

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

7,000

Open access books available

186,000

International authors and editors

200M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.

For more information visit www.intechopen.com



Chapter

Deep Multiagent Reinforcement Learning Methods Addressing the Scalability Challenge

Theocharis Kravaris and George A. Vouros

Abstract

Motivated to solve complex demand-capacity imbalance problems in air traffic management at the pre-tactical stage of operations, with thousands of agents (flights) daily, even in a restricted airspace, in this paper, we review deep multiagent reinforcement learning methods under the prism of their ability to scale toward solving problems with large populations of heterogeneous agents, where agents have to unavoidably decide on their joint policy, without communication constraints.

Keywords: deep reinforcement learning, multiagent systems, scalability

1. Introduction

Scalability in training large numbers of deep reinforcement learning agents, which must decide on actions jointly, is a major issue that becomes apparent in many real-life problems. This issue is related to numerous aspects of deep multiagent reinforcement learning (DMARL), such as assignment of credits to the learners for their choices, assumptions regarding homogeneity or interchangeability of the agents, society structure due to interaction of agents' decisions, agents' communication requirements, abilities, and constraints.

In this chapter, we provide a review of deep multiagent reinforcement learning (DMARL) methods, examining their ability to scale up to large agent populations. "Large" here could mean anything from hundreds up to several thousands of agents.

We are motivated to solve complex demand-capacity balancing problems in air traffic management (congestion problems regarding air sectors), where we may have 6000 flights daily, even in a restricted airspace (e.g., the Spanish airspace). Specifically, in the air-traffic management (ATM) domain, *demand and capacity balancing* problems (DCBs) are a kind of congestion problems that arise naturally whenever demand of airspace use exceeds capacity, resulting to "hotspots." Hotspots are resolved via capacity management or flow management solutions, including regulations that generate delays and reroutings to flights, causing unforeseen effects for the entire system, and increasing uncertainty regarding the scheduling of (ground and airspace) operations. For instance, flight delays cause the introduction/increase of time buffers in operations' schedules and may accumulate demand for resources within specific

periods. These are translated into costs and negative effects on airlines' reliability, customers' satisfaction, and environmental footprint. Representing flights by self-interested, heterogeneous agents (each with its own possible regulations, preferences, and constraints), the automation of problems' resolution requires agents to jointly decide on their policies regarding their own regulations [1]. Driven by the resolution of problems at the pre-tactical phase of operations (i.e., from some hours to few days before the actual flights), there are no communication or observation constraints for agents: In any case, agents need to coordinate with any agent with whom they participate to any specific congestion problem, given also that these problems may emerge dynamically in space and time.

In such large-scale, complex multiagent settings, we need to consider quality of solutions (as regulations incur additional costs to operations) and DMARL methods' training scalability. Factors that affect training scalability are as follows:

- The *training paradigm* adopted: Agents may train independent, centralized, or shared models.
- The *types of models* learnt following any paradigm: A policy model, a value model, or both types of models may be fit, following any of the training paradigms.
- *Assumptions regarding agents homogeneity*: Agents may be considered to be heterogeneous, homogeneous (i.e., following the same policy, which may be however differentiated due to contextual features), or even interchangeable.
- *Effectiveness of communication*: Communication between agents may be explicit (i.e., passing information by any means) or implicit (i.e., via the environment, or via sharing models' parameters), and orthogonal, either be performed in a global (e.g., broadcasting messages to all agents) or local scale (i.e., in a "neighborhood"). Here, optimality of communication is something that agents could learn via elaborated (e.g., attention) mechanisms.

In addition to the above factors, decomposing rewards among agents is a relevant issue, affecting both scalability and the quality of joint policies. This issue of credit assignment concerns designating high reward to the agents with a desirable behavior, thus avoiding agents enjoying the rewards without contributing in achieving the intended joint goal.

2. Deep MARL

Tabular function representations in reinforcement learning (RL) have many successes [2] in relatively low-dimensional problems, but it has two major drawbacks: (a) The designer of the application had to hand-craft the state representations, and (b) methods store each state or state-action value (V -value or Q -value, respectively) independently, resulting in slow learning in large state-action spaces and poor generalization abilities.

