the EHRs due to complexity and time allocation per patient in the clinics, which led to a lack of information in this regard. Thus, intra-consultation ECHOs had to be removed from the process.

3.3 Third Data Rodeo

Extra-consultation ECHO tests could be checked, along with other tests and interventions. Thus, if any test was requested, it was introduced in the IPI as trace metadata. Age and gender were also included, as well as the type of doctor. This would let clinicians find out e.g., if registrars asked for more ECHO tests than consultants. Also, if there were more men than women with heart problems, etc.

It was also decided that, since the interest was in the first consultation for a referred patient from primary care, when there was more than one consultation, only the first consultation would be considered, along with the extra tests and the discharge. Further events in time would be discarded for the process model.

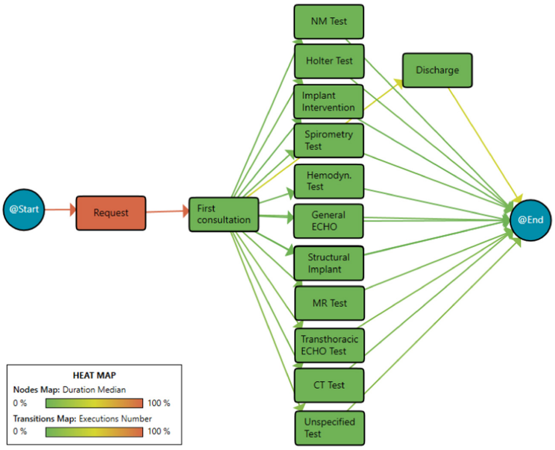


Fig. 3. Final IPI view.

Figure 3 shows a view of the final IPI. It included the following trace metadata: – Month of consultation – Year of consultation – Time to next consultation, in days (−1 if data was not available) – Gender – Age – Test/intervention required – Diagnostics (array with the different diagnostics) – Type of doctor.

The IPI was created with 15 nodes, as depicted in Fig. 3. The transition probability from consultation to discharge was high, 47%.

The number of traces was extracted for each month and year in the available data. This is shown in the column chart in Fig. 4. It can be observed that 2020 (January to August, amidst the pandemic) had a lower number of clinics than 2019.

Transition probability in 2019 from first consultation to discharge was 50% while it was reduced to 17% in January to August in 2020. This may be because of patients not going to the clinic due to the lockdown in the first pandemic wave. This would increase the percentage of patients that would need an extra test or intervention.

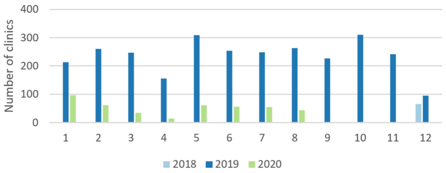
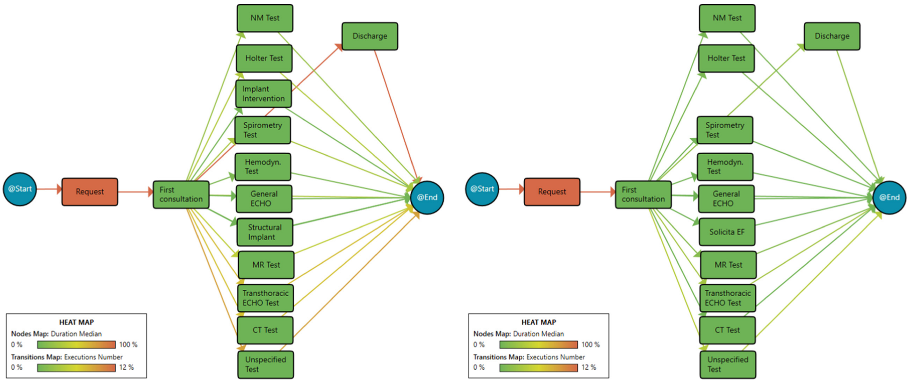


Fig. 4. Number of traces per month, equivalent to number of new patient clinics.

The percentage of unspecified tests/intervention requests went up from 10% in 2019 to 31% in 2020. Also, the most demanded test in 2020 was trans-thoracic ECHO (23%) compared to an 8% in 2019.

All these data were easy to interpret at one glance watching the IPI view (Fig. 5) by looking at the color-coded transition probabilities.



(a) 2019 IPI visualization

(b) 2020 IPI visualization

Fig. 5. IPI view with same color gradient for 2019 and 2020.

It could also be seen that age groups had different number of clinics depending on the month. In Fig. 6, August and December 2019 are compared in local percentage (distribution of 100% between the age groups). Elderly people were more treated in December than in August, and adults were the ones that coped with that relative decrease in the same months.

Registrar-requested external ECHOs against those requested by consultants were compared. Although more clinics were performed by consultants (68.9%) than by registrars (37.1%), more than half of the ECHO tests were requested by registrars (specifically, 57,1%).

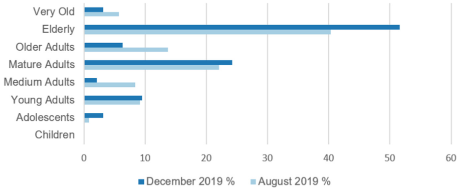


Fig. 6. Age group clinics, in relative percentage, depending on the month.

4 Conclusion and Discussion

After the initial IPM Shakedown, clinicians were provided with a comprehensive and easy-to-use tool that allowed them to answer most clinical questions proposed in the first meeting.

During the last years national and European cardiology societies have stablished metrics and benchmarks that every cardiology department should meet in the outpatient setting [6]. One of the key markers is average wait for a first consultation since the time a new referral is done. With IPM, clinicians observed the median waiting time was 19 days and 20 h, showing a clear improvement point in comparison with national standards. Overall view of gender and age distribution on the patient cohort was obtained, providing a better understanding of population that access a cardiology department and correlating with general population aging. These data were extracted by the clinicians by inspection into the IPI.

Through IPM analysis clinicians could define and classify the outcome of the clinic in big generic cardiac syndromes or the absence of a specific diagnosis in patients with a structural normal heart. The discharge rate from the clinic was 47% reaching the acceptable benchmark set by expert consensus mentioned before. However, improving communication with other specialists and primary care and the implementation of novel alternatives such us e-consultation could be an option to reduce even more unnecessary referrals.

Reducing the number of unnecessary requested tests is of key importance for a public funded healthcare system. Prior to rationalisation of diagnostic and interventions, it is fundamental to know the exact volume and statistics of requests generated by the one-stop clinic. This task was successfully achieved with the analysis of the process. The IPI is effectively an audit of the outpatient service that will promote the update of clinical protocols, and refreshment educational sessions, reducing unnecessary and costly tests, benefiting both patient safety and the heath system economy.

Time allocated per patient for a one-stop clinic is usually enough for a general cardiologist to perform an external ECHO if deemed necessary after formal clinical interview and physical examination. The existence of too many early requests of extra ECHOs is perceived as a failure of the main goal of this kind of clinics. The indicator of 10% of ECHOs directly requested from one-stop clinic shows an improvement opportunity.

The SARS-Cov2 pandemic had a major impact in the healthcare system during 2020, lock-downs and resource reallocation to an over-saturated inpatient care dropped the number of first consultations [12]. Currently, keeping up with missed appointments is a struggle in outpatients services. IPM analysis has helped the cardiology department to quantify in an accurate way the damage made to the outpatient service during the worst months of Covid spread.

In conclusion, clinicians were provided with a useful tool for data analysis. The results through IPM were used as a complete audit of outpatient service deriving into clinical protocol changes and exposing improvement opportunities. Developments in the IPI are still to come (such as introducing the distance between the patient and the hospital, and other data that will help clinicians with new questions), but it is mature enough to answer the initial queries.

Acknowledgment. This activity has received funding from EIT Health (www.eithealth.eu) ID 20328, the innovation community on Health of the European Institute of Innovation and Technology (EIT, eit.europa.eu), a body of the EU, under Horizon 2020, the EU FP for Research & Innovation. https://www.eithealth.eu.

References

1. Borges-Rosa, J., Oliveira-Santos, M., Simoes, M., et al.: Process mining tools: where should we build another PCI centre to reduce STEMI mortality? Eur. Heart J. **41**(Supplement 2) (2020)
2. Falces, C., Sadurní, J., Monell, J., et al.: One-stop outpatient cardiology clinics: 10 years' experience. Rev. Esp. Cardiol. **61**(5), 530–533 (2008)
3. Fernandez-Llatas, C.: Bringing interactive process mining to health professionals: interactive data rodeos. In: Fernandez-Llatas, C. (ed.) Interactive Process Mining in Healthcare. HI, pp. 119–140. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-53993-1_8
4. Fernandez-Llatas, C.: Interactive Process Mining in Healthcare. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-53993-1
5. Fernandez-Llatas, C.: Interactive process mining in practice: interactive process indicators. In: Fernandez-Llatas, C. (ed.) Interactive Process Mining in Healthcare. HI, pp. 141–162. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-53993-1_9
6. González-Juanatey, J.R., Virgós Lamela, A., García-Acuña, J.M., Pais Iglesias, B.: Clinical management in cardiology. Measurement as a means to improvement. Rev. Esp. Cardiol. **74**(1), 8–14 (2021)
7. González-Llopis, F., Palazón-Bru, A., et al.: Clinical effectiveness of a cardiology outpatient management plan to reduce inefficiency in consultations. Postgrad. Med. **133**(2), 166–172 (2021)
8. Hakim, A.: Improving healthcare systems through process mining. In: 2020 IEEE 23rd International Multitopic Conference (INMIC), pp. 1–4 (2020)
9. Kusuma, G., Hall, M., Johnson, O.: Process mining in cardiology: a literature review. IJBBB **8**, 226–236 (2018)

10. Mendling, J., Djurica, D., Malinova, M.: Cognitive effectiveness of representations for process mining. In: Polyvyanyy, A., Wynn, M.T., Van Looy, A., Reichert, M. (eds.) BPM 2021. LNCS, vol. 12875, pp. 17–22. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-85469-0_2
11. Van der Aalst, W.: Process Mining: Data Science in Action. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
12. Wosik, J., Clowse, M.E.B., Overton, R., et al.: Impact of the COVID-19 pandemic on patterns of outpatient cardiovascular care. Am. Heart J. **231**, 1–5 (2021)

Discovering Care Pathways for Multi-morbid Patients Using Event Graphs

Milad Naeimaei Aali[1](✉), Felix Mannhardt[2], and Pieter Jelle Toussaint[1]

[1] Norwegian University of Science and Technology, Trondheim, Norway
milad.naeimaei@ntnu.no

[2] Eindhoven University of Technology, Eindhoven, The Netherlands

Abstract. Patients suffering from multiple diseases (multi-morbid patients) often have complex clinical pathways. They are diagnosed and treated by different specialties and undergo other clinical actions related to various diagnoses. Coordination of care for these patients is often challenging, and it would be of great benefit to get better insight into how the clinical pathways develop in reality. Discovering these pathways using traditional process mining techniques and standard event logs may be difficult because the patient is involved in several highly independent clinical processes. Our objective is to explore the potential of analyzing these pathways using an event log representation reflecting the independent clinical processes. Our main research question is: How can we identify valuable insights by using a multi-entity event data representation for clinical pathways of multi-morbid patients? Our method was built on the idea to represent multiple entities in event logs as event graphs. The MIMIC-III dataset was used to evaluate the feasibility of this approach. Several clinical entities were identified and then mapped into an event graph. Finally, multi-entity directly follows graphs were discovered by querying the event graph visualizing them. Our result shows that paths involving multiple entities include traditional process mining concepts not for one clinical process but all involved processes. In addition, the relationship between activities of different clinical processes, which was not recognizable in traditional models, is visible in the event graph representation.

Keywords: Health care · Multi morbidity · Multi-entity Process Mining

1 Introduction

Based on the UN annual report, the number of older people is envisaged to be nearly 2.1 billion by 2050, growing to a size more than twice as large as in 2017 [1]. As a result of the aging population, it is expected that "multi-morbidity" is going to increase [2]. Multi-morbidity refers to any co-occurrence of conditions in the

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 352–364, 2022.
https://doi.org/10.1007/978-3-030-98581-3_26

Table 1. Example of event log with single-dimensional or single-entity event data.

Case Identifier	Event	Timestamps	Property-X	Property-Y
1	a	2013-10-29T05:00:00	X1	Y1
1	b	2013-10-30T06:00:00	X2	Y2
1	c	2013-10-31T07:00:00	X3	Y3
2	a	2013-10-01T08:00:00	X4	Y4
2	c	2013-10-19T09:00:00	X5	Y5
3	a	2013-10-29T06:00:00	X6	Y6

same person [2]. Sometimes the term "co-morbidity" is used instead of multimorbidity, while the term co-morbidity is defined as the combination of extra disorders besides an index disease [2]. The treatment of multi-morbid patients is a complicated task since they generate several challenges. These include recognizing signs and symptoms of different illnesses, managing multiple medications and treatments, interacting between various health conditions, and allocating resources by medical centers. These lead us to develop care pathways for patients with multi-morbidity in a way that overcomes these challenges.

Care pathways, as one of the central tools used in healthcare, can be described as a straightforward statement of the aims, a representation of the interactions between the health's resources and patients, or a description of roles, sequential decisions, and activities related to the care process [3]. The primary goal of care pathways is reducing variability in the treatment of diseases [4]. Since care pathways are a set of time-framed events focusing on a specific situation that delivers guidance about how to deal with conditions that appear during treatment's processes [4]. It can be itself considered as a process which is a sequence of events with a common goal [5].

Processes can be graphically represented by process models [5] which explain responsibilities, inspect compliance, predict performance using simulation [6], manage complexity, reduce variation, and enhance coordination [5] in processes. Discovering process models or process discovery from event logs is one of the main tasks in process mining. Event logs contain sequences of events recorded from information systems. Any registered event refers to at least (1) an activity (i.e., a well-defined step in the process), (2) a case or process instance representing a single entity, (3) a unique timestamp. Logs fulfilling these requirements are called single-dimensional or single-entity event data [7], which an example of this type of log was shown in Table 1. Single entity event data also can refer to properties (e.g., the person executing or initiating the activity) [5].

If we want to satisfy all practitioners in the healthcare sector and achieve a holistic process view for care pathways [8], we should consider more than one clinical process of patients' care pathways. But, the standard type of event data forces us to deploy an event log for each clinical process of patient care pathways. On the other hand, if we have multi-entity event data, meaning events refer to multiple entities (e.g., each clinical process of a care pathway), relational databases and traditional process mining techniques are ineffective.

Table 2. Excerpt of a event log with multi-entity event data relating to multiple entities that can be converted into an event graph representation [7].

Event	Timestamps	EntityTypeA	EntityTypeB	EntityTypeC	PropertyX	PropertyY
a	2013-10-29T05:00:00	1	Origin 1	Origin 4	X1	Y1
b	2013-10-30T07:00:00	1		Origin 4	X4	Y4
c	2013-10-31T07:00:00	1		Origin 5	X5	Y5
f	2013-10-31T09:00:00	1	Origin 1	Origin 4	X7	Y7
a	2013-10-01T08:00:00	2	Origin 2	Origin 4	X2	Y2
b	2013-10-30T06:00:00	2		Origin 4	X3	Y3
c	2013-10-31T07:00:00	2		Origin 5	X5	Y5
f	2013-10-31T09:00:00	2	Origin 2	Origin 4	X7	Y7
a	2013-10-29T05:00:00	3	Origin 1	Origin 4	X1	Y1
b	2013-10-30T06:00:00	3		Origin 4	X3	Y3
c	2013-10-19T09:00:00	3		Origin 5	X6	Y6

This study explores the potential of analyzing care pathways for patients with multi-morbidity using a multi-entity event data representation reflecting the independent clinical processes. Our main research question is: *How can we identify valuable insights by using a multi-entity event data representation for care pathways of multi-morbid patients?* The remainder of this research is structured as follows. Section 2 reviews state-of-the-art research about the use of multi-entity event data in process mining and how to represent and store them. Section 3 introduces MIMIC-III that is used to illustrate and validate our approach. In Sect. 4, we show how to build multi-entity event data for multi-morbid patients. In Sect. 5, we show preliminary results that are, then, discussed in Sect. 6. We conclude with an outlook on future work in Sect. 7.

2 Related Work

Multi-entity event data can not be stored in the same way as single-entity event data; furthermore, in this setting process discovery is not possible with traditional methods. In this section, we explore the related literature from several perspectives to select a good format for multi-morbid care pathways event data.

2.1 Multi-entity Event Data

In the approach of [9], known as object-centric process mining, each case notion is referred to as one object type (e.g., application and vacancy can be two case notions or two object types, and each of them has its own case identifiers). In that approach, events can refer to multiple case notions instead of referring to a single case notion. A process model is first discovered for all objects sequentially. Then, each directly-follows relation is labeled to its related object type. For example, if event-1 that is related to object-1 happened right before event-2 that is related to object-1, event-2 directly follows event-1, and so on.

Another type of multi-entity event data was proposed by [7]. Based on [7], there does not need to be a single case notion, but events are related to one or more *entities* of different entity types. Entities themselves can also be related to each other. The required input events have been shown in [7] is similar to the one shown in Table 2. Information about the relations may also be extracted from other sources, e.g., relational database keys. Process models can be discovered in a flexible manner per entity or for various combinations of several entities. Our event log format for storing multi-entity event data is based on this model.

2.2 Storing Event Data

A classical approach for storing event data is using relational databases (RDBs). A relatively new approach is using an event graph which is a mathematical graph data structure that is built by converting relational database concepts to vertices, edges, nodes, and relationships [7]. This leads to a natural representation of multi-entity event data and the possibility to discover multi-entity models by querying from event graphs.