To resolve these problems, the deep Q -network [3] method successfully combined RL with neural networks (NNs). It is well established that the combination of RL with nonlinear function approximators such as NNs can be unstable and result in divergence [4]. This instability has several causes: the correlations present in the sequence

of observations, the fact that small updates to Q values may significantly change the policy—and therefore change the data distribution of the samples produced by the rollouts, and the correlations between the action values and the target values. Deep Q-Network (DQN) [3] uses an NN to approximate Q-values, modeling the agent's policy.

Two vital elements of DQN that address these issues are the target network and the experience replay memory. The target network mitigates the effect of constantly moving targets, by incorporating a second network from which the targets are sampled. This network is periodically updated with the weights of the online network. The addition of a uniform experience replay memory decorrelates the samples collected during rollouts, by randomizing over the data, thereby smoothing over changes in the data distribution. During learning, the method applies Q-learning updates on samples (or minibatches) of experience drawn uniformly at random from the pool of samples stored.

Since the original DQN publication, many extensions have emerged [5–9], with double DQN [10] and prioritized experience replay [11] being the most well known. Double Q learning was originally introduced by H. Hasselt [12] and aims to address the problem of overestimating action values, a phenomenon inherent to the Q-learning method. This addition to the original method is considered to be standard practice and is particularly useful in the multiagent domain, where nonstationarity is a common phenomenon. Originally, the idea behind this method is to utilize two independent tabular representations of the Q function during training, where each Q function is updated with targets produced from the other Q function. Double DQN transfers this approach in the Deep RL setting, by exploiting the second approximator using a target network.

In the initial DQN approach, experience transitions are uniformly sampled from a replay memory. This approach ignores the significance of samples, replaying them at the same frequency that they were originally observed in the environment. Prioritized experience replay [11] enforces a priority over the samples, aiming to replay important transitions more frequently, and subsequently improve learning efficiency. In particular, the method proposes to assign higher priority to transitions with high expected learning progress, as measured by the magnitude of their temporal-difference (TD) error.

This new paradigm of combining Q-learning with NNs swiftly crossed over to the multiagent (MAS) field. Tampuu et al. [13] studied cooperation and competition between two DQN agents. The publication investigates the interaction between two agents in the well-known video game Pong, by utilizing two independent DQN architectures, one for each agent. By solely adjusting the rewarding scheme, competitive and collaborative behaviors emerge.

Deep deterministic policy gradient (DDPG) [14] combines DQN with deterministic policy gradient (DPG) [15] to address continuous actions. As a variant of actor-critic algorithms it incorporates two separate NNs: the actor, which models the policy, and the critic, which provides feedback on the desirability of an action a in a state s . This desirability can be expressed by means of learnt V -values, Q -values, or advantages. MADDPG [16] extends DDPG in MAS settings: Each agent is given a dedicated policy network. This approach renders the method not scalable: It is not viable to maintain and train hundreds, if not thousands, of policy networks. In addition, detrimental to the scalability is the fact that, during training, the method utilizes a centralized critic, which takes as input the observations and actions of all agents. The input size of the critic can explode, depending on the size of the agent population and the state dimensions.

Multiagent deep deterministic policy gradient (MADDPG) employs a technique that has many variants and is vital in understanding deep MARL. This technique is called centralized training and decentralized execution (CTDE). A naive way to provide agents with a joint policy is to train in parallel independent policies, one for each agent, following the independent learners paradigm. This approach can produce results in low-dimensional problems, but in real-world scenarios is inefficient and unstable. In order to alleviate these problems produced by the nonstationarity of an MAS environment, many algorithms, one of which is MADDPG, employ some form of centralized training, thus exploiting information that is available during training but often unavailable during execution. In particular, MADDPG trains a centralized critic, which takes as input the global state and agents' joint action. During the execution phase, this information is inaccessible by agents, and agents act in a decentralized manner. Another well-known application of CTDE is QMIX [17]. During training, the learning algorithm has access to all local action observation histories and global state, but each agent learns a policy that conditions actions on local agent observations.

Parameter sharing, first introduced by Gupta et al. [18], is the extreme case of CTDE: It learns a single policy shared by many agents. The main idea is that this single policy should be able to adequately describe the behavior of different agents with the same goals. Immediately apparent is the important assumption that the agents are homogeneous, meaning that they decide on the same state-action space. This assumption is relaxed by the same publication [18], which “tagged” observations with agent identifiers, allowing differentiating agents according to their goals, and responding accordingly. The resulting policy can be more robust, given the fact that it has been trained with samples that potentially belong to different parts of the state space, explored by different agents. We consider the parameter sharing approach to be the cornerstone of any scalable algorithm.

Castañeda proposes two multiagent variants of DQN [19]. Repeated update Q-learning extends DQN by updating each action inversely proportional to the probability of selecting it, practically changing the learning rate based on the action probability. It is designed to mitigate the inherent overestimation of state values by Q-learning, much like double Q-learning [10]. Loosely coupled Q-learning defines a diffusion function and utilizes eligibility traces in order to associate negative rewards with states. The combination of the diffusion function with eligibility traces is used to define a function, which indicates the necessity for cooperation, expressing the probability of an agent carrying on an action independently. It combines this with Dec-MDP and two Q-value functions, one for independent acting (when appropriate) and one for coordinating with others.

Credit assignment concerns the difficulty to give credit and provide higher reward to the agents with a desirable behavior, thus accelerating learning in MAS settings. A common phenomenon, called “lazy agent,” occurs when an agent does not participate actively in a cooperative solution, enjoying the rewards resulting from the cooperation of others. Various approaches have been proposed to deal with this problem, with the difference reward [20] and reward shaping [21, 22] being the most well known.

Difference reward [20] introduces a method to visualize the desirable properties of a reward function, thus facilitating the creation of new reward structures based on the specific needs of the domain. Specifically, the authors in [20] focus on two aspects of the reward function. The first is called *factordness* and expresses how well the reward promotes coordination among agents in different parts of a domain's state-space. The second is called *learnability* and expresses how easy it is for an agent to learn to

maximize its reward, by measuring the reward's sensitivity to the agent's actions. The main idea behind reward shaping is to utilize some prior knowledge while engineering the reward function, in order to reduce the training time, by reducing the number of suboptimal actions taken [22]. In the most general form, reward shaping can be as simple as $R' = R + F$, where R is the original reward and F a positive scalar reward, designed to encourage the agent to move toward the goal. Here, most of the research focuses on the principles, which would lead to an effective reward design, as well as how the optimal policy changes as an effect of reward design.

Value decomposition network (VDN) [23] is the first DMARL method that addresses the credit assignment problem. VDN represents joint action value as a summation of local (individual agents') action values conditioned on agents' local observations. The fact that the agents' local value function depends only on local observations facilitates agents' understanding of the received rewards. This method is not scalable, because it utilizes distinct NNs for each agent's policy. QMIX [17] provides a more general case of VDN using a mixing NN that combines all independent Q-values into Q_{tot} , thus approximating a broader class of functions for joint action values. Specifically, moving from the VDN's assumption of additivity, QMIX's mixing network can approximate monotonic relations between individual and the global Q-values. Both works are based on the individual-global-max (IGM) principle, according to which, the joint greedy action should be equivalent to the set of individual greedy actions of agents.

VDN and QMIX address a subset of factorizable MARL tasks due to their structural constraint in factorization of additivity and monotonicity, respectively. Later works have improved the representation ability of the mixing network. QTRAN [24] achieves more general factorization by transforming the original joint action-value function into a new, easily factorizable one with the same optimal actions as the original. NDQ [25] combines value function factorization learning with communication learning, by introducing an information-theoretic regularizer, aiming to reduce interagent communication while maintaining performance.