A series of experiments were conducted in [10] to compare the performance and efficacy of relational databases and event graphs, sho1higher capabilities of event graphs. Extracting multi-entity event data needs to flatten event data because only a single case notion can be chosen [7] leading to traditional process mining. Additionally, a graph database can store all of the case notions of a multi-entity directly follows graph in only one graph [7].

Recently, event graphs were deployed for storing data. The work in [11] introduces an approach to store and retrieve single-entity event logs into/from graph databases. That approach defines how log files shall be stored in a graph database, and it also illustrates how directly follows graphs (DFG) can be calculated in the graph database. In another recent literature, task executions and routines in processes were classified and detected using event graphs [12]. In that research, at first, the event log was transformed into an event graph. Then graph theory was used to detect task execution patterns and their changes over time.

Converting multi-entity event data to an event graph was formalized in [7] by conceptualizing event log, events, entities, and classes. Based on [7] each event log has several events, and each event in one hand correlates to entities, and on the other hand, can be observed by classes. Meanwhile, the events can be related to each other if they directly follow each other. Entities can be related to each other based on the occurrence of their events. As well, the classes can follow each other by directly following relationships. Based on these reasons, in sum, an event graph seems to be a better approach compared to the relational database for storing multi-entity event data.

Vogelsang et al. [13] looks at process mining from multiple dimensions. Still, these dimensions are related to properties of cases such as region, age of patients, and not event data. In the approach, several single-entity event data, separated based on the difference between regions, ages, and so forth, were used.

Overall, we found that the subject of using event graphs in a healthcare setting and, in particular, discovering care pathways from multi-entity event data using event graphs was not yet explored in previous literature.

Table 3. List of patient ICD code and its repetitive in patients.

Diagnoses based on ICD codes	Patients frequency	List of Patients (Admission IDs)
7746	232	A1 - A2 - ... - A232
7661	163	B1 - B2 - ... - B163
7706	142	C1 - C2 - ... - C142
76519 - 76528	99	D1 - D2 - ... - D99
76518 - 76528	68	E1 - E2 - ... - E68
77089	63	F1 - F2 - ... - F63
...	...	...

3 Multi-entity Event Data in MIMIC-III

For evaluation of the feasibility of using event graphs for clinical pathways of multi-morbid patients, the MIMIC-III [14] is used. MIMIC-III is a freely accessible tertiary care database that involves information relating to patients admitted to critical care units (CCU) of Beth Israel Deaconess Medical Center in Boston, Massachusetts, during 2001 and 2012. Data from MIMIC-III were downloaded from several sources such as critical care monitoring information systems, bedside monitors, hospital and laboratory electronic health record databases, and social security administration.

The ninth revision of the international statistical classification of diseases and related health problems (ICD-9) is widely used diagnostic coding system. Each ICD-9 code corresponds to a single diagnostic disease except the codes starting with E and V, which are related to external causes of injury and additional classification. We use the ICD-9 code system for specifying multi-morbid patients by considering patients with several ICD-9 codes as patients with multi-morbidity.

We use a subset of data from MIMIC-III. To extract event data from MIMIC-III, first, from `DIAGNOSES_ICD` Table, values of *icd9_code* column, excluding codes start with E and V, were grouped by each distinct patient's hospital admission identifier (*hadm_id*). The `DIAGNOSES_ICD` table involves patients identifiers (*subject_id*), patients hospital admission identifiers (*hadm_id*), the sequence order in which the ICD-9 diagnoses were made (*seq_id*), and ICD-9 (*icd9_code*). After that, the patient admission identifier was grouped by an collection of ICD codes as shown in Table 3. Each row of Table 3, shows the number of observances of a disease (or group of diseases), which has been coded by ICD-9 format, at the time of admission of patients to the hospital. If the first row of the table shows more than one disease, we consider them as multi-morbidity cases. Meanwhile, a patient can have several admission identifiers that show the patients admitted to the hospital several times at different times.

From this initial look at a subset of the MIMIC-II dataset on multi-morbid patients, multiple entities can be identified, e.g., admissions, diseases (ICD codes), and so on. We now describe the relevant entities in detail and extract them to build an event graph representation.

4 Event Graphs for Multi-morbid Patients Pathways

This study explores how to analyze multi-entity event data for patients with multi-morbidity based on an event graph. Based on our research question, a hypothesis for this research was formulated as follows: *Applying event graph produces valuable insights when using multi-entity event data for clinical pathways of multi-morbid patients.* Our strategy is to design an experiment for the research to investigate this. This section describes the method we followed to investigate this question and build event graphs to discover care pathways for multi-morbid patients.

4.1 Identifying and Extracting Entities

Each distinct clinical process related to patients with multi-morbidity is called an entity. Since several clinical processes are involved in treating multi-morbid patients, entities can easily be identified by considering those clinical processes. We identified the following entities in the subset of the MIMIC-III dataset:

1. **Logistic.** This entity events contains admission, discharging, registering to Emergency department (ED), discharging from ED, In-hospital death (if died), calling-out request (when patients ready to discharge), and transferring between different services, care unit and wards. Six MIMIC-III tables were used to download this entity's events: `PATIENTS`, `ADMISSIONS`, `CALLOUT`, `SERVICES`, `ICUSTAYS`, `TRANSFERS`.
2. **Laboratory_Measurement.** This entity contains events of the type abnormal laboratory measurements, Which play an essential role in diagnosing and treating patients' diseases. For extracting these events *label*, *value*, *valueuom*, and *flag* columns of `D_LABITEMS`, and `LABEVENTS` tables were used.
3. **Prescriptions.** This entity contains starting and ending timestamps of medication-related order entries, i.e., prescriptions such as the drug which is prescribed to the patient, its dose's value, form, and unit of medication, for extracting of this entity `PRESCRIPTIONS` table was used.
4. **Diagnosis.** This entity was related to the first event at the beginning of each time of patients admissions. It involves a group of ICD codes showing patients' diseases in each admission. `DIAGNOSES_ICD` table relationship with other tables was used for downloading ICD codes of this entity.
5. **Admission.** In the end, the hospital admission identifier was appended to multi-entity event data. If an event is related to the NULL admission number, it is associated with the outpatient clinic.

Table 4 shows an example of created multi-entity event data for patients identified 4900. It is possible to extract multi-entity event data for each row of Table 4, while we consider the admission identifier or its equivalent patient identifier as a case identifier.

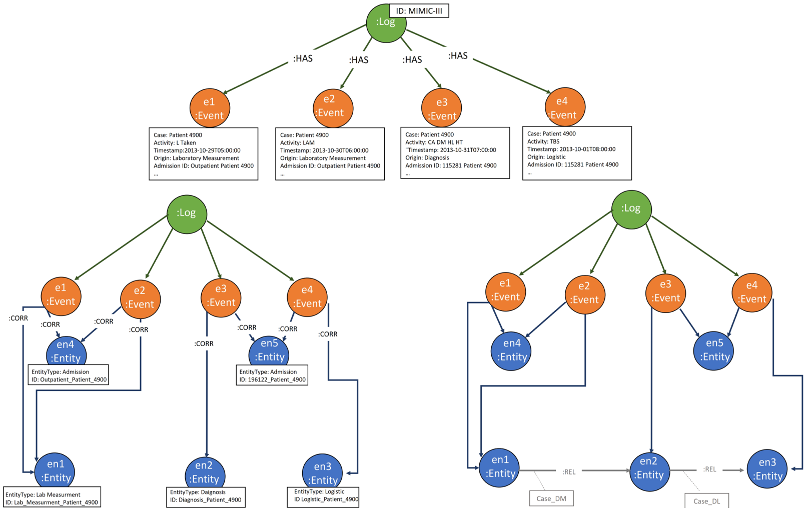


Fig. 1. Graph creation for Patient_4900: Steps ① (top), ② (bottom left), and ③ (bottom right)

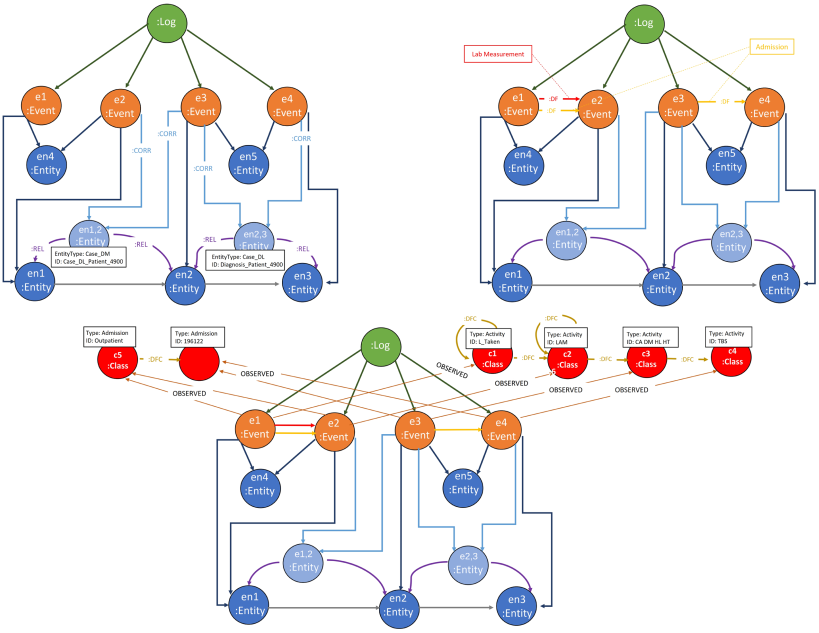


Fig. 2. Graph creation for Patient_4900: Steps ④ (top left), ⑤ (top right), ⑥ (bottom)

Table 4. Excerpt of an event log extracted from MIMIC-III with multiple entities. We abbreviate event labels in the remainder as follows: L_Taken = Laboratory Test Taken, LAM = Laboratory Abnormal Measurement, CA = Coronary Atherosclerosis, DM = Diabetes Mellitus, HL = Hypercholesterolemia, HT = Hypertension, TBS = Transfers Between Services, TIW 27 = Transfer Into Ward: 27, HA = Hospital Admission.

Patient Identifier	Event	Timestamps	EntityType	Admission
Patient_4900	L_Taken	2013-10-29T05:00:00	Lab. Measurement	Outpatient
Patient_4900	LAM	2013-10-30T06:00:00	Lab. Measurement	Outpatient
Patient_4900	CA DM HL HT	2013-10-31T07:00:00	Diagnosis	115281
Patient_4900	TBS	2013-10-01T08:00:00	Logistic	115281
Patient_4900	TIW 27	2013-10-19T09:00:00	Logistic	115281
Patient_4900	HA	2013-10-29T06:00:00	Logistic	174010
...	...	...	...	...

4.2 Building the Event Graph

We showed the steps we followed to create the event graph from the multi-entity event data based on the approach introduced in [7] in Figs. 1 and 2: ① Each record of the event log was converted to a node, called event node; then another node was created for the event log. After that, relationships from each event node to the log node was created. ② Nodes for the cases' entities and their properties, called entity nodes was generated, then each event node was correlated to its relative entity node. ③ The entities nodes were related to each other based on their event's sequential occurrence. ④ The relationship between the entities nodes were reified. ⑤ Directly follows relation between the events node was created based on entities and properties, and ⑥ Event class nodes and property class nodes were created respectively for distinct events and properties, and finally aggregated directly follows relationships for the event and property class nodes were created.

5 Results of Application to MIMIC-III

A preliminary evaluation of our approach relies on a qualitative discussion. We analyze the generated multi-entity directly follows graphs from the MIMIC-III database and evaluate to which extend they support our hypothesis. We implemented the event graph creation using Python and the Neo4J library and adapted the code provided by [7] for our case[1]. Multi-entity directly follows graphs were discovered by querying the event graph with CQL and visualized it with Graphviz.

The multi-entity directly-follows graphs of two patients are shown in Figs. 3 and 4. These two patients, Patient_4900 and Patient_14606, are examples of

[1] Available on https://github.com/mnaeimaei/MIMICIII-Event-Graph.

multi-morbid patients who have been admitted to the hospital several times and had more than one disease at each time of admission.

Based on Fig. 3, before hospital admission, a laboratory measurement was taken (L Taken Node) for Patient_4900, the abnormal measurements (LAM node) of laboratory test is one of the bases for diagnosing diseases for that patient. The patient was admitted to the hospital three times. In each of them, several diseases were diagnosed for the patient, and after that, several activities related to **Logistic**, **Laboratory_Measurement** and **Prescriptions** entities happened for the patient. In the first, second, and third admission, respectively, four, six, and four diseases were diagnosed for the patient. The activities for **Logistic**, **Laboratory_Measurement** and **Prescriptions** entities is different in each admission because there is difference between diagnoses diseases of each three admission. It means the activities done for patients are related to their diseases. We can see that the disease CA (Coronary Atherosclerosis) and DM (Diabetes Mellitus type II) was diagnosed in all three admissions, which indicate some common activities related to entities have occurred in all three times of admission. On the other hand, we have diseases such as HL (Hypercholesterolemia), HH (Hemorrhage), MN (Malignant neoplasm), MF (Myocardial Infarction), which were diagnosed in only one admission time. It shows that first, there are unique activities related to entities related to this disease. Second, they were treated in hospital.

According to Fig. 4, the patient was admitted to the hospital without any laboratory measurement, which means that patient diagnoses related to the first admission are not related to previous measurements. For patient_14606, a group of diseases was diagnosed in the patient's first admission: CA (Coronary Atherosclerosis), CS (Coronary Syndrome), HD (Hyperlipidemia), HM (Hypothyroidism), HT (Hypertension). After that several activities related to **Logistic**, **Laboratory_Measurement** and **Prescriptions** entities were conducted for treating those diseases. After the first patient admission, a laboratory test was taken that was used as the basis of diagnoses for the second admission. In the second admission of Patient_14606 another group of diseases was diagnosed: DM (Diabetes mellitus), CC (Carotid Artery Occlusion), VD (Vascular Disease), HL (Hypercholesterolemia), HM (Hypothyroidism), HT (Hypertension) since then activities related to **Logistic**, **Laboratory_Measurement** and **Prescriptions** entities happen. In the third admission of Patient_14606, another group of diseased were diagnosed: CH (Congestive Heart Failure), CD (Cardiac Dysrhythmia), HM (Hypothyroidism), CC (Carotid Artery Occlusion). For the Patient_14606, we can see that diseases related to coronary disease were not diagnosed in the second and third time, indicating activities in the first admission treated these diseases. Also, diseases are repeated in all three admissions, which indicates these diseases are chronic diseases or the activities are done for the patient were not useful.

6 Discussion

Based on the Figs. 3 and 4, discovered multi-entity directly follows graph for those patients show all traditional process mining concepts (e.g., sequence of

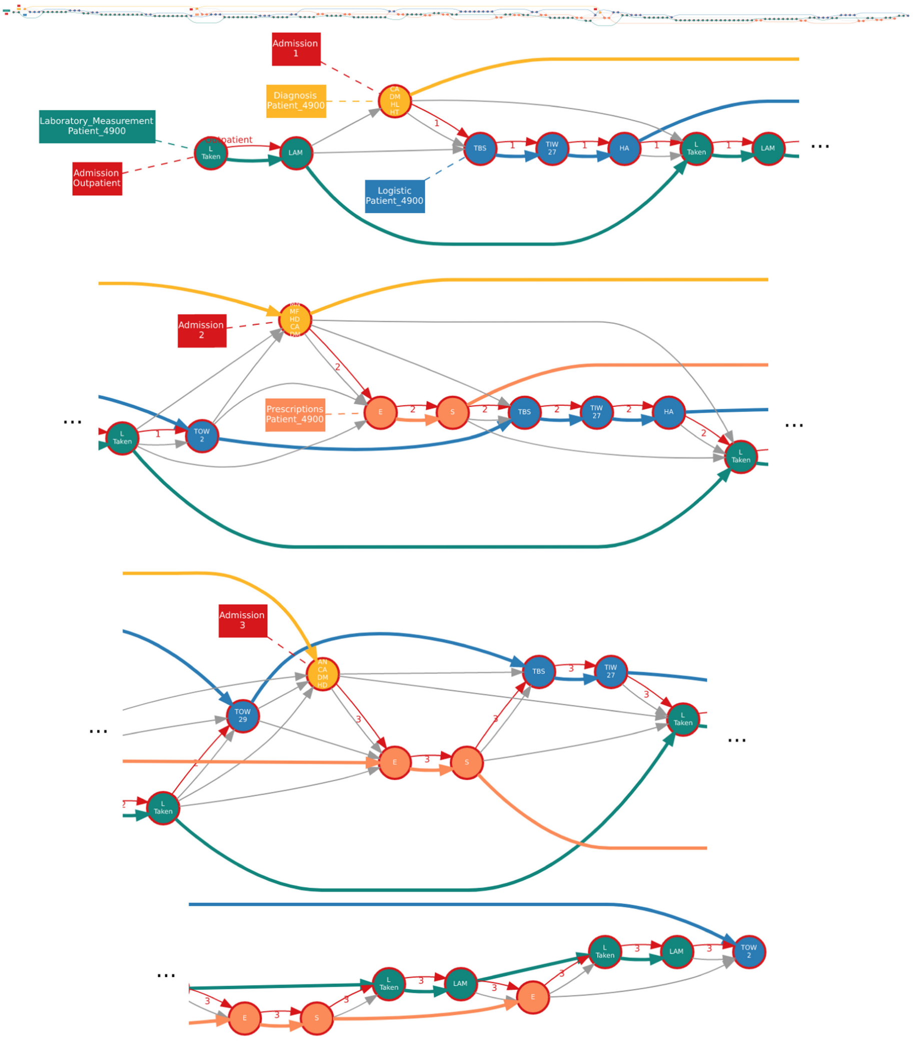


Fig. 3. Multi-entity directly-follows graph for Patient_4900. (Color figure online)

activities) and for all involved clinical processes in only one graph. Meanwhile, the relationship between the different clinical processes activities that were not detectable in traditional models was clearly shown in discovered directly follows graph. This graph shows how diagnoses for multi-morbid patients evolved during the care pathways and how these diagnoses relate to other events, and how the trajectory of patients varies for each group of diagnoses.

The multi-entity directly-follows graph of **Patients_4900** and **Patient_14606** involves four entities which each of which has been shown with different colors. Before the first Admission of the **Patients_4900**, the patient had

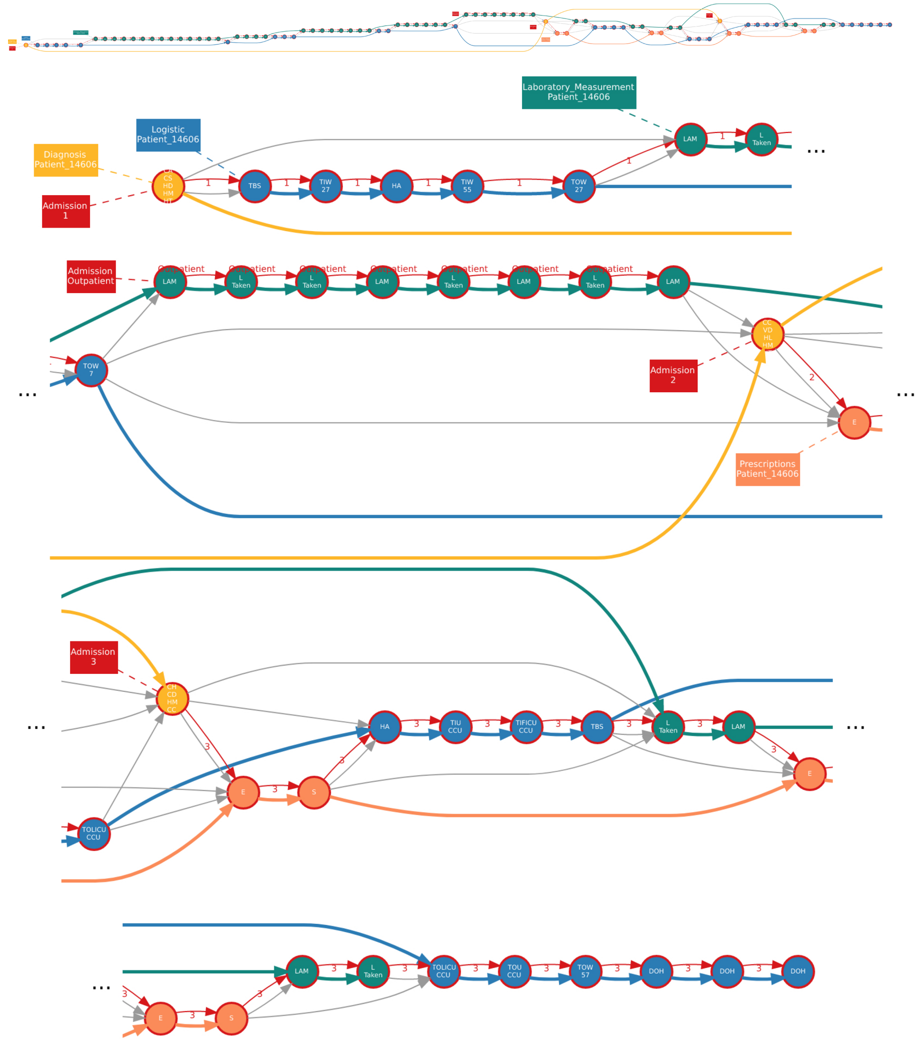


Fig. 4. Multi-entity directly-follows graph for Patient_14606 (top) and details (bottom) (Color figure online)

abnormal values related to out-of-hospital laboratory measurements from clinics which the patient had visited. These measurements can be one of the bases for diagnosing diseases for the first Admission of that patients. These diseases were shown in **Diagnoses** entity. Meanwhile, in discovered graphs, the admission number of patients was indicated by separate red edges.

These graphs demonstrate that analyzing care pathways of patients with multi-morbidity is completely applicable using an event graph. The discovered graphs for distinct patients can illustrate all single-entity concepts such as activities, cases, and their properties for all entities simultaneously. Based on these results, the hypotheses of the research, applying event graphs produce valuable insights when using multi-entity event data for clinical pathways of multi-morbid patients, seems to be valid.

7 Conclusions

In this research, we could discover insightful graphs comparing traditional process mining by using multi-entity event data stored in an event graph. We evaluate the potential of the event graph approach proposed by Essser and Fahland [7] for clinical data by using the MIMIC-II database. Some of the limitations of this paper are related to the case study, such absence of resources in the MIMIC-III database and shifting times. Another limitation is related to missing visualization methods for multi-entity event data. Creating appropriate visualization approaches and automating process discovery can be future research. Enabling to show sub-processes inside an event is a highly insightful capability for graphs, which can be future work. As well, multi-entity graph notations need to be researched and created.

References

1. UN and United Nations, Department of Economic and Social Affairs, Population Division (2017)
2. Marengoni, A., et al.: Aging with multimorbidity: a systematic review of the literature. Ageing Res. Rev. **10**(4), 430–439 (2011)
3. Schrijvers, G., van Hoorn, A., Huiskes, N.: The care pathway: concepts and theories: an introduction. Int. J. Integr. Care **12**(Special Edition Integrated Care Pathways), e192 (2012)
4. Fernandez-Llatas, C. (ed.): Interactive Process Mining in Healthcare. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-53993-1
5. van der Aalst, W.M.P.: Data science in action. In: Process Mining. Springer
6. Artale, A., Kovtunova, A., Montali, M., van der Aalst, W.M.P.: Modeling and reasoning over declarative data-aware processes with object-centric behavioral constraints. In: Hildebrandt, T., van Dongen, B.F., Röglinger, M., Mendling, J. (eds.) BPM 2019. LNCS, vol. 11675, pp. 139–156. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26619-6_11
7. Esser, S., Fahland, D.: Multi-dimensional event data in graph databases. J. Data Semant. **10**, 109–141 (2021). https://doi.org/10.1007/s13740-021-00122-1
8. Martin, N., et al.: Recommendations for enhancing the usability and understandability of process mining in healthcare. Artif. Intell. Med. **109**, 101962 (2020)
9. Aalst, W.M.P.: Object-centric process mining: dealing with divergence and convergence in event data. In: Ölveczky, P.C., Salaün, G. (eds.) SEFM 2019. LNCS, vol. 11724, pp. 3–25. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30446-1_1
10. Jeevan, J., Sureka, A.: Graph or relational databases: a speed comparison for process mining algorithm. arXiv preprint arXiv:1701.00072 (2016)

11. Jalali, A.: Graph-based process mining. In: Leemans, S., Leopold, H. (eds.) ICPM 2020. LNBIP, vol. 406, pp. 273–285. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-72693-5_21
12. Klijn, E.L., Mannhardt, F., Fahland, D.: Classifying and detecting task executions and routines in processes using event graphs. In: Polyvyanyy, A., Wynn, M.T., Van Looy, A., Reichert, M. (eds.) BPM 2021. LNBIP, vol. 427, pp. 212–229. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-85440-9_13
13. Vogelgesang, T., Appelrath, H.J.: Multidimensional process mining: a flexible analysis approach for health services research. In: Proceedings of the Joint EDBT/ICDT 2013 Workshops (2013)
14. Johnson, A.E.W., et al.: MIMIC-III, a freely accessible critical care database. Sci. Data **3**(1), 1–9 (2016)

TPSA 2021: 2nd International Workshop on Trust, Privacy, and Security in Process Analytics

2nd International Workshop on Trust, Privacy, and Security in Process Analytics (TPSA)

Trust, privacy, and security aspects have been considered in process mining research from two fundamental perspectives:

A. the *application of process mining* to investigate whether systems are trustworthy, whether privacy regulations are adhered to, and whether security properties of systems have been (actively) violated;
B. the *analysis of and the design of process mining techniques* such that *trust*, *privacy*, and *security* properties are provided when applied to analyze processes.

The main objective of the TPSA workshop is to give a forum for the trust, privacy, and security aspects and the responsible application of process mining including other concerns such as fairness, transparency, and accuracy. We invite researchers and industry to share their research, ideas, experience reports, and challenges in this area. Perspective (A) relates closely to the responsible application of process mining in the broader context of *responsible data science*, i.e., how to use process mining while guaranteeing criteria such as FACT (Fairness, Accuracy, Confidentiality, Transparency). The other perspective (B) is using process mining to determine whether other systems exhibit behavior that is desired from a trust, privacy, or security viewpoint.

Here privacy relates to the concern that event logs may contain personal data of both customers and employees and the challenge of protecting the information about individuals while still being useful for process mining (e.g., differential privacy, k-anonymity, homomorphic encryption, secure multi-party computing). However, process mining could also be used to investigate whether privacy regulations are being followed and pinpoint compliance violations. Often, security aspects (e.g., encryption) are closely connected when processing personal data cannot be avoided. On the other hand, the workshop is about the concept of trust, which is required both from the perspective of trust in organizational and technological measures that event logs are not misused (e.g., for worker surveillance) as well as from the perspective of trust that the results of a process mining analysis faithfully reflect reality (e.g., data quality, traceability, auditability).

We received six papers that cover all three topics and both perspectives. From them, we were able to accept three full papers for presentation and inclusion in the workshop proceedings. In addition, we have solicited researchers to send short papers on their research-in-progress that did not meet the acceptance criteria or was not yet mature enough for a full paper. We received four extended abstracts for this session, which are not part of this proceedings.

“Trustworthy Artificial Intelligence and Process Mining: Challenges and Opportunities” was presented as first paper in the workshop. It gave a broad view on how process mining relates to the field of trustworthy Artificial Intelligence not only from a technical but also a regulatory standpoint. The second paper “Process Mining in Trusted Execution Environments: Towards Hardware Guarantees for Trust-aware Inter-organizational Process Analysis” proposes a technical solution for enabling process mining across data from several organizations. The full paper session was wrapped up with the paper "Quantifying the Re-identification Risk in Published Process Models" that provided an extension of a method to quantify the re-identification of individuals from event logs to the risk of re-identifying individuals from published process models.

The following four papers were presented in the research-in-progress session: "BERMUDA: Towards Maintainable Traceability of Events for Trustworthy Analysis of Non-process-aware Information Systems", "a Generalizable Approach for Determining The Sensitivity of A Trace within An Event Log", "Utility-aware Event Log Anonymization for Privacy-Preserving Process Mining?" and "Conceptualizing a Log Generator for Privacy-aware Event Logs".

Around 20 attendees were present during the workshop presentations and panel discussion. Due to the generous support of the ICPM organizers, we have been able to award the two best presentations. The Best Presentation Award of the TPSA workshop in 2021 went to Marcel Müller who presented the paper "Process Mining in Trusted Execution Environments: Towards Hardware Guarantees for Trust-aware Inter-organizational Process Analysis". The Runner-up Award was given to Paul Cosma for presenting his work on "BERMUDA: Towards Maintainable Traceability of Events for Trustworthy Analysis of Non-process-aware Information Systems".

Organization

Organizing Committee

Felix Mannhardt	Eindhoven University of Technology
Agnes Koschmider	Kiel University
Nathalie Baracaldo	IBM Almaden Research Center

Program Committee

Luciano Garcia Bañuelos	Tecnologico de Monterrey, Mexico
Stephan Fahrenkrog-Petersen	Humboldt University of Berlin
Marwan Hassani	Eindhoven University of Technology
Judith Michael	RWTH Aachen
Florian Tschorsch	Technische Universität Berlin
Moe Wynn	Queensland University of Technology
Sebastiaan van Zelst	RWTH Aachen/FIT
Nicola Zannone	Eindhoven University of Technology
Xixi Lu	Utrecht University

Process Mining in Trusted Execution Environments: Towards Hardware Guarantees for Trust-Aware Inter-organizational Process Analysis

Marcel Müller[1,2](✉), Anthony Simonet-Boulogne[3], Souvik Sengupta[3], and Oliver Beige[2]

[1] Technische Universität Berlin, Berlin, Germany
marcel.mueller@tu-berlin.de
[2] JadenX Research, Berlin, Germany
{marcel.mueller,oliver.beige}@jadenx.com
[3] iExec Blockchain Tech, Lyon, France
{anthony,souvik}@iex.ec

Abstract. Process mining techniques enable business process analysis on event logs extracted from information systems. Currently, industry applications and research in process mining predominantly analyze intra-organizational processes. Intra-organizational processes deal with the workflows within a single organization. However, analyzing inter-organizational processes across separate companies has the potential to generate further insights. Process analysts can use these insights for optimizations such as workflow improvements and process cost reductions. It is characteristic for inter-organization process analysis that it is not possible to uncover the insights by analyzing the event logs of a single organization in isolation. On the other hand, privacy and trust issues are a considerable obstacle to adopting inter-organizational process mining applications. The independent companies fear competitive disadvantages by letting third parties access their valuable process logs. This paper proposes a concept for inter-organizational process mining using trusted execution environments in a decentralized cloud. The hardware-based approach aims to technically prevent data leakage to unauthorized parties without the need for a trusted intermediary. The contributions of this paper are theoretical and identify future research challenges for implementing the concept.

Keywords: Process Mining · Privacy · Trusted Execution Environments · Inter-organizational Process Mining

1 Introduction

Process mining analyzes the real-world execution of business processes. The analysis utilizes event logs extracted from information systems to construct a

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 369–381, 2022.
https://doi.org/10.1007/978-3-030-98581-3_27

business process model [1]. A variety of process mining techniques and configurations enable analysts to derive different insights into their processes. Process mining can be used to identify compliance violations, find process bottlenecks, and investigate the root causes of undesired process behavior. Usually, process mining analyzes processes within a specific organization (intra-organizational processes). Yet, in practice, inter-organizational workflows are standard in various industries. In an inter-organizational business process, different organizations execute separate parts of a shared workflow. Examples of such processes include e-commerce, supply chain management, or international bank transactions. However, analyzing data from other companies is trust- and privacy-intensive [2–4]. Event logs record valuable information of an organization's real-life operation details. These details can be exploited to analyze a collaborator's internal processes and to gain competitive advantages. Thus, many organizations refrain from participating in inter-organizational process mining and optimization. However, mining inter-organizational processes as a whole can enable different insights and benefits that cannot be derived from analyzing the private processes of collaborators in isolation. All parties may benefit from such a high-level analysis.

Figure 1 illustrates an example inter-organizational hiring processes. The process model shows how a certain company (the *seeking company*) finds new employees for its software development jobs. Since recruiting processes are time- and cost-intensive, the company outsources the initial recruiting task to three independent recruiters. The process starts with the seeking company defining the job requirements. Afterward, the organization contacts three different recruiters in parallel for the first-level candidate screening. Their task is to find the best-suited candidates for their job opening. The three recruiters have different strategies to find the best candidates. Recruiter 1 approaches the task by searching candidates on professional platforms like LinkedIn[1]. The recruiter sends cold messages to candidates and conducts a general pre-interview with them. The objective of the pre-interview is to find out if all formal requirements for the candidate to become a potential employee are fulfilled. This might include, for instance, having the right working permits. Afterward, Recruiter 1 conducts a technical interview to see if the candidate has the right skill set for the job. In the end, Recruiter 1 decides whether the candidate is suited for the position. If yes, the recruiter forwards the CV to the seeking company. Recruiter 2 has a different approach. This recruiter makes a job post on an open online job board like Indeed[2]. After a while, the recruiter receives some applications and assesses the CVs of the candidates. Recruiter 2 does not conduct general interviews and proceeds directly to the technical interview. After the technical interview, this recruiter also decides and forwards the candidate to the seeking company. Recruiter 3 starts the candidate search on a professional platform, like Recruiter 1. However, Recruiter 3 is not a technical expert and does not conduct technical interviews after the general pre-interview. All three recruiters forward

[1] https://www.linkedin.com/.

[2] https://de.indeed.com/.

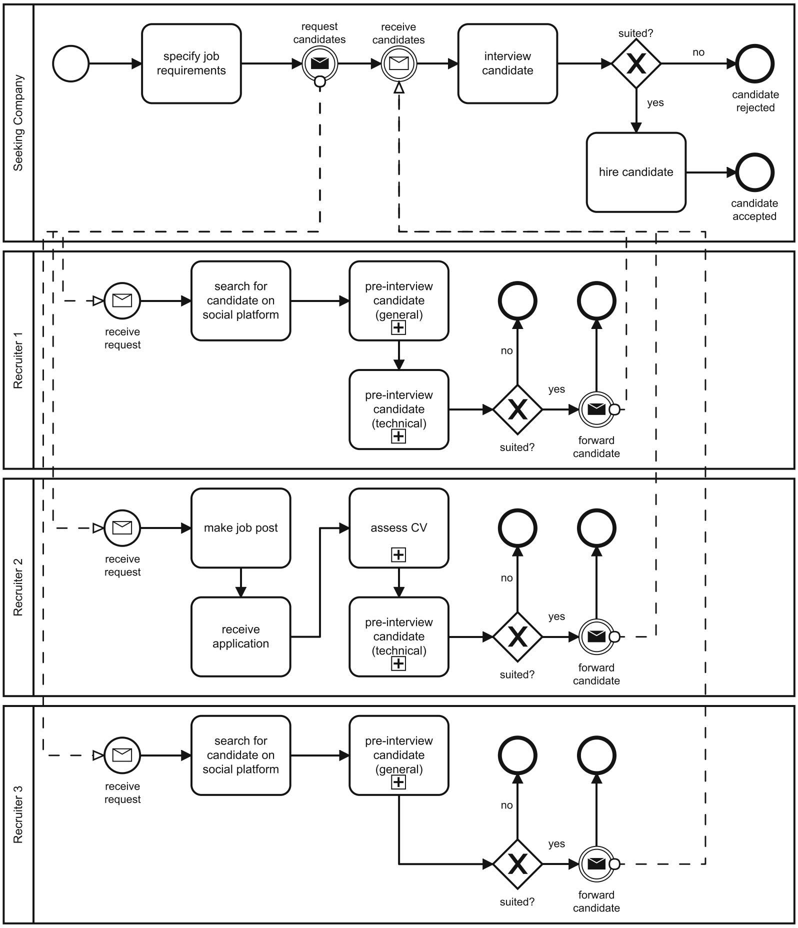


Fig. 1. Example inter-organizational process in human resources using the BPMN 2.0 standard [5].

their candidates to the seeking company, where a final interview is conducted. Afterward, the decision of whether or not to hire the candidate is made.