ROMA [26] combines MARL, mixing networks in particular, with the role concept [27–33]. A role is a comprehensive pattern of behavior, often specialized in some tasks. Agents with similar roles will show similar behaviors and thus can share their experiences to improve performance. The main drawback of this approach is the demand of exploitation of prior domain knowledge in order to decompose tasks and predefine the responsibilities of each role, which necessitates adding to role-based MAS dynamic and adaptive abilities to perform effectively in dynamic and unpredictable settings. However, the specification of roles may not be appropriate for any domain. ROMA introduces two regularizers to enable roles to be dynamically identifiable, in order to exploit the benefits of both, role-based and learning paradigms. Following the QMIX [17] framework, it utilizes independent policy networks and a centralized mixing network.

QPLEX [34] focuses on ensuring that the IGM principle (as specified above) stands, while reformalizing it as an advantaged-based IGM. QPLEX replaces the original mixing network of QMIX [17] with a duplex dueling network architecture [5], which induces the joint and local (duplex) advantage functions, to factorize the joint action-value function into individual action-value functions. This duplex dueling structure encodes the IGM principle into the neural network architecture. The architecture uses an individual action-value function for each agent in combination with the centralized duplex dueling component. The method manages to achieve higher win rates in numerous Starcraft II scenarios [35] against multiple baselines such as VDN, QMIX, and QTRAN [17, 23, 24] in experiments with up to 27 agents.

Another well-known approach is the counterfactual multiagent policy gradients method (COMA) [36]: It decomposes the global reward to the agents, utilizing a counterfactual baseline inspired by difference rewards [20]. Similarly to MADDPG, COMA utilizes a centralized critic and each agent has a distinct policy network, so the drawbacks regarding scalability are common to MADDPG.

Concluding the above, the line of research on value decomposition has different focal points rather than scalability, thus does not produce methods ideal to scale up to thousands of agents. In addition to that, as far as credit assignment problem is concerned, although a vital aspect of MARL, we do not consider that the resolution of this problem is a prerequisite toward methods' scalability. However, it is a related issue in settings where rewards are not inherently decomposed to individual agents.

3. Scalable deep MARL

Differentiable interagent learning (DIAL) [37] is the first proposal for learnable communication utilizing DQNs. Agents generate real-valued messages, which are broadcasted to the other agents. During centralized training, these messages are practically gradients, allowing end-to-end training across agents. During decentralized execution, messages are discretized and mapped to a predefined set of communication actions. There are two major reasons that this approach can not scale effectively: With no parameter sharing in place, each agent uses its own, independent policy NN. Also, every agent communicates with everyone else throughout training. This, beyond the communication cost, could result in agents receiving misleading or noisy information, in the form of gradients.

CommNet [38] aims to learn a communication protocol alongside the policies of cooperating agents. CommNet consists of a centralized feed-forward NN, with the observations of all agents as input and their actions as output. Each layer represents a communication step between the agents and consists of one decision module per agent. While the first layer uses an encoder function, every hidden layer module takes the internal state h as well as the broadcasted communication vector c as input and outputs the next internal state. These internal states are averaged and broadcasted as the next communication vector. CommNet has three major limitations regarding scalability. First, all agents use the same centralized network, which has to be big enough to accommodate this design approach, due to the size of the input in the form of the global state. In addition, this renders the method unsuitable for heterogeneous agents. Second, CommNet calculates the mean of messages between layers, therefore assuming that all agents are of equal importance, which could be unsuitable in some settings. Finally, the most important disadvantage is the fact that all agents have to communicate with everyone. This could result in significant communication overhead with noise in the communication channel: e.g., an agent could receive a communication vector consisting of the average observations of irrelevant agents. The last drawback could be mitigated by the local connectivity model extension proposed in the original CommNet publication [38]. According to this extension, agents can only receive messages from a dynamic group of agents, which are within a certain range. However, no experiments are provided for this extension.

Toward resolving the mandatory global communication problem, an extension of CommNet, called vertex attention interaction network (VAIN) [39], employs an attention mechanism. This attention mechanism improves the performance of the original method by modeling the locality of interactions and thus determining which

agents will share information. The authors claim that this method can make CommNet suitable for as many as 1000 agents. In the experiments provided, VAIN achieved better score than CommNet, although for a small number of agents.