This example shows an inter-organizational process where all four organizations act independently. Yet, they have a common goal to make to recruiting process as efficient as possible. Especially the seeking company wants to reduce their recruiting time. Their final interviews are conducted by the seeking company's most skilled tech specialists. This circumstance makes every final interview cost-intensive. In such a case, gathering the event logs from all recruiters and applying process mining techniques can help determine process dependencies and causalities. However, recruiters do not want to disclose details of their recruiting process to third parties. They experimented in the past to find the best recruiting strategy. Thus, by sharing their detailed approaches, they could have competitive disadvantages.

This paper proposes a theoretical concept of executing process mining tasks in trusted execution environments (TEEs). TEEs are hardware-based approaches that enable processing data in a secure enclave. Thus, there is no *technical* possibility to leak the information to unauthorized third parties. The outcome is a theoretical concept and an identification of research challenges that need to be solved before the concept can be implemented in practice.

The remainder of this paper is structured as follows. Section 2 reviews the current state of the art in privacy-preserving inter-organizational process mining and trusted computing. Afterward, Sect. 3 introduces our novel concept for privacy-aware process mining in TEEs. Section 4 discusses the implementation challenges of the concept, before Sect. 5 concludes on the impact of this scientific contribution.

2 Related Work

Recently, privacy aspects of inter-organizational process mining have seen an increase in academic and professional interest. The current state of the art in privacy-preserving process mining approaches can be divided into two main segments. The first segment focuses on privacy preservation of information related to a *individuals* encoded in a process log, e.g., employee information. The other group of approaches focuses on protecting the information of an *organization* and its business secrets.

Individual-focused privacy-preserving process mining approaches focus on the privacy of the information of individuals that are included in event logs. These concepts have applications in fields like individual health care or manufacturing workflows [6]. For instance, a large hospital might want to analyze its emergency room response processes. Therefore, they need data related to specific cases of emergency room arrivals. The event log might include information specific to a patient. Regulatory frameworks like the General Data Protection Regulation (GDPR) [7] or the Health Insurance Portability and Accountability Act of 1996 (HIPAA) [8] require this data to be protected. Current research on individual privacy-preserving process mining employs concepts such as differential privacy and k-anonymity [9–11]. In addition, other approaches employ encryption as the main method of privacy preservation. These concepts use standards like the Advanced Encryption Standard (AES) [12] or the Pallier Cryptosystem [13] to encrypt personal information encoded in the event log. Such cryptography-based concepts often also enable privacy preservation of business-related information as well [14,15].

Organization-focused privacy-preserving process mining techniques have the ultimate goal to protect business-related information. The leakage of valuable organizational information to third parties might lead to compliance issues and competitive disadvantages. Thus, especially inter-organizational business processes pose a specific challenge to such organization-focused privacy-preserving process mining techniques. It is characteristic of inter-organizational process mining to acquire event logs from different organizations. These joint insights may generate a greater value than the insights created from separate event logs in an isolated manner. Current research in this field proposed different concepts for executing the mining process. One approach is to mine partitioned data in a decentralized fashion. The partitions can be used to generate a common model without revealing the raw data [16]. The common model may differ for organizations since they all have different knowledge. Cryptography-based approaches encrypt the valuable information [14]. Lately, also approaches that use private computing paradigms such as homomorphic encryption [15] or secure multi-party computation [17] emerged. Both of these approaches are famously known as the privacy-preserving computation (PPC) methodologies. Lately, trusted execution environments (TEEs) have been introduced [18]. TEEs are a hardware-level privacy-aware computation paradigm. This paper presents a theoretical concept for organization-focused privacy-preserving process mining using TEEs.

Trusted execution environments (TEEs) are a hardware-based approach for trusted computation provided by some modern micro-processors, e.g. Intel *Software Guard Extension* (SGX) [18] and ARM Trustzone [19]. The main component of a TEE is a secure element that resides within a separate area of the CPU chip. Code and data in the secure element are entirely isolated from other programs and from the host operating system. This paradigm protects the data from theft and the code from tampering. TEEs, and Intel SGX in particular, provide low-level primitives for defining specific rules (e.g. which software package can decrypt a dataset). These rules are enforced by using hardware-based cryptography. However, expressing complex multi-processor workflows like the one we have described above requires a higher-level rule system. This rule system is in charge of orchestrating the encryption of the input data, the provisioning of several secure enclaves, and the dataflow between them. Distributed ledger technologies (DLTs) offer a decentralized execution environment with an immutable record of transactions. Thus, DLTs can be a well-suited platform for orchestrating process mining. The organizations can use smart contracts to define authorizations, to record job requests, and to verify remote attestations (proof of a correct execution in TEE) with no risk of their intent being altered. Current approaches combining distributed ledgers (for expressing rules) and TEEs (for enforcing them) for trusted computing include iExec [20] and Ekiden [21].

The primary purpose for adopting the hardware-based TEE (HW TEE) in our contribution unfolds as follows. Besides enabling data integrity and confidentiality, HW TEEs also ensure code integrity, code confidentiality, programmability, attestability, recoverability, and authenticated application launch facilities. Thus, these characteristics make the HW TEE a suited option for the organizations for doing privacy-preserving process mining.

3 A Concept for Privacy-Aware Process Mining in Trusted Execution Environments

In this paper, we introduce a theoretical concept on how TEEs and blockchains can be used for privacy-aware inter-organizational process mining. The setup of the orchestration of the TEEs is inspired by the iExec decentralized cloud computing framework[3]. However, the general concepts presented in the following are independent of any framework to orchestrate TEEs.

3.1 System Architecture

Our concept consists of different system components. The following paragraphs introduce them and their workflows in the inter-organizational mining process. The architecture diagram in Fig. 2 visualizes the interactions.

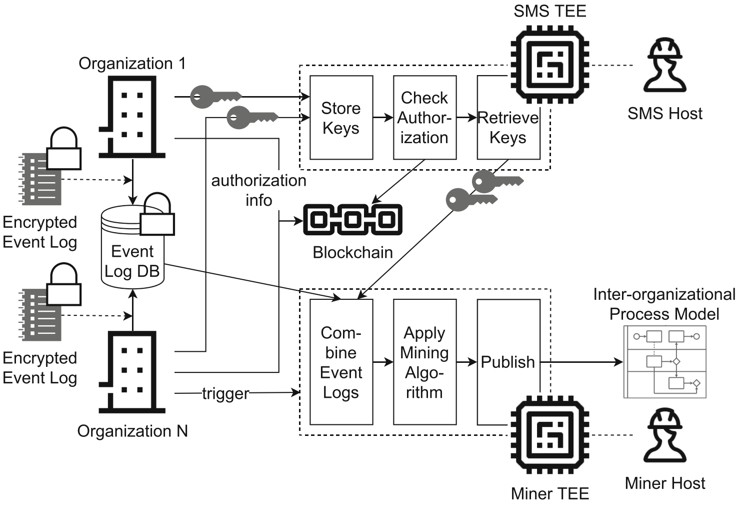


Fig. 2. Architecture diagrams of different roles interacting with each other.

Organizations. In our concept, N organizations have their private information system where they acquire new events. These private information systems are isolated from each other.

Secret Management System. The secret management system (SMS) is a key component that acts as a secure intermediary between the organizations that provide data and the process miner which processes it. Because TEEs require

[3] https://iex.ec.

data be encrypted specifically for a given enclave session, the workflow could not run asynchronously without it. In our example, this would translate to having the organizations encrypt their logs after the process mining has started. In our design, the SMS itself is a secure application running in an enclave. It receives the decryption keys of all the data from the organizations and manages it according to the authorized orders. The orders are signed and recorded on the blockchain. The SMS is thus a critical component that holds all of the decryption keys to every log file, but the fact that it runs in a TEE guarantees that only no one and nothing besides its code can access them, not even the administrator of the machine it is running on.

Blockchain. In our concept, we use the blockchain as tamper-proof storage of authorization statements using smart contracts [22]. The SMS is only allowed to give the keys to authorized entities. Thus, the organizations create transactions to trigger smart contracts stating which miner can retrieve the keys for a specific order.

Event Log Database. The public event database stores the encrypted event log files of all organizations. This ensures ensure the integrity of the private logs and enables the inter-organizational mining process to retrieve them. The database host never has access to the keys of the event logs.

Miner. The miner is responsible for applying process mining techniques to the combined event logs of different organizations. The three-step subprocess consists of combining the event logs, mining the process models, and making the insights available to the organizations. All mining tasks are executed in a TEE so that the miner host cannot interfere.

3.2 Workflow

The following sections describe the process of privacy-preserving process mining using TEEs in detail. Figure 3 shows the process model of our concept.

Prerequisites. We assume that the following activities happened before the core process. All N organizations need to synchronize the case ids and select a process mining technique upfront. In process mining, a *case* is a unique identifier that groups a set of events. All events in a case belong logically together. In the running example, every instance of a recruiting process of a backend developer consists of different events. Such events may be that a recruiter found a new candidate or that the hiring company made a decision. All events that belong to the same *instance* of the process can be grouped together in a case. A sequential order of timestamped events within a case is called a *trace*. The organizations might use different systems to track the events that fall into their domain. Thus, they need to synchronize case identifiers. In that way, the process miner can later merge different sub-traces that belong to the same case in a TEE. It is also

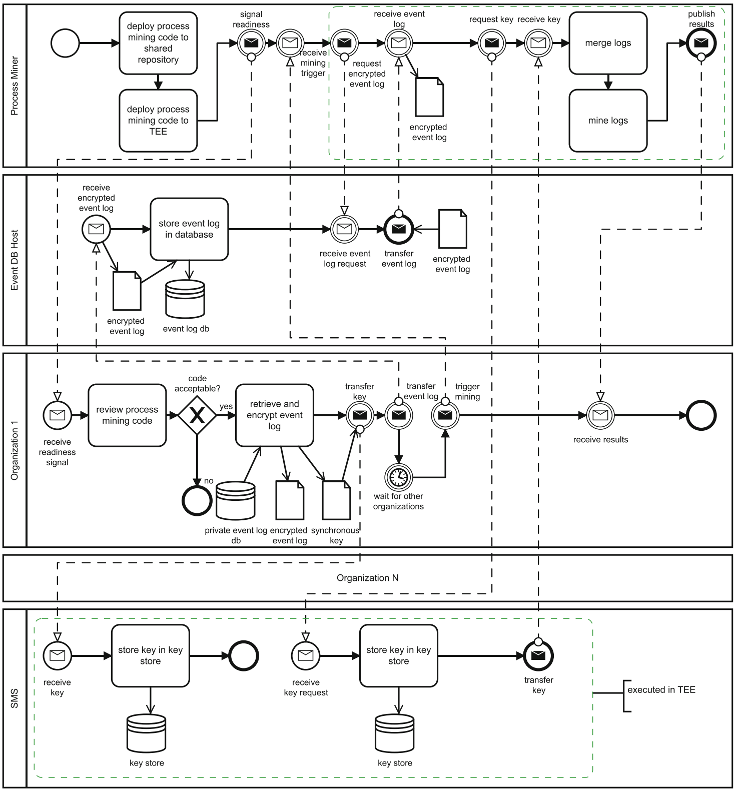


Fig. 3. Illustration of the high-level process of privacy-preserving process mining using TEEs. The green boxes indicate workflow executed within a TEE. The diagram uses the BPMN 2.0 standard for illustration [5]. The collapsed pool indicates a variable number of organizations that all follow the same logic. (Color figure online)

required that the process miner is in possession of an implementation of the process mining technique. Finally, each organization generates a unique symmetric encryption key and uses it to encrypt an archive containing all of their log files.

Initialization. The core workflow component for privacy-aware process mining in TEEs is the process mining code, which must be audited and approve it by all of the organizations. In practice, the source code is shared in a repository that all organizations can access. All organizations audit the code and if approved they record a rule linking the hash value of the packaged code to the hashed value of the encrypted archive. This requirement ensures that the SMS will later give the right program access to the right data. In our concept, the rules are recorded in a blockchain smart contract [23] to preserve the integrity of the hashes in a decentralized fashion. All organizations have blockchain peers and guarantee the integrity of transactions.

Review and Data Contribution. After the deployment phase, the process miner signals readiness to all N organizations. The organizations can review the process mining code. In this review, they can assess if the code meets all privacy and compliance requirements. In case an organization decides that the code does not fulfill its requirements, the process terminates. After a positive code review, the organizations retrieve their event logs from their private information systems. They encrypt their event logs separately. Therefore, an organization needs to encrypt its event log using a symmetric encryption mechanism, such as AES-256 [12]. A hash of the encrypted data set is submitted to the blockchain with a transaction. The organizations can later use the hash to ensure the integrity of the data set. The encrypted event log itself is stored at an independent host as an encrypted file. The organizations share their symmetric key with a secret management service (SMS) of their choice. This SMS is a simple program that lets only authorized entities access keys. The right functionality of the SMS can be guaranteed because the SMS program is also executed in a TEE. Its code is open source. Thus, every entity can audit its code. Through the attestation that a TEE produces, it is possible to prove that only the desired program has been executed and nothing else.

Mining. A organization needs to trigger the process mining task. This trigger is expressed through a blockchain transaction as an *execution order* that needs to be signed by the requesting organization. The execution order specifies which code (the selected process mining technique) should be executed and which input data sets (the N event logs of the organizations) should be mined. The process miner starts the mining process in a TEE. In that way, the host does not have any influence on the execution of the mining program, and a remote attestation proves the correct execution. First, the process miner requests all the encrypted event logs from the independent event data storage; then, it makes a request to the SMS to obtain the keys to decrypt the process logs. The SMS uses TEE primitives to verify that the miner is actually running in an enclave. The SMS only allows it to access the keys if there is an order that assigns the miner to a task that includes the data sets of the respective organizations. The correctness of this logic can be guaranteed through the attestation of the TEE. After the process miner received and decrypted all event logs in the enclave, the merging of the logs begins. While merging, the different sub-traces of the separate organizations are used to end-to-end traces. This trace reflects the full inter-organizational processes with all the sub-processes of the collaborators. The merging process yields a full event log that the selected process mining technique can then mine.

Insights. In the end, the process mining TEE compiles the aggregated result of the mining process. These result is a joined process model that encompasses the whole inter-organizational process. Furthermore, the TEE distributes them to all N organizations. In that way, the N organizations only get insights from the merged and aggregated process. However, they cannot get any insights into the sub-processes of a specific organization.

3.3 Security Comparison

The following sections discuss the security and privacy features of our proposed approach to currently existing technical foundations for privacy-preserving process mining. Secure multiparty computation (SMPC), homomorphic encryption (HE), and differential privacy (DP) enable different mechanisms for ensuring privacy and confidentiality in inter-organizational process mining. SMPC keeps the executing system from exposing the input data [24]. However, it does not provide any guarantees for the output data. HE mitigates potential vulnerabilities in the storage or computing environment from compromising the data. However, in the case of HE, if some party gets the access privilege, the authorized party can easily access entire datasets [25]. DP provides a layer of privacy by obfuscation in case some data concerning individual entities leaks. Yet, it can not counteract vulnerabilities in the infrastructure used to store or manage the data [25]. Thus, executing process mining code in a hardware TEE differs from the current concepts for privacy-preserving process mining. It ensures complete computation confidentiality through memory encryption at the hardware level. Inputs and outputs to computing tasks are encrypted. This makes hardware-based TEEs suitable for developing our multi-organizational trusted process mining framework, as long as the user trusts the hardware design.

4 Implementation Challenges

To implement our presented concept, we need an orchestration layer that can provide provision TEE resources for process mining tasks. Therefore, we adopt a *decentralized cloud* paradigm. There, workers can contribute their TEE resources to a process mining task. We adopt this paradigm so that the organizations do not have to deal with the overhead of setting up TEE resources on their premises. Furthermore, the incentivization mechanism ensures that attestations are always distributed to all involved organizations.

Currently, the iExec framework [20] and Hyperledger Avalon [26] are the two only decentralized cloud computing frameworks that can orchestrate Intel SGX enclaves [18]. Both utilize the blockchain to store orders, resource allocations, and attestation securely. In the following, we explore the steps needed to implement the presented concepts using iExec. We make this choice because it is more advanced in its development maturity than other approaches. However, our principles are independent of any framework.

The iExec worker infrastructure is deployed on top of the Ethereum blockchain. A suite of support tools allows anyone to record TEE applications packaged as Docker containers. Data management tools enable the management of encrypted data sets and setting fine-grained authorization rules. These authorizations include which application can access which data set and which users can trigger an execution. The authorization is implemented in iExec with a secret management service (SMS) similar to our proposed inter-organizational process mining concept.

While the iExec framework is the most suited candidate to implement our concepts, further development is required to support some specific needs for process mining. Namely, this includes the possibility of assigning several data sets to a single execution and distributing the result to multiple users. At the time of writing, iExec does not support consuming multiple data sets in a single execution. Furthermore, the SMS in iExec is in a prototype stage. Its full implementation in an Intel SGX TEE is still not complete. The upcoming release of SGX 2 CPUs by Intel should significantly improve the performance and scalability of the service.

Once these challenges are overcome, the novel approach to privacy-preserving inter-organizational process mining as presented in this paper can be researched, implemented, and evaluated further.

5 Conclusion

In this paper, we introduced a novel concept for privacy-aware inter-organizational process mining using trusted execution environments. The contributions are theoretical. We identified challenges for future work that need to be solved to implement the concept.

Our concept can enable process mining in application domains with sensitive data that currently do not utilize process analysis in cross-organizational processes. The first application area we foresee is supply chain management. Several logistics companies must collaborate to transport a parcel from a sender to a receiver in international deliveries. All companies want to optimize their workflows as much as possible. The inclusion of the whole inter-organizational process could help optimize shipping times and improve customer satisfaction. Another application area is fraud in finance. Currently, detecting money laundry circles requires transaction logs from different banks. Due to the privacy requirements of their customers, banks are reluctant to share data with any third party. Introducing our concept for money laundry detection could build trust since all processing steps of shared data can be audited, and the TEEs guarantee that no other unauthorized code is executed.

References

1. van der Aalst, W.: Process Mining: Data Science in Action. Springer, Berlin (2016). https://doi.org/10.1007/978-3-662-49851-4
2. Müller, M., Ostern, N., Koljada, D., Grunert, K., Rosemann, M., Küpper, A.: Trust mining: analyzing trust in collaborative business processes. IEEE Access **9**, 65044–65065 (2021)
3. Müller, M., Garzon, S.R., Rosemann, M., Küupper, A.: Towards trust-aware collaborative business processes: an approach to identify uncertainty. IEEE Internet Comput. **24**(6), 17–25 (2020)
4. Elkoumy, G.: Privacy and confidentiality in process mining-threats and research challenges (2021). arXiv preprint: arXiv:2106.00388

5. OMG. Business process model and notation (BPMN), version 2.0. https://www.omg.org/spec/BPMN/2.0/PDF. Accessed on 29 July 2021
6. Mannhardt, F., Petersen, S.A., Oliveira, M.F.: Privacy challenges for process mining in human-centered industrial environments. In: 2018 14th International Conference on Intelligent Environments (IE), pp. 64–71. IEEE (2018)
7. Directive 95/46/ec (general data protection regulation). https://eur-lex.europa.eu/eli/reg/2016/679/oj. Accessed 30 July 2021
8. Health insurance portability and accountability act of 1996 public law 104–191 (1996). https://www.govinfo.gov/content/pkg/PLAW-104publ191/html/PLAW-104publ191.htm. Accessed 30 July 2021
9. Mannhardt, F., Koschmider, A., Baracaldo, N., Weidlich, M., Michael, J.: Privacy-preserving process mining. Bus. Inf. Syst. Eng. **61**(5), 595–614 (2019)
10. Fahrenkrog-Petersen, S.A., van der Aa, H., Weidlich, M.: PRETSA: event log sanitization for privacy-aware process discovery. In: 2019 International Conference on Process Mining (ICPM), pp. 1–8. IEEE (2019)
11. Fahrenkrog-Petersen, S.A., van der Aa, H., Weidlich, M.: PRIPEL: privacy-preserving event log publishing including contextual information. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) BPM 2020. LNCS, vol. 12168, pp. 111–128. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58666-9_7
12. Daemen, J., Rijmen, V.: AES proposal: Rijndael (1999)
13. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_16
14. Burattin, A., Conti, M., Turato, D.: Toward an anonymous process mining. In: 2015 3rd International Conference on Future Internet of Things and Cloud, pp. 58–63. IEEE (2015)
15. Tillem, G., Erkin, Z., Lagendijk, R.L.: Mining encrypted software logs using alpha algorithm. In: SECRYPT, pp. 267–274 (2017)
16. Liu, C., Duan, H., Zeng, Q., Zhou, M., Faming, L., Cheng, J.: Towards comprehensive support for privacy preservation cross-organization business process mining. IEEE Trans. Serv. Comput. **12**(4), 639–653 (2016)
17. Elkoumy, G., et al.: Secure multi-party computation for inter-organizational process mining. In: Nurcan, S., Reinhartz-Berger, I., Soffer, P., Zdravkovic, J. (eds.) BPMDS/EMMSAD -2020. LNBIP, vol. 387, pp. 166–181. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49418-6_11
18. Costan, V., Devadas, S.: Intel SGX explained. IACR Cryptol. ePrint Arch. **2016**(86), 1–118 (2016)
19. Pinto, S., Santos, N.: Demystifying ARM TrustZone: a comprehensive survey. ACM Comput. Surv. (CSUR) **51**(6), 1–36 (2019)
20. Zhang, L., Bakshi, S., Zao, K.: Off-chain trusted computing. IEEE Internet Things Mag. **3**(2), 8–9 (2020)
21. Cheng, R., et al.: Ekiden: a platform for confidentiality-preserving, trustworthy, and performant smart contracts. In: 2019 IEEE European Symposium on Security and Privacy (EuroS&P), pp. 185–200. IEEE (2019)
22. Müller, M., Ostern, N., Rosemann, M.: Silver bullet for all trust issues? Blockchain-based trust patterns for collaborative business processes. In: Asatiani, A., et al. (eds.) BPM 2020. LNBIP, vol. 393, pp. 3–18. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58779-6_1
23. Buterin, V., et al.: Ethereum white paper. GitHub Repos. **1**, 22–23 (2013)

24. Sayyad, S.: Privacy preserving deep learning using secure multiparty computation. In: 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), pp. 139–142. IEEE (2020)
25. Zorarpacl, E., Ozel, S.A.: A hybrid approach of homomorphic encryption and differential privacy for privacy preserving classification. Int. J. Appl. Math. Electron. Comput. **8**(4), 138–147 (2020)
26. Hyperledger avalon. https://github.com/hyperledger/avalon. Accessed 30 Aug 2021

Quantifying the Re-identification Risk in Published Process Models

Karim Maatouk and Felix Mannhardt(✉)

Eindhoven University of Technology, Eindhoven, The Netherlands
f.mannhardt@tue.nl

Abstract. Event logs are the basis of process mining operations such as process discovery, conformance checking, and process optimization. Sensitive information may be obtained by adversaries when re-identifying individuals that relate to the traces of an event log. This re-identification risk is dependent on the assumed background information of an attacker. Multiple techniques have been proposed to quantify the re-identification risks for published event logs. However, in many scenarios there is no need to release the full event log, a discovered process model annotated with frequencies suffices. This raises the question on how to quantify the re-identification risk in published process models. We propose a method based on generating sample traces to quantify this risk for process trees annotated with frequencies. The method was applied on several real-life event logs and process trees discovered by Inductive Miner. Our results show that there can be still a significant re-identification risk when publishing a process tree; however, this risk is often lower than that for releasing the original event log.

Keywords: Process mining · Process discovery · Re-identification Risk

1 Introduction

Process mining is the science of understanding processes and improving them based on event data. Event data is mined for insights that can help industries in optimizing their processes, re-engineering them, and aiding their decision-making. Process discovery is one application of process mining allowing to understand the underlying processes by visualizing a process model of how the process was executed.

This means that event data availability is key to any process mining task; without event data, there is no process mining. However, the publishing of event data, in many cases, is subject to constraints due to privacy concerns; hence, limiting the availability of event data. Fields, such as healthcare make use of process mining techniques for optimizing their processes and consider patient information whose privacy concerns are of utmost importance.

So, privacy is an important topic in the process mining field given the growing collection and use of data which may originate from personal activities or process event logs that contain information on individuals. Maintaining privacy of individuals in process mining use cases is difficult since event data is sequential and often individual cases or events are related to sensitive information about individuals. An example of

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 382–394, 2022.
https://doi.org/10.1007/978-3-030-98581-3_28

such information would be the information about medical tests performed on a patient in a hospital. Existing work, therefore, has provided methods to quantify the risk of re-identification of individual information in a published event log [8,10]. This allows to assess the risk when releasing the original event log or to judge the effectiveness of a specific anonymization technique on the privacy of an event log.

The existing work on privacy-risk quantification only considers the various privacy related risks, e.g., the re-identification risk as in [10], for published event logs or closely derived representations such as directly-follows graphs [5] (Sect. 3). However, in many use cases the event log does not need to be public and could be only available to a process mining system that discovers process models providing an abstract representation of the source data. Still, there is a risk of re-identification of sensitive information based on such published process models that were mined from the event log. Such re-identification risk of discovered process models and how it differs from the re-identification risk of the source event log has not yet been investigated.

This work explores which privacy attacks are possible using the information in a published process model and aims to quantify the re-identification risk for a given published process model. Such quantification would enable new evaluation options for anonymization schemes and help to judge whether a certain process model can be released to a specific audience. The input to our method are frequency-annotated process models that can be converted to process trees [11] such as, e.g., discovered by the Inductive Miner [7]. We propose a randomized log replay technique to generate multiple possible event logs (scenarios) given the constraints of the process model and its frequencies. Based on these generated event log scenarios, we leverage the existing re-identification risk measures proposed by Rafaei et al. [8] (Sect. 4). The method was evaluated on several real-life event logs (Sect. 5) and the results were compared to the re-identification risk of the original logs.

2 Problem Statement

Process models, which are a graphical representation of the process, can be of different types such as Petri nets, Process trees, or Directly-Follows Graphs (DFGs). These process models can be discovered automatically using process discovery algorithms such as, e.g., Inductive miner [7]. A process discovery method takes an event log as input and returns a process model as a compact representation of the process behavior that was observed.

Events in an event logs contain in addition to case identifiers, which refer to the process instance in which the event occurred, other event information such as timestamp, activity, resource, and cost. For each case that consists of all events with the same case identifier in an event log, we can write its trace, i.e., the sequence of events ordered by timestamp, in a concise form representing the activities found in a case. For example, the trace for case 1 in the event log of Table 1 can be represented as

$$\langle \text{register request, examine thoroughly, check ticket, decide, reject request} \rangle.$$

A common output format of process discovery algorithms are Petri nets, which are a graphical representation of a given process. Since both event logs and process models contain information, there are risks of privacy attacks that seek to reveal sensitive

Table 1. Fragment of an event log about handling requests for compensation [2]

Case	Event	Timestamp	Activity	Resource	Cost
1	35654423	30-12-2010:11.02	register request	Pete	50
1	35654424	31-12-2010:10.06	examine thoroughly	Sue	400
1	35654425	05-01-2011:15.12	check ticket	Mike	100
1	35654426	06-01-2011:11.18	decide	Sara	200
1	35654427	07-01-2011:14.24	reject request	Pete	200
2	35654483	30-12-2010:11.32	register request	Mike	50
2	35654485	30-12-2010:12.12	check ticket	Mike	100
2	35654487	30-12-2010:14.16	examine casually	Pete	400
2	35654488	05-01-2011:11.22	decide	Sara	200
2	35654489	08-01-2011:12.05	pay compensation	Ellen	200

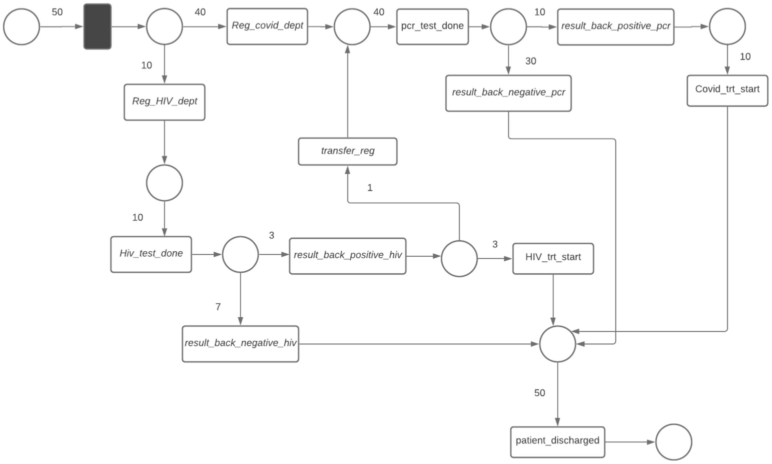


Fig. 1. Petri net with frequencies of the Medical Center COVID, HIV testing process.

information given the attacker has some background information about the individuals in an event log or process model. Petri nets can represent a variety of information that might unintentionally be revealed by the publisher of the Petri net to stakeholders or to the public. In other scenarios, a privacy attack might occur and an adversary might be able to disclose sensitive or classified information based on the published Petri net. Although the data might not be present explicitly in the Petri net, the attacker might be able to draw out sensitive information by using different techniques; especially when combined with other data obtained from a breach or other publicly available data about the individuals. For the purposes of illustration, we use the following example in Fig. 1 of a testing process for COVID and HIV in a medical facility. The process illustrates the process from the point of registration in the relevant department (COVID or HIV) up to the testing process, result (negative or positive), and the discharge of the patient.

The Petri net is frequency-annotated with the number of occurrences of the respective activities (transitions).

A re-identification attack occurs when an adversary attempts to reverse the anonymity of certain information that was masked by an operation to remove sensitive information. The adversary can use information that was acquired publicly or from a data breach. Attacks are most successful when adversaries correlate or match different datasets and information to mount an attack. Process models may contain sensitive information about individuals or processes. The adversary might attempt to use the information in the process model to re-identify individuals in their attack. In a dataset or an event log, an adversary will attempt to single out an individual's identity based on the uniqueness of a record's or event's identifiers in order to mount a re-identification attack. Singling out a record, the adversary can attempt a linkage attack using other datasets obtained by the adversary which can lead to the re-identification of individual information.

In the context of a frequency-annotated process model, a similar paradigm can be established to understand how an adversary can use the process model information to re-identify individuals. Singling out individuals in a process model can be done based on infrequent paths (runs) in the process model. An infrequent path in a process model allows an adversary to single out an activity of that process. To illustrate, in the Petri net shown in Fig. 1, we notice a single case in which a patient transferred from the HIV testing department to a COVID testing department. This information does not reveal the individual's identity; however, coupled with other information that the attacker might have, it can lead to a successful re-identification by an adversary. For instance, if the adversary has background information about patients registrations for COVID department and they detect that a unique individual transferred from HIV department, without undergoing registration directly for a PCR test, they can conclude that individual is Carla Sanders from the background information. They can also know from the process model that Carla Sanders has also undergone an HIV test as well as the result of the HIV test (positive). Therefore, the individual identity and HIV results are revealed in that attack. In Fig. 1, there is a single activity transfer_reg after ongoing a result_back_positive_hiv as indicated by the frequency on the activity(transition). Assume that also in the background information, Carla Sanders transferred registration to COVID department is recorded. Therefore, the adversary can identify Carla Sanders as the individual belonging to the trace containing transfer_reg and also can know the complete trace prior to the transfer, such as that she had undergone a positive HIV test.

By singling out the infrequent trace transfer_reg of frequency 1 of the Petri net in Fig. 1, the adversary mounts a linkage attack using certain background knowledge re-identify the individual as Carla Sanders who is the only one that has undergone transfer on that day April 2, 2020; and thus can identify that she has undergone a positive HIV test. This shows that a re-identification attack is also possible using the information available in a published Petri net.

3 Related Work

Privacy preserving process mining has increasingly gained interest in the process mining community. This comes with legislations and data protection regulations across the

EU becoming stricter, especially with the General Data Protection Regulation (GDPR) [1]. A main concern for privacy-preserving process mining approaches is how to ensure privacy when event logs may be published. Since the processes studied can be in fields dealing with sensitive data such as healthcare, financial institutions, and other critical fields, privacy is of utmost importance.

Quantifying the re-identification risk is closely related to research on privacy-preserving process mining. Knowing the risk can help evaluate the effectiveness of privacy-preserving methods in process mining. It can also act as a comparison measure before and after applying privacy models. Although re-identification attacks are hugely researched in different fields [3,4,6,9], there is only a couple of published research in the process mining community on quantification of disclosure risk in event logs and directly-follows graphs. Most related to our research are [5,8,10].

In [10], Nunez von Voigt et al. present a method to quantify the re-identification risk in event logs. The authors propose two measures to quantify the risk. Both measures are based on uniqueness in the event logs. The two measures are: uniqueness based on case attributes and uniqueness based on traces. In the first measure, the uniqueness of the case attributes in an event log is used to estimate the re-identification risk. The second measure considers the uniqueness of traces in an event log to account for event logs that do not have a lot of case attributes where the only information in the event log is the traces themselves. The work quantifies the risk in publicly available event logs and demonstrates how the re-identification risk can be very high for some of them and that almost every case can be re-identified in some scenarios. This sheds light on the need for adequate methods to quantify the re-identification risks and means to protect against them. In [8], Rafiei et al. introduces two measures for quantifying disclosure risk in published event logs to evaluate the effectiveness of privacy-preserving techniques. The two proposed measures are identity (case) disclosure and attribute(trace) disclosure. The first measure, case disclosure, uses uniqueness to measure how trace owners can be re-identified. The second measure, trace disclosure, measures how the sensitive attributes such as the complete trace of a case, can be disclosed. The method takes into account the background knowledge that the attacker might have about the event log when quantifying the risk. The method considers three types of background knowledge: set, multiset, and sequence. The set background knowledge is simply the set of activities in a process that the attacker might know are related to an individual. A multiset background knowledge provides additional knowledge about the number of occurrences of the process activities, while the sequence background knowledge provides additional information about the order in which the activities have occurred for the trace owner. The paper applies the method on two publicly available real-life event logs. In [5], Elkoumy et al. discuss the re-identification probability in DFGs, which are an output of process mining techniques. The work expresses the re-identification and the guessing advantage of an attacker by calculating the guessing probability given a DFG.