Another method with scalability potential is BiCNet, [40]. BiCNet is an extension of the actor-critic formulation where agents use a bidirectional recurrent NN (RNN) [41]. The recurrent network serves as a bidirectional communication channel but also as a local memory saver: Each agent is able to maintain its own internal states, as well as to share information with its neighbors. Similarly to CommNet, the communication channel serves to broadcast agents' local observations. Contrary to CommNet agents, BiCNet agents utilize additive messages, while the method makes no assumptions on agents' homogeneity/interchangeability. The major drawback is that, similarly to CommNet, all agents communicate with everyone. Later works [42] show that the ability of BiCNet to learn effective policies is reduced as the number of the agents increases. This deterioration of effectiveness is attributed to the lack of a mechanism capable of capturing the importance of information from different agents.

TarMAC [43] proposes an architecture designed specifically to allow each agent to chose to which other agents to address messages to, as well as to which messages it will pay attention to. It introduces a signature-based soft attention mechanism with a key, which encodes properties of intended recipients (as opposed to specific agent identification), to be part of the message. The receiver of the message takes this key into consideration and decides whether it is relevant. The method is enhanced by multiple rounds of communication and collaborative reasoning. TarMAC utilizes the actor-critic framework, with a shared policy network and a centralized critic. It is evaluated on four diverse environments against CommNet [38], as well with variants of itself without the attention mechanism or communication at all. The sophistication of the communication scheme could be a disadvantage regarding scalability, as multiple rounds of communication between thousands of agents would result in significant overhead. The main drawback of the method though, with scalability in mind, is the usage of a centralized critic, which takes as input the predicted actions and hidden states of all agents.

Coder [44] is a hierarchical approach, with three distinct hierarchical levels to solve a traffic congestion problem with agents being the traffic intersections. First, a centralized base environment is trained, which is a small and simple subproblem compared with the original one, i.e., a single intersection managing incoming traffic. Here, two training alternatives are considered, a DQN variant and a DDPG variant called Wolpertinger architecture [45]. The next step is called regional DRL, where the parameters of the base environment are shared to a small number of neighboring agents controlling similar but not identical parts of the environment (e.g., different traffic dynamics). To tackle these differences, additional refinement through training is required. The final level combines all regional policies and adds a global dense layer. In order to choose actions while incorporating some form of coordination between the regional policies, an iterative action search is employed. This search starts with the concatenation of the actions produced by the regional nets and attempts to find a globally better alternative. This approach is shown to work with a maximum size of 96 adjacent intersections. A similar approach to Coder [44], we call it here KT DQN [46] expands DQN. Aiming to speed up training and produce better results, it uses single agent training before applying the policy to MAS settings. In doing so, the method freezes all the weights of the single-agent policy network model that are transferred to the MAS scenario, with the exception of the ones between the last hidden layer and the output. Coder and KT DQN utilize a hierarchical approach that initializes learning from a simplified single agent environment. This can indeed speed-up learning, in

contrast to start learning from scratch. However, this approach is not straightforwardly transferable between applications. For example, using the regional DRL step of coder calls for a separate design, i.e., decomposing the problem into subproblems, for different applications.

An interesting take on how to utilize an agent population in order to facilitate exploration is presented by J. Leibo [47]. The method proposed is a variant of IMPALA [48]: In the simulation used every agent is a member of an animal species. These homogeneous agents share the same policy network. The paper does not address interagent communication or cooperation, explicitly. Instead, it utilizes the multiagent framework mainly to encourage a more robust exploration. Cooperative behavior emerged when incentives are given to the agents: For example, experiments are presented with agents rewarded for specializing in eating a specific kind of food. The total number of agents present in the simulation reported are 960, and the population is considered to be dynamic.