4 Approach

Existing re-identification risk quantification techniques calculate re-identification risk from event logs which constitute of traces. Our main idea is to calculate the re-identification risk from frequency-annotated process models by generating possible sets of traces corresponding to the original event log from the process model.

4.1 Approximating Re-identification Risk by Simulation

The generation of the exact traces corresponding to the original log from the frequency-annotated process model is not possible in all process models. This can be done accurately for process models that do not contain concurrent transitions and cycles as described in the challenges described earlier. This is due to the fact that when the process model contains concurrent transitions and cyclic behaviour, there can be multiple execution traces, of which we do not know which of them correspond to the trace in the original event log. The brute force approach to calculate the re-identification risk would consider the generation of all the combinations of transitions in a process model and their possible trace sets which could be very computationally expensive.

However, since most process models that describe business processes contain concurrent and cyclic behaviour, we adapt the solution to estimate the quantification risk using approximation techniques. Our proposed approach is to generate execution traces from the frequency-annotated process model without considering all the possible traces sets in a process model. This is because when the process model has many transitions and especially nested cyclic and concurrent ones, the number of possible traces sets increases substantially. Afterwards, we employ the existing measures for quantification of risk on the approximated possible traces of the process model.

The idea is to generate the execution traces in an ordered manner and not consider all the possible traces combinations. For this purpose we restrict ourselves to process models or Petri nets that can be represented as *process trees*. Then, we can traverse the process model in an ordered manner without randomizing the options to fire transitions, Therefore, for XOR transitions and concurrent transitions, we only consider a fixed order of firing of the activities, and not all the combinations possible. In the following, we assume as input a frequency annotated process tree.

4.2 Process Trees

Process models using graph-based notations can complicate process discovery from such event logs and result in unsound process models which complicates discovery. We use process trees or block-structured models that are sound by construction [2]. A process tree is defined as follows:

Definition 1. (Process Tree) [2]. *Let $A \subseteq \mathcal{A}$ be a finite set of activities with $\tau \notin A$. $\oplus = \{\rightarrow, \times, \wedge, \circlearrowleft\}$ is the set of process tree operators.*

- *If $a \in A \cup \{\tau\}$, then Q = a is a process tree,*
- *If $n \geq 1, Q1, Q2, ..., Qn$ are process trees, and $\oplus \in \{\rightarrow, \times, \wedge\}$, then $Q = \oplus(Q1, Q2, ...Qn)$ is a process tree, and*

- *If $n \geq 2$ and $Q1, Q2, ..., Qn$ are process trees, then $Q = (\circlearrowleft (Q1, Q2, \ldots, Qn)$ is a process tree.*

$\mathcal{L}_A$ is the set of all process trees over A.

The leaf nodes of a process tree represent process activities and the other nodes represent operators. Process trees have four operators: sequence operator, exclusive choice operator, parallel operator, and loop operator denoted as: $\{\rightarrow, \times, \wedge \ \circlearrowleft\}$ respectively. The four operators can also be abbreviated as seq, xor, and, xor loop which will be adopted in this work. The operator nodes define the order of execution of their children nodes. In the following, we define the semantics of the execution of a process tree by its operator type:

Definition 2 (Semantics of a Process Tree). *Let $\mathcal{P}$ be a process tree. Let $\mathcal{N} \in \mathcal{P}$ be any non-leaf node. Let $\mathcal{T}(\mathcal{N})$ be of range $\{\rightarrow, \times, \wedge \ \circlearrowleft\}$ be the type of Node $\mathcal{N}$ operator. Let $\mathcal{T}(\mathcal{N})$ have children nodes $\{a, b\}$ with a being the leftmost node, and b being the rightmost node. The execution of the children of the node $\mathcal{T}(\mathcal{N})$ is done as follows:*

- *if $\mathcal{T}(\mathcal{N}) = \rightarrow$, a is executed then b is executed. Trace is $\{a, b\}$*
- *if $\mathcal{T}(\mathcal{N}) = \times$, a is executed or b is executed. Trace is $\{a\}$ or $\{b\}$*
- *if $\mathcal{T}(\mathcal{N}) = \wedge$, a and b are executed, a or b may come first. Trace is $\{a, b\}$ or $\{b, a\}$.*
- *if $\mathcal{T}(\mathcal{N}) = \circlearrowleft$, a is executed, b can be executed any number of times 0...n. For each execution of b, a is executed again. Trace is $\{a\}$ or $\{a, b, a\}$ or $\{a, b, a, ..., a, b, a\}$.*

4.3 Frequency Constrained Traversal of the Process Tree

After having assigned all the nodes of the process tree with their respective firing frequencies, the next step is to traverse the process tree according to those frequencies. This will allow us to generate simulated traces that are similar in their frequencies of transitions to the inputted Petri net. The traversal of the process tree is done in a top-bottom approach on the tree while decrementing the counts of the nodes traversed until the frequencies are fully satisfied in the process tree. The traversals of the tree must satisfy the counts on the nodes of the tree. We define the execution order of the nodes in a process tree by our simulation approach as follows:

Definition 3 (Execution Order of Nodes in Process Tree by our Approach). *Let P be a process tree and $T(N)$ be the type of the tree operator at the root node of P which can be one of $\{SEQ, XOR, AND, LOOP\}$. Let N be a node of a process tree which can be equal to P or any subtree of P that is a non-leaf node. The execution order of the children of process tree P by our approach is as follows:*

1. *If $T(N) = SEQ$: execute leftmost child first followed by second leftmost and so on.*
2. *If $T(N) = AND$: execute all the children of the AND node in a fixed order from left to right.*
3. *If $T(N) = XOR$: execute the first child of the XOR node with remaining frequency > 0*
4. *If $T(N) = LOOP$: execute leftmost child, repetition is possible by executing rightmost child additionally then executing the leftmost child again. The overall number of repetitions is equal to the frequency of the rightmost child of the XOR LOOP node.*

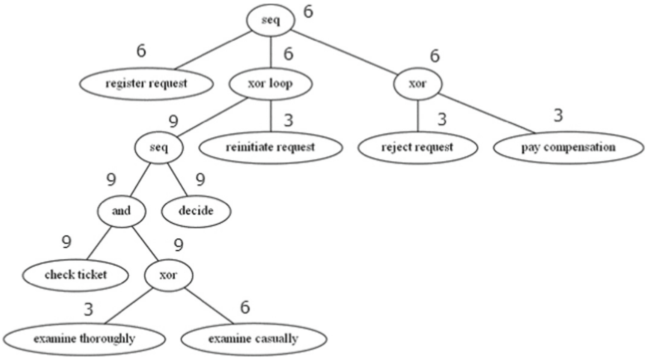


Fig. 2. Frequency Annotated Process Tree of Handling Request For Compensation Log

In Fig. 2, we show the frequency-annotated of the initial example log in Table 1 and its corresponding frequency-annotated process tree as decorated by our approach in the previous section. We give one example run of the tree traversal execution order in our approach according to Definition 3 for the process tree in Fig. 2. The execution of the process tree should start from the root node and is as follows:

1. root SEQ node is executed
2. children of SEQ node are executed: register request, XOR LOOP, and xor in order from left to right.
3. When XOR LOOP is executed, children of the loop are executed: SEQ, reinitiate request. For each execution of the right child of the loop (reinitiate request), the leftmost child (SEQ) of the XOR LOOP is executed again. That is, leftmost child(SEQ) is executed followed by the execution of AND node, decide node. Next, children of AND node (check ticket and XOR) are executed in left-to-right order. Next, examine thoroughly is executed given its remaining frequency > 0; otherwise, examine casually is executed.
4. XOR is executed, and then its child reject request is executed given its remaining frequency > 0; otherwise, pay compensation is executed.

The traversal of the process tree is done multiple times until the frequencies of the process tree are satisfied. While nodes are executed, the count of the executed nodes is decremented until the count reaches zero satisfying the frequencies observed - at which the execution is stopped. Therefore, we obtain from the process tree executions that are equivalent in their activity frequencies to the original event log which the Petri net was mined from. However, the individual traces may differ due to the higher abstraction level of the process model.

5 Evaluation

We evaluate our approach on real-life event logs to investigate feasibility and validity.

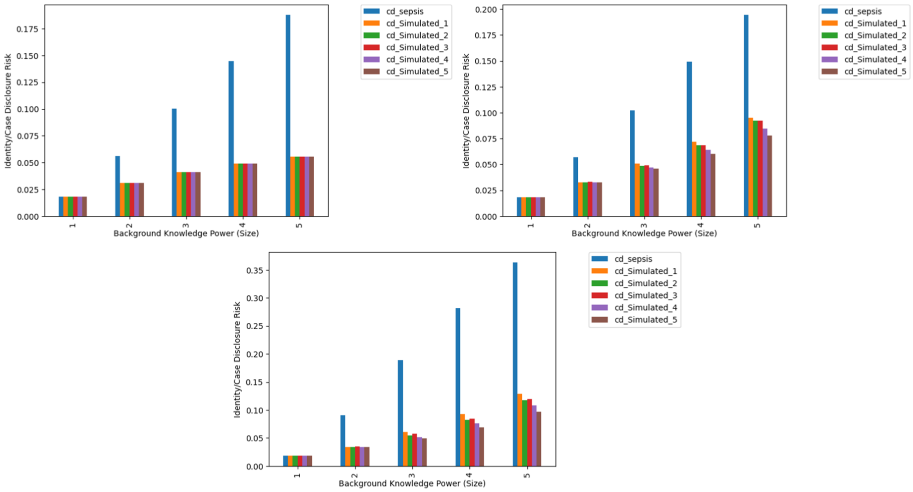


Fig. 3. Case disclosure results for the original log and our simulated event logs.

5.1 Experimental Setup

We performed experiments with multiple real-life event logs which are publicly available at the 4TU Centre for Research Data[1]. Here, we only report the results on the Sepsis Cases and the Road Traffic Fine Management (RTFM) logs since they were frequently used in the related work. For each log, we generate a frequency-annotated process model using Inductive Miner [7]. From the process model, we obtain five simulated event logs by applying our approach. We did not opt for more simulations since there is little variation between the results of the simulated event logs. Afterwards, we calculate the identity (case) disclosure and trace disclosure measures as mentioned in our approach in Sect. 4 and implemented in Rafiei et al. in [8] using the *p-privacy-qt* library published by their work. Then, we report the case disclosure and trace disclosure for both the original event log and the five simulated event logs generated from the mined process model of the original event log. Our approach is implemented in Python and can be found on GitHub[2].

5.2 Identity (Case) Disclosure Results

In Fig. 3, we demonstrate the identity (case) disclosure risk on the original Sepsis-cases log and the simulated logs generated by our approach. The identity (case) disclosure risk increases with increasing the background knowledge power size. The more background information available to the attacker, the more the risk of re-identifying individuals in the event log. The risk also increases with varying the background knowledge type from set to multiset and sequence respectively which is also explained by more background

[1] https://data.4tu.nl/.
[2] https://github.com/Karimmaatouk15/quantification_reidentification_risk_process_models/.

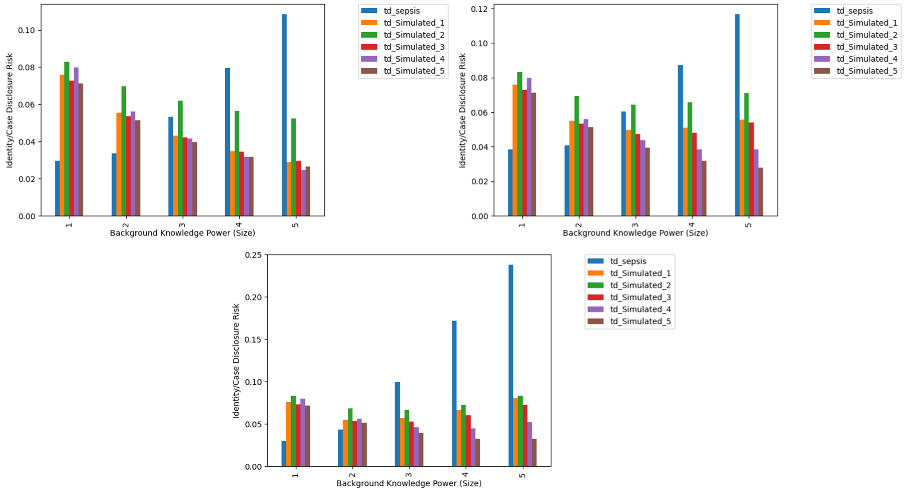


Fig. 4. Trace disclosure results for the original Sepsis log and our simulated event logs.

information available to the attacker about the activities. This increase is also noticeable in all other event logs that are studied in the experiments. As can be seen also in Fig. 3, the identity (case) disclosure risk in the 5 simulated event logs generated from the process model mined from the original Sepsis Cases event log is less than or equal to the identity (case) disclosure risk in the original event log with the gap between the original log and the simulated logs increasing with the increase in the background knowledge power size. The risk of the simulated logs from the process model is an estimator, in our approach, of the identity (case) disclosure risk of the process model mined from the original event log.

5.3 Trace Disclosure

We report the results of the experiments to quantify the trace disclosure risk. In Fig. 4, we notice that the trace disclosure risk increases for the original sepsis-cases event log but not for the simulated event logs by the increase in the background knowledge power size. However, for other event logs such as the Road Traffic Fine Management event log in Fig. 5, the trend is different, we notice that the trace disclosure risk decreases for the original RTFM log and is varying for the simulated event logs by the increase in the background knowledge. This indicates that the trace disclosure, indeed, does not follow the same trend as the identity (case) disclosure risk with the increase in the size of the background knowledge power. The trace disclosure, then, can be high even for weaker background knowledge power size, which was also found in [8].

5.4 Discussion

The distributions of the identity disclosure risks of the five simulated logs of the process model mined from the Sepsis Cases log are all below or equal to the risk of the original

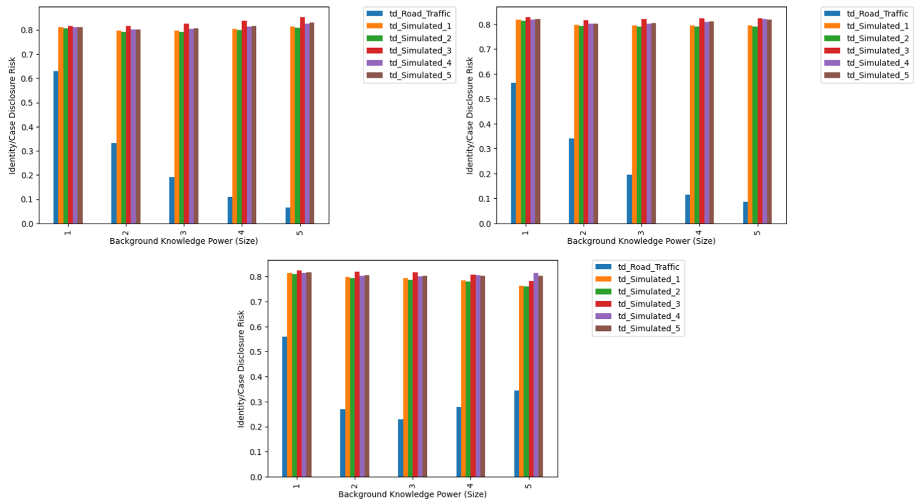


Fig. 5. Trace disclosure results for the original RTFM log and our simulated event logs.

event log. This result was also observed for all other tested event logs. Therefore, for all the studied event logs, on average, the identity disclosure risk is less in the simulated logs from process model than the original event log. This confirms our intuition that a process model abstracts certain behavior and, therefore, provides less information to an adversary. It also confirms that our simulation approach is feasible and our constrained simulation seems to return valid results. Clearly, the identity disclosure risk of the simulated event logs generated from the process model that was discovered from the original event log should (overall) not be higher than the risk of the original log. As we also noted, the identity disclosure risk and trace disclosure risk are varying not much between the simulated event logs. However, the results of the experimentation on the event logs does not guarantee that the hypotheses will be fulfilled for all event logs.

We already discussed from a theoretical perspective that a process model reveals less information than an event log. Thus, it is safe to say that process models are safer to publish generally than the event logs they were mined from. Moreover, the experiments also show that the identity disclosure risk is significantly less, on average, in process models than the original event logs they were mined from. In some cases, however, the re-identification risk can be equal or lower for some background knowledge sizes as shown in the results of our experiments. This may be an artefact of our simulation method but could also indicate that a process model can also have a similar risk in some cases similar to publishing a log.

Regarding the trace disclosure risk, the results are less clear. Depending on the log, our simulated event logs result in a higher risk compared to the risk of the original event log. Indeed, our method may generate less variants than contained in the original log and the disclosure risk appears to be higher. Thus, our method is not well suited to investigate the trace disclosure risk.

6 Conclusion

We discussed possible privacy attacks that an adversary can mount using a published process model in the form of a block-structured Petri net or process tree. We proposed a method to quantify the re-identification risk of such models that is based on a constrained simulation and leveraging existing work on quantifying re-identification risk. In our experiments, we validated the feasibility of our approach on several event logs and reported detailed results on the Sepsis Cases event log. Our conclusion is that our approach returns results that are in line with the intuition that when discovering a process model from an event log certain behavior is abstracted from and, thus, the re-identification is should, in general, be lower than that on the original event log. In future work, we want to evaluate this method in a more statistically rigorous manner, and work on more efficient approaches to approximate the re-identification risk directly from a non-block-structured Petri net without generating event logs.

References

1. General Data Protection Regulation (GDPR) - Official Legal Text
2. van der Aalst, W.: Process Mining - Data Science in Action
3. Dankar, F.K., El Emam, K., Neisa, A., Roffey, T.: Estimating the re-identification risk of clinical data sets. BMC Med. Inform. Decis. Making **12**, 66 (2012)
4. Domingo-Ferrer, J.: Disclosure risk. In: Liu, L., Özsu, M.T. (eds.) Encyclopedia of Database Systems, pp. 848–849. Springer, Boston (2009). https://doi.org/10.1007/978-0-387-39940-9_1506
5. Elkoumy, G., Pankova, A., Dumas, M.: Privacy-preserving directly-follows graphs: balancing risk and utility in process mining (2020). arXiv:2012.01119
6. Emam, K.E., Dankar, F.K., Vaillancourt, R., Roffey, T., Lysyk, M.: Evaluating the risk of re-identification of patients from hospital prescription records. Can. J. Hosp. Pharm. **62**(4), 307–319 (2009)
7. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from event logs - a constructive approach. In: Colom, J.-M., Desel, J. (eds.) PETRI NETS 2013. LNCS, vol. 7927, pp. 311–329. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38697-8_17
8. Rafiei, M., van der Aalst, W.M.P.: Towards quantifying privacy in process mining. In: Leemans, S., Leopold, H. (eds.) ICPM 2020. LNBIP, vol. 406, pp. 385–397. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-72693-5_29
9. Rocher, L., Hendrickx, J., Montjoye, Y.A.: Estimating the success of re-identifications in incomplete datasets using generative models. Nat. Commun. **10**, 1–9 (2019)
10. Nuñez von Voigt, S., et al.: Quantifying the re-identification risk of event logs for process mining. In: Dustdar, S., Yu, E., Salinesi, C., Rieu, D., Pant, V. (eds.) CAiSE 2020. LNCS, vol. 12127, pp. 252–267. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49435-3_16
11. van Zelst, S.J.: Translating workflow nets to process trees: an algorithmic approach. Algorithms **13**(11), 279 (2020)

Trustworthy Artificial Intelligence and Process Mining: Challenges and Opportunities

Andrew Pery[1], Majid Rafiei[2](✉), Michael Simon[3], and Wil M. P. van der Aalst[2]

[1] ABBYY, Ottawa, Canada
[2] Chair of Process and Data Science, RWTH Aachen University, Aachen, Germany
majid.rafiei@pads.rwth-aachen.de
[3] XPAN Law Partners, Boston, USA

Abstract. The premise of this paper is that compliance with Trustworthy AI governance best practices and regulatory frameworks is an inherently fragmented process spanning across diverse organizational units, external stakeholders, and systems of record, resulting in process uncertainties and in compliance gaps that may expose organizations to reputational and regulatory risks. Moreover, there are complexities associated with meeting the specific dimensions of Trustworthy AI best practices such as data governance, conformance testing, quality assurance of AI model behaviors, transparency, accountability, and confidentiality requirements. These processes involve multiple steps, hand-offs, re-works, and human-in-the-loop oversight. In this paper, we demonstrate that process mining can provide a useful framework for gaining fact-based visibility to AI compliance process execution, surfacing compliance bottlenecks, and providing for an automated approach to analyze, remediate and monitor uncertainty in AI regulatory compliance processes.

Keywords: AI ethics · Fairness · Artificial intelligence · Trust mining · Process mining

1 Introduction

AI-based technologies are becoming pervasive, impacting virtually every facet of our lives. While AI has a lot of promise, not all of its impacts are good. There is growing evidence that AI models can embed human and societal biases and deploy them at scale. As such, the ever-increasing growth of AI highlights the vital importance of balancing AI utility with the fairness of outcomes, thereby engender a culture of trustworthy AI. Fairness is the foundation for Trustworthy AI. Intuitively, fairness seems like a simple concept. However, it embodies consideration of a number of dimensions, such as trade-offs between algorithmic accuracy versus human values, demographic parity versus policy outcomes and power-focused questions such as who gets to decide what is fair.

J. Munoz-Gama and X. Lu (Eds.): ICPM 2021 Workshops, LNBIP 433, pp. 395–407, 2022.
https://doi.org/10.1007/978-3-030-98581-3_29

These are vexing challenges for AI developers, policy-makers and consumers alike. For AI developers, clarity of what constitutes AI fairness is a key consideration given the juxtaposition of ethical, legal, and reputational issues. For policy-makers and regulators, the challenge is how to promote innovation while protecting consumers from the harmful impacts of AI. For consumers of AI, its about trustworthiness, whether they can rely upon AI outputs to be accurate and transparent, with safeguards in place to protect them from adverse outcomes.

This paper explores the challenges and opportunities associated with fostering a culture of Trustworthy AI, with particular focus on: (1) The current state of Trustworthy AI, including a survey of key industry and standards organization initiatives with emphasis on the proposed EU Artificial Intelligence Act, (2) The relationship between Trustworthy AI and Responsible Data Science (RDS), and (3) Contribution of trust aware process mining to facilitate a data-driven analytical framework to surface uncertainties, variabilities, and vulnerabilities in Trustworthy AI compliance processes.

The remainder of the paper is organized as follows. In Sect. 2, we define the contours of Trustworthy AI principles. In Sect. 3, we explore the proposed EU Artificial Intelligence Act (AIA) that intends to operationalize and implement rigorous risk-based prescriptive processes for ensuring a culture of Trustworthy AI. In Sect. 4, we map the relationship between RDS and Trustworthy AI, including a discussion of challenges associated with contextualizing AI fairness as a foundation for Trustworthy AI. In Sect. 5, we discuss the applications and benefits of process mining as an important tool to enable organizations to make data-driven decisions relating to the obligations and conformance requirements inherent in the proposed EU AI regulation.

2 Trustworthy AI

Surveys reveal an undercurrent of pervasive distrust of AI systems. Cathy O'Neil, a leading advocate for AI algorithmic fairness, highlighted three main reasons behind consumer distrust of AI: *opacity*, *scale*, and *damage* [12]. Fairness is the foundation for trustworthy AI. It is the connective tissue that binds together the principles of ethical use, interpretability, transparency, accountability, and confidentiality that engenders trust and promotes the use of AI for social good. Trustworthy AI is a governance framework designed to mitigate potential adverse impacts on consumers as AI is poised to profoundly and indelibly change our lives. As mentioned in [17], Trustworthy AI is changing the dynamic between user and system into a relationship.

2.1 Achieving Trust in AI

Trustworthy AI starts with human agency and autonomy. Trust in AI systems is enhanced when there is a human-in-the-loop who monitors the overall performance of AI systems and when circumstances dictate, remediates potential adverse outcomes. Trust in AI is strengthened by giving users the ability to make informed decisions about the impact of AI on their personal and economic well-being.

AI is perceived by consumers to be *a black box.* Data inputs to the AI systems, their learning models, and how they arrive at decisions are neither visible, nor understood by consumers. Furthermore, many AI developers defensively protect their algorithms as proprietary and a competitive differentiator. *Interpretability* and *explainability* of AI are two important elements that strengthen trust in AI. Interpretability of AI provides insight into the cause and effect between inputs and outputs of an AI system and how AI predicts outcomes. Explainability of AI goes one step further by providing users with not only insight into how AI models work but also traceability of AI decisions and documentation relating to the process of data gathering, labeling, and methods used for training AI algorithms.

Consumers have limited recourse to hold AI developers accountable for the adverse impacts of AI systems. While there is sectoral legislation, e.g., Sect. 5 of the FTC (Federal Trade Commission) Act[1], available for consumers to remedy disparate treatment attributable to AI systems it is an onerous process to prevail. Moreover, for the disparate impact, the burden of proof requires statistical analysis that a protected class is treated differently from others, which is hardly something that would be accessible to average consumers. For these reasons, accountability, including redress mechanisms in the event of demonstrated harmful impact need to be addressed to achieve trust in AI.

2.2 The Emergence of Trustworthy AI Principles

We can see efforts being made, to varying degrees, that recognize and deal with issues relating to trust in AI by the data sciences community (see Sect. 4), standards organizations, e.g., IEEE [16], NIST (National Institute of Standards and Technology) [13], and by public sector organizations.

In 2019, OECD member countries adopted OECD Council Recommendation on Artificial Intelligence[2] consisting of five principles of human centered values of fairness of AI, inclusive investments in AI, transparency, accountability, and robustness of AI systems. The OECD recommendations were subsequently endorsed by the G20 with particular reference to the view that the "digital society must be built on trust among all stakeholders including governments, civil society, international organizations, academics, and businesses through sharing common values and principles including equality, justice, transparency, and accountability taking into account the global economy and interoperability".

While Trustworthy AI principles serve as a helpful framework, they are just that. Adherence to Trustworthy AI is fragmented at best and they lack effective enforcement mechanisms to safeguard against potentially harmful impacts. For this reason, the momentum has shifted towards the regulation of AI: "The calls for modest regulation that lets industry take the lead are part of a failed

[1] https://www.federalreserve.gov/boarddocs/supmanual/cch/ftca.pdf.

[2] https://legalinstruments.oecd.org/en/instruments/OECD-LEGAL-0449.

regulatory philosophy, one that saw its natural experiment over the past several decades come up lacking. AI is too important and too promising to be governed in a hands-off fashion, waiting for problems to develop and then trying to fix them after the fact".[3]

3 The Proposed EU Regulation of AI

On April 20, 2021 the European Commission released the proposal for the regulation of artificial intelligence[4], the ambition of which is to balance the socio-economic benefits of AI and new risks or negative consequences for individuals or society. The proposed Artificial Intelligence Act (AIA) takes a risk-based approach to regulate AI by fostering an "ecosystem of trust that should give citizens the confidence to take up AI applications and give companies and public organisations the legal certainty to innovate using AI". In the following, we demonstrate five governing principles for trustworthy AI proposed by AIA.

3.1 Scope of the Proposed Regulation

The proposed AIA applies to all providers, i.e., natural or legal persons, public authorities, agencies, or any other body that develops an AI system, that places or makes available on the market or puts into service AI systems or services in the EU (cf. Article 3). The AIA also assigns responsibility to users, importers, distributors, and operators who make use of or make substantial modifications to the functionality and performance of AI systems (cf. Article 26–29). The geographic scope for the AIA will operate irrespective of whether such providers are established in the EU or a third country, and so will cover where the system users are in the EU or the output of the systems is used in the EU (cf. Article 2). AI systems under the regulation encompass a wide range of methods and algorithms including supervised, unsupervised, and reinforcement machine learning for a given set of human-defined objectives that generate outputs such as content, predictions, recommendations, or decisions influencing the environments they interact with (cf. Article 3).

3.2 Risk-Based Approach

The foundation of the AIA is a risk-based approach that classifies AI systems into three categories based on a combination of factors that include the intended purpose, the number of impacted persons, and the potential risk of harms (cf. Article 5–7):

- Prohibited AI: Systems that use subliminal techniques that cause physiological or psychological harm, exploit vulnerable groups, effectuate social scoring by public authorities that may result in discrimination or unfavorable treatment, and remote biometric systems used by law enforcement in public spaces (subject to well-defined exceptions) (cf. Article 5).

[3] https://www.brookings.edu/research/ai-needs-more-regulation-not-less/.
[4] https://ec.europa.eu/commission/presscorner/detail/en/ip_21_1682.

- High Risk: Annex III provides a list of systems that are used in critical infrastructures, educational or vocational training, human resources, essential private and public services, law enforcement, migration, asylum and border control management, and administration of justice and democratic processes (cf. Article 7).
- Low Risk: While not explicitly named (we use the term *low risk* of our own choosing), by default, all systems not categorized as *prohibited* or *high-risk.* Providers of such systems are encouraged to institute responsible use of AI best practices on a voluntary basis (cf. Article 69).

3.3 Promote Fair and Trustworthy AI Best Practices

The AIA sets forth a comprehensive legislative mandate to ensure fairness in the application of AI systems that safeguards fundamental human values and promotes socio-economic rights. Some of these mandates are as follows: obligation on providers to implement appropriate risk management measures throughout the entire lifecycle of AI systems (cf. Article 9), rigorous data governance processes (cf. Article 10), technical documentation, and record-keeping processes to enable monitoring of compliance (cf. Article 11–12), transparency that enables full interpretation of outputs (cf. Article 13), and Human-in-the-loop oversight (cf. Article 14).

3.4 Transparency and Accountability

According to the AIA, providers of AI systems will be required to implement a range of processes to ensure full transparency into and accountability for AI systems (cf. Article 19–23) such as (1) conformity assessment and certification processes, (2) auditability, including accessible event logs, and (3) Explainability, potentially to coordinate with the human-in-the-loop for adjudication and remediation.

3.5 Enforcement

The AIA incorporates an onerous enforcement mechanism that even surpasses the fines under the GDPR (cf. Article 71). Some examples are as follows: up to €10m or 2% of the total worldwide annual turnover for the supply of incorrect, incomplete or misleading information to the authorities, up to €20m or 4% of the total worldwide annual turnover for non-compliance with any other AIA requirement or obligation, and up to €30m or 6% of the total worldwide annual turnover for violations of prohibited practices.

While the proposed AIA is far from ratification and still subject to vigorous debate within the EU Parliament and Council, the momentum towards its adoption is inevitable. Like the GDPR, the AIA will serve as a model for other jurisdictions that will seek to finally exert control over what has been the unregulated, hyperbolic growth of AI across the globe.

4 Responsible Data Science and Trustworthy AI

Responsible Data Science (RDS) is a discipline that is influential in shaping Trustworthy AI best practices. RDS refers to the collection of techniques and approaches trying to reap the benefits of data science and big data while ensuring *fairness*, *accuracy*, *confidentiality* and *transparency* [2]. To minimize adverse AI outcomes of AI the role of RDS is to: (1) Avoid unfair conclusions even if they are true, i.e., the fairness principle, (2) Answer questions with a guaranteed level of accuracy, i.e., the accuracy principle, (3) Answer questions without revealing secrets, i.e., the confidentiality principle, and (4) Clarify Answers such that they become indisputable, i.e., the transparency principle.

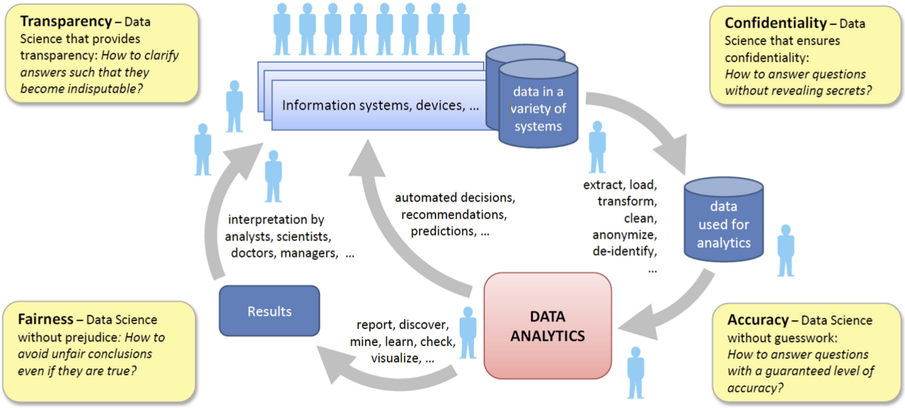

Transparency – Data Science that provides transparency: How to clarify answers such that they become indisputable?
Information systems, devices, ...
data in a variety of systems
Confidentiality – Data Science that ensures confidentiality: How to answer questions without revealing secrets?
extract, load, transform, clean, anonymize, de-identify, ...
data used for analytics
automated decisions, recommendations, predictions, ...
interpretation by analysts, scientists, doctors, managers, ...
Results
report, discover, mine, learn, check, visualize, ...
DATA ANALYTICS
Fairness – Data Science without prejudice: How to avoid unfair conclusions even if they are true?
Accuracy – Data Science without guesswork: How to answer questions with a guaranteed level of accuracy?

Fig. 1. The data science pipeline facing the four FACT challenges [2].

RDS applies a methodology throughout the entire life cycle of information to support trustworthy AI best practices by applying these four principles of fairness, accuracy, confidentiality, and transparency to the *data science pipeline* resulting in rigorous data governance as illustrated in Fig. 1.

RDS delivers a robust framework for the ethical design of AI systems that addresses the following key areas: (1) Unbiased outcomes through the application of appropriate fairness constraints to the training data, (2) Algorithmic outcomes interpreted in a manner that is meaningful to end users, (3) Resilience in how AI systems deliver accurate results and respond to change in inputs, (4) Accountability for the system's outcomes, and (5) Safeguarding the confidentiality of training data through privacy enhancing measures. However, providing each aspect of RDS has its own challenges from contextualizing the aspect to implementing it in data science and AI systems. In [6], the authors describe the challenges regarding the *confidentiality* aspect for *process mining* which combines process and data science. In the following, we provide the challenges regarding the *fairness* aspect.

4.1 Contextualizing Fairness in AI Systems: Challenges

The idea of *fairness* is somewhat amorphous. At its highest level of abstraction, fairness is a normative concept that comes from our conscience. Dator defines a fair system as follows: "What is fairness then? We all have desires and we want people to treat us according to those desires. We also know that people around us have similar desires and want to be treated accordingly. Fairness is closely related to fair play so it seems logical to conclude that a fair system is a system where everybody is treated in a similar way" [4]. There are a number of challenges associated with contextualizing and applying such a high level of abstraction to a more concrete algorithmic AI fairness framework.