Mean field MARL [49] proposes two algorithms with the explicit aim to improve MARL scalability. The main idea is to calculate the mean action of an agent's neighborhood, considering that each agent interacts with a virtual mean agent instead of n agents. The two proposed alternatives are mean field Q (MF-Q) and mean field actor-critic (MF-AC), among which the MF-Q approach has superior sample efficiency, which becomes more apparent as the number of agents gets bigger. MF-Q has been empirically shown to scale up to 1000 agents. Mean field MARL does not explicitly address credit assignment, but provides rigorous mathematical proof of convergence to Nash equilibrium. Although both algorithms are scalable, their main drawback lies in the coordination scheme: Averaging the neighborhood of an agent can result in loss of important information and lackluster cooperation [50].

Inspired by the factorization machines [51, 52], FQL [53] utilizes a composite deep neural network architecture, combining Q and V nets, for computing a low-rank approximation of the Q -function, by reformulating the multiagent joint-action Q -function into a factorized form. Depending on agent roles, agents are divided into G groups, and agents within each group share network parameters. While ensuring efficiency, the uniformity assumption between agents might not be suitable for various real-world applications. In addition, since the factorized Q -function for each agent requires the knowledge of the other agents' current states and their last actions for both training and execution, the algorithm mainly addresses the MARL problems with a central controller that communicates the global information to all the agents. In the experimental section, the method produces competitive results in settings with agent populations as large as 500 agents. As the size of the agent population is increased, from 100 to 500, MF-Q [49], which is one of the baselines used, seems to be a more suitable method.

CoLight [54] aims to enable agent cooperation in a large-scale road network with hundreds of traffic signals, recognizing that RL-based methods at the time fail to reach optimal interagent communication. To achieve this, authors introduce the utilization of graph attentional networks. At the core of the method is a Q -value prediction deep network. This network incorporates an observation embedding layer, the hidden neighborhood cooperation layers, and the output Q -value prediction layer. Similarly to mean field MARL [49], the method "averages" neighbors' influence in the broadcasted messages. Although experiments with simulated as well as real-world data involve up to 196 agents, authors claim scalability to thousands of agents, based on method's complexity analysis [54]. Contrasted against various baselines, CoLight shows advanced scalability, as well as sample efficiency.

As already pointed out, many major approaches such as DIAL, CommNet, and BicNet [37, 38, 40] suffer from the lack of agents' ability to differentiate between useful information driving interagent cooperation from noisy or misleading information. ATOC [42] strives to eliminate this specific issue by proposing an attentional communication model that achieves interagent communication between a large amount of agents. The method expands DDPG and uses an attention module as well as a bidirectional LSTM, which serves as a communication channel. The LSTM module plays the role of integrating neighboring agents' internal states, thus creating a message from their combined intentions, and guiding the agents toward coordinated decision-making. Actor and critic networks share parameters to ensure scalability. Indeed, the method seems to challenge the known approaches (DDPG [14], CommNet [38], and BicNet [40]), while agents show division of work and meaningful cooperation.

Graph convolutional reinforcement learning (DGN) [50] has the explicit goal to handle highly dynamic environments, where agents constantly change neighborhoods. Toward this goal, the paper proposes an MAS framework in which agents are connected in a graph where each agent is a node and edges connect neighbors. Neighborhoods are determined by agent distance or other measures, e.g., communication range or critical interactions, and can vary over time. Communication is allowed only between neighbors, in order to minimize inefficiency. DGN consists of an observation encoder, convolutional layers with relation kernels, and a Q network. The method was compared against well-known algorithms such as CommNet [38] and MFQ [49] and is shown to achieve very competitive results in environments with up to 140 agents. However, experiments with greater agent population are needed in order to assess the effectiveness of the method as the environment gets more complex.

Lin et al. [55] proposes two distinct methods, which, very closely to our aims, aim to solve large-scale demand-capacity imbalance problems in the air traffic management domain. The environment simulates a population of approx. 5000 homogeneous agents, acquiring identical rewards. Both methods assume that agents have the same action values. In practice, this assumption means that the agents are not simply homogeneous, but interchangeable. This assumption allows the building of a centralized action-value table, which is utilized for coordination of agents' actions. The first method is called contextual deep Q-learning. It utilizes a table with centralized action values to form a collaborative context, which prevents agents from choosing suboptimal actions, thus avoiding unwanted or redundant behavior such as agents exchanging positions with one another. The second proposed method, called contextual actor-critic, describes an actor critic variant of the contextual DQN. It uses a centralized value function shared by all agents and a parameter-sharing policy network.