First, fairness may be influenced by cultural, sociological, economic, and legal considerations. What may be considered as fair in one culture may be perceived as unfair in another. Unequal distribution of opportunity may require the application of distributive fairness that levels the playing field. For example, in the context of credit applications, there ought to be an equal probability of loan eligibility by ensuring that AI algorithmic outcomes do not discriminate against members of protected groups [3]. There are other instances where the application of corrective fairness may be necessary, for example, to remedy adverse impacts in the administration of justice, housing, education, and employment.

Second, equality does not necessarily result in the fairness of outcomes. While under Human Rights legislations disparate treatment on the basis of race, gender, nationality, disability, and sexual orientation is prohibited there may still be instances of adverse outcomes, based on other facially-neutral variables that cause a disparate impact, i.e., unintentional discrimination [5]. Consider Amazon's free same day delivery service based on an AI algorithm that included attributes, such as distance to the nearest fulfillment center, local demand in designated zip code areas, and frequency distribution of prime members to determine profitable locations for free Same-Day Delivery. The algorithm was found to be biased against minorities even though race was deemed not to be a factor in the determination of same day delivery, and minority residents in the selected zip codes were *about half as likely* to be eligible as white residents.[5]

The third challenge is balancing algorithmic fairness with fairness outcomes [10]. In this context, fairness encompasses policy and legal considerations, and leads us to ask: *what ought to be fair?* For example, in the context of hiring practices, what ought to be a fair percentage of women in management positions that AI algorithms should incorporate as thresholds to promote gender parity?

The fourth challenge relates to trade-off in balancing demographic parity with the utility of outcomes. For example, if AI algorithms remove disparate impact in the incarceration of minorities, how would that impact broader policy considerations such as the administration of justice?

Finally, fairness implicates issues of power. Before we can decide what is fair, we need to decide who gets to decide that. The conundrum we must confront is that the minority groups who are so typically the victims of algorithmic bias

[5] https://eu.usatoday.com/.

are rarely given a seat at the table when it is time to define what is fair. The unfortunate result is that far too often, the definition of fairness is simply what those already in power need it to be to maintain that power.

4.2 Implementing Fairness: Challenges for Data Scientists

Fairness constraints need to be considered in the context of specific use cases and for desired outcomes. Bias may be introduced at various levels within an AI system. Training data may introduce proxies that discriminate. Historical bias may unconsciously result in adverse outcomes, for example through word embeddings [4]. Representation bias through under or, over representation of training data may produce disparate impacts. The algorithms may not sufficiently adjust for fairness constraints. Inadequate testing for disparate treatment and impact may have adverse consequences for protected groups. While some argue that AI algorithms in fact minimize bias there is compelling evidence that they can and often amplify biases. Examples span facial recognition, criminal justice, hiring practices, and loan approvals [9].

Regardless of any contextualization, any definition, and any implementation approach of the fairness which is the cornerstone for Trustworthy AI, what is essential is to gain visibility to and remediate potential gaps in Trustworthy AI compliance processes. In the next section, we demonstrate how process mining could play a role in fulfilling such requirements.

5 Process Mining for Promoting Trustworthy AI

Compliance with the proposed EU AIA requires an understanding of process execution and interactions between multiple internal and external stakeholders, risk assessment of diverse systems of record that incorporate AI systems, and cooperation with various regulatory bodies and standards organizations.

The proposed AI regulation operationalizes and codifies trustworthy AI principles with prescribed mandates to institute *appropriate data governance and management practices.* The governance mechanism is complex and requires human and systems-based interactions between diverse internal and external stakeholders and EU and national regulators. Monitoring conformance with AIA is delegated to national supervisory authorities, they are empowered to order companies to take corrective actions, access all information, documentation, and data required to enforce compliance with the proposed regulation.

Given the complexity and variability of interactions implicit in achieving compliance with the proposed regulation it is our contention that *process mining* can be a valuable tool to help organizations gain visibility to various dimensions of prescribed process flows stipulated by the regulation, accelerate the analysis of how information flows, surface process bottlenecks, visualize interactions generated by event logs from disparate systems of record that may reveal areas of compliance and reputational risks. Process mining bridges the gap between data science and process science using event data captured from different types of information systems [1]. It is a data-driven approach that enables organizations to gain insight into interactions between people, systems, and organizations based on "as-is" visualization of process execution.

There are many techniques and activities in the context of process mining. However, the three main types of activities in process mining are *process discovery*, *conformance checking*, and *enhancement*. Process discovery techniques take an event log and discover a process model without using any other information. Conformance checking techniques take a process model and an event log of the same process to check whether reality, as recorded in the event log, conforms to the model and vice versa. Enhancement techniques are used to extend or improve a given process model using the information about the process recorded in some event logs [1].

Process Mining can facilitate compliance with AIA by many functionalities such as: (1) Surfacing AI regulatory compliance process gaps and uncertainties, (2) Capturing user interactions performing compliance tasks, (3) Comparing process execution variations, (4) Highlighting compliance task outliers and errors, (5) Identifying potential root causes for improper execution, (6) Real-time monitoring of processes to ensure conformance to prescribed process execution paths, and (7) Triggering alerts in the event of non-compliant process tasks or changes in conditions. Furthermore, the AIA proposed regulation is inherently collaborative in nature wherein process execution spans across different organizations.

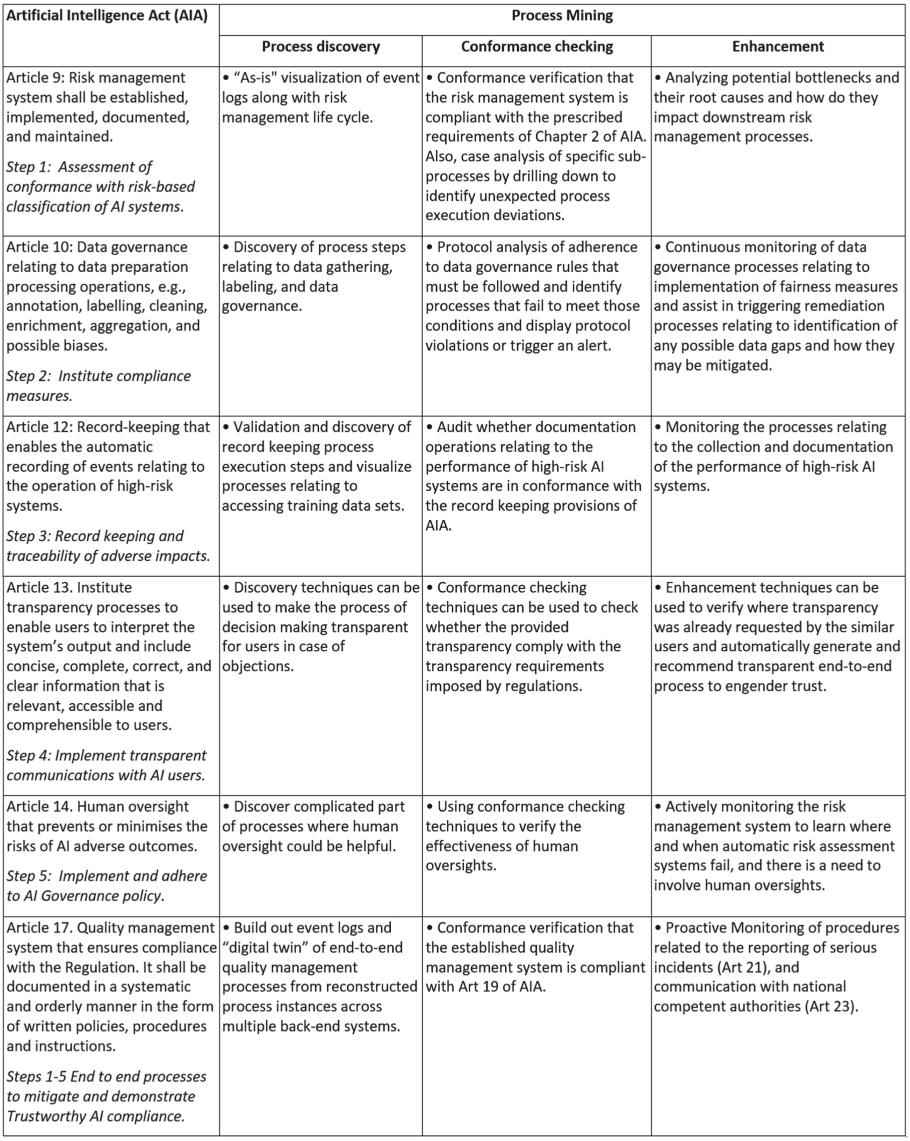
As discussed in [11], in collaborative processes where different organizations execute different parts of a shared process, the internal activities carried out by each organization are beyond the control of the other collaborators resulting in uncertainty regarding process execution. Whenever there is uncertainty in a process, there is a need for trust. Hence, collaborative business processes are especially trust-intensive. In such trust-intensive environments, process mining can be used to clarify the flow of activity execution among several organizations.

Compliance with AIA constitutes a number of interdependent steps. Performing these steps may involve variabilities in process execution paths and hand off between different stakeholders and prescribed conformance obligations to meet Articles 16–23 and Annex VII of the AIA:

- Step 1: R&D teams develop and bring to market AI systems in accordance with the risk classification system defined by the proposed regulation. If it is a high-risk AI system then a priori conformance assessment must be undertaken and a declaration of conformity must be submitted to the appropriate National Supervisory Authority. Then the AI system may be placed on the market.
- Step 2: Legal and Compliance teams must institute compliance measures in accordance with Sect. 2 of the proposed regulation that ensures adherence to data governance, accountability, transparency, accuracy, robustness, and cybersecurity provisions.
- Step 3: Data Science teams must undertake continuous monitoring of AI systems, collect data on the system's operation and take corrective action if

needed. The post-market monitoring system must actively and systematically collect, document, and analyze relevant data provided by users.

- Step 4: Customer-facing functions such as Sales, Marketing, and Support, are responsible for providing clarity and certainty as to the expected AI system inputs and outputs in a way that users are informed that they are interacting with an AI system, augmented with human oversight who monitor their oper-



Artificial Intelligence Act (AIA)	Process Mining		
	Process discovery	Conformance checking	Enhancement
Article 9: Risk management system shall be established, implemented, documented, and maintained. *Step 1: Assessment of conformance with risk-based classification of AI systems.*	• "As-is" visualization of event logs along with risk management life cycle.	• Conformance verification that the risk management system is compliant with the prescribed requirements of Chapter 2 of AIA. Also, case analysis of specific sub-processes by drilling down to identify unexpected process execution deviations.	• Analyzing potential bottlenecks and their root causes and how do they impact downstream risk management processes.
Article 10: Data governance relating to data preparation processing operations, e.g., annotation, labelling, cleaning, enrichment, aggregation, and possible biases. *Step 2: Institute compliance measures.*	• Discovery of process steps relating to data gathering, labeling, and data governance.	• Protocol analysis of adherence to data governance rules that must be followed and identify processes that fail to meet those conditions and display protocol violations or trigger an alert.	• Continuous monitoring of data governance processes relating to implementation of fairness measures and assist in triggering remediation processes relating to identification of any possible data gaps and how they may be mitigated.
Article 12: Record-keeping that enables the automatic recording of events relating to the operation of high-risk systems. *Step 3: Record keeping and traceability of adverse impacts.*	• Validation and discovery of record keeping process execution steps and visualize processes relating to accessing training data sets.	• Audit whether documentation operations relating to the performance of high-risk AI systems are in conformance with the record keeping provisions of AIA.	• Monitoring the processes relating to the collection and documentation of the performance of high-risk AI systems.
Article 13. Institute transparency processes to enable users to interpret the system's output and include concise, complete, correct, and clear information that is relevant, accessible and comprehensible to users. *Step 4: Implement transparent communications with AI users.*	• Discovery techniques can be used to make the process of decision making transparent for users in case of objections.	• Conformance checking techniques can be used to check whether the provided transparency comply with the transparency requirements imposed by regulations.	• Enhancement techniques can be used to verify where transparency was already requested by the similar users and automatically generate and recommend transparent end-to-end process to engender trust.
Article 14. Human oversight that prevents or minimises the risks of AI adverse outcomes. *Step 5: Implement and adhere to AI Governance policy.*	• Discover complicated part of processes where human oversight could be helpful.	• Using conformance checking techniques to verify the effectiveness of human oversights.	• Actively monitoring the risk management system to learn where and when automatic risk assessment systems fail, and there is a need to involve human oversights.
Article 17. Quality management system that ensures compliance with the Regulation. It shall be documented in a systematic and orderly manner in the form of written policies, procedures and instructions. *Steps 1-5 End to end processes to mitigate and demonstrate Trustworthy AI compliance.*	• Build out event logs and "digital twin" of end-to-end quality management processes from reconstructed process instances across multiple back-end systems.	• Conformance verification that the established quality management system is compliant with Art 19 of AIA.	• Proactive Monitoring of procedures related to the reporting of serious incidents (Art 21), and communication with national competent authorities (Art 23).

Fig. 2. Process mining cadence to meet AIA prescriptive compliance obligations.

ation and be able to decide, to override or reverse the output of the high-risk AI system.
- Step 5: Implementation of a Quality Management System with auditable and traceable documentation relating to the techniques, procedures for the design, of the high-risk AI systems, including procedures for data management, data analysis, data labeling, data storage, data aggregation, data retention and report serious incidents that may result in adverse outcomes.

Figure 2 further maps the compliance steps, the obligation provisions of the AIA, and process mining functionality to support Trustworthy AI. The figure illustrates how process mining techniques can facilitate AIA obligations. The FACT challenges of RDS are also taken into consideration in process mining as a subdiscipline called Responsible Process Mining (RPM) which is recently receiving increasing attention [7,8,14,15].

6 Conclusion

Trustworthy AI engenders a climate of trust essential for achieving sustainable competitive advantages in an intensely competitive environment where the application of AI is a disruptive force. The proposed EU regulation of AI is a comprehensive prescriptive measure which imposes onerous obligations, redress mechanisms on AI developers and businesses deploying AI systems. To mitigate compliance, reputational, and business risks process mining is poised to provide a data-driven approach to discover how existing Trustworthy AI compliance processes work, surface and remediate process bottlenecks, visualize different pathways of process execution and identify and remediate variations from prescribed protocols. Process mining can be a useful toolbox for ensuring that certain AI systems are designed and developed in accordance with common necessary requirements before they are put on the market and operationalized through harmonized technical standards.

Acknowledgments. Funded under the Excellence Strategy of the Federal Government and the Länder. We also thank the Alexander von Humboldt Stiftung for supporting our research.

References

1. van der Aalst, W.M.P.: Process Mining - Data Science in Action, 2nd edn. Springer, Heidelberg (2016)
2. van der Aalst, W.M.P.: Responsible data science: using event data in a people friendly manner. In: Hammoudi, S., Maciaszek, L.A., Missikoff, M.M., Camp, O., Cordeiro, J. (eds.) ICEIS 2016. LNBIP, vol. 291, pp. 3–28. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-62386-3_1
3. Binns, R.: On the apparent conflict between individual and group fairness. In: Proceedings of the 2020 conference on fairness, accountability, and transparency, pp. 514–524 (2020)

4. Dator, J.: What Is Fairness? University of Hawaii Press, pp. 19–34 (2006). https://doi.org/10.1515/9780824841966-004
5. Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.: Fairness through awareness. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, pp. 214–226. ITCS 2012, Association for Computing Machinery, New York, NY, USA (2012). https://doi.org/10.1145/2090236.2090255
6. Elkoumy, G., et al.: Privacy and confidentiality in process mining - threats and research challenges. CoRR abs/2106.00388 (2021). https://arxiv.org/abs/2106.00388
7. Elkoumy, G., Pankova, A., Dumas, M.: Mine me but don't single me out: Differentially private event logs for process mining. CoRR abs/2103.11739 (2021). https://arxiv.org/abs/2103.11739
8. Fahrenkrog-Petersen, S.A., van der Aa, H., Weidlich, M.: PRIPEL: privacy-preserving event log publishing including contextual information. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) BPM 2020. LNCS, vol. 12168, pp. 111–128. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58666-9_7
9. Grother, P., Ngan, M., Hanaoka, K.: Face recognition vendor test part 3: Demographic effects (2019)
10. Kleinberg, J., Ludwig, J., Mullainathan, S., Rambachan, A.: Algorithmic fairness. In: AEA papers and proceedings, vol. 108, pp. 22–27 (2018)
11. Müller, M., Ostern, N., Koljada, D., Grunert, K., Rosemann, M., Küpper, A.: Trust mining: analyzing trust in collaborative business processes. IEEE Access **9**, 65044–65065 (2021). https://doi.org/10.1109/ACCESS.2021.3075568
12. O'neil, C.: Weapons of math destruction: how big data increases inequality and threatens democracy. Crown (2016)
13. Phillips, P., Hahn, A., Fontana, P., Broniatowski, D., Przybocki, M.: Four principles of explainable artificial intelligence (2020)
14. Rafiei, M., van der Aalst, W.M.P.: Group-based privacy preservation techniques for process mining. Data Knowl. Eng. **134**, 101908 (2021). https://doi.org/10.1016/j.datak.2021.101908
15. Rafiei, M., van der Aalst, W.M.P.: Privacy-preserving continuous event data publishing. CoRR abs/2105.11991 (2021). https://arxiv.org/abs/2105.11991
16. Shahriari, K., Shahriari, M.: IEEE standard review - ethically aligned design: a vision for prioritizing human wellbeing with artificial intelligence and autonomous systems. In: 2017 IEEE Canada International Humanitarian Technology Conference (IHTC), pp. 197–201 (2017). https://doi.org/10.1109/IHTC.2017.8058187
17. Stanton, B., Jensen, T.: Trust and artificial intelligence (2021-03-02 05:03:00 2021). https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=931087

Author Index