Closely related works, with scalability in mind, are those presented by Nguyen et al. [56–58]. In particular, AC for CDec-POMDPs [57] is based on the FEM algorithm [56] and presents an actor-critic algorithm for optimizing collective decentralized POMDPs. It achieves extreme scalability and provides experiments with up to 8000 agents. Subsequent work proposes MCAC [58], which focuses on difference rewards, also addressing the credit assignment problem. A fundamental idea underlining the work described in these papers and a vital aspect in order to achieve scalability is exploiting the count of agents taking the same action a in a state s . This count, as well as other, more complex measures based on this, serves a statistical basis for training, eliminating the need to collect trajectory samples from every agent. Instead, the resulting policy is dependent on count-based observations. Therefore, this method

does not explicitly assume communication. It rather assumes full access to all the count-based information during training. During execution, agents execute individual policies without accessing centralized functions. Similarly to contextual DQN [55],

Method	Q Net (s)	V Net	Agents	Communication	MAS population
DQN [3]	I	—	Het	—	2
PS DQN [18]	PS	—	Hom	Impl, global (policy)	200
MADDPG [16]	I	C	Het	Impl, global (critic)	6
VDN [23]	I	—	Het	Impl, global (policy)	2
QMIX [17]	I	—	Het	Impl, global (mixing net)	8
QTRAN [24]	I	C	Het	Impl, global (mixing net)	4
NDQ [25]	I	C	Het	Expl, local (message encoder)	10
ROMA [26]	I	C	Het	Impl, global (mixing net)	27
QPLEX [34]	I	C	Het	Impl, global	27
COMA [36]	I	C	Het	Impl, global (critic)	5
DIAL [37]	I	—	Het	Expl, global (messages)	4
CommNet [38]	C	—	Hom	Expl, global (communication vectors)	500
VAIN [39]	C	—	Hom	Expl, local (attention mechanism)	50
BiCNet [40]	PS	PS	Het	Expl, global (messages in RNN)	32
TarMAC [43]	PS	C	Hom	Expl, local (soft attention mechanism)	20
Coder [44]	PS (KT)	PS (KT)	Hom	Impl, global (policy)	96
KT DQN [46]	PS (KT)	—	Hom	Impl, global (policy)	20
MF-Q [49]	PS	—	Hom	Expl, local (mean agent)	1000
MF-AC [49]	PS	PS	Hom	Expl, local (mean agent)	1000
FQL [53]	PS	PS	Het	Impl, global (policy)	500
CoLight [54]	PS	—	Hom	Expl, local (attention mechanism)	196
ATOC [42]	PS	PS	Hom	Expl, local (attention mechanism/ LSTM)	100
DGN [50]	PS	—	Hom	Expl, local (observations in conv layers)	140
cDQN [55]	PS	—	IC	Expl, local (collaborative context)	5000
cA2C [55]	PS	C	IC	Expl, local (collaborative context)	5000
FEM [56]	PS	PS	IC	Impl, global (policy)	20
AC CDec-POMDPs [57]	PS	PS	IC	Impl, Global (policy)	8000
MCAC [58]	PS	PS	IC	Impl, Global (policy)	8000

Table 1.
Reviewed methods' characteristics.

this approach assumes that the agents are interchangeable. This assumption is necessary to achieve the required scalability, but is not applicable to every problem.

4. Concluding remarks

In this paper, we provide a review of state of the art DMARL methods with significant scalability potential and interagent coordination capabilities in large-scale MAS settings. **Table 1** lists all the reviewed methods with the characteristics, which are considered essential for their scalability, as mentioned in the introductory section. The abbreviations used are the following: Independent learners (I), Parameter Sharing (PS), Centralized model (C), Knowledge Transfer (KT), Hetero/Homo-geneous agents (Het/Hom), Interchangeable agents (IC), Implicit (Impl), Explicit (Expl).

The first conclusion we can draw is the importance of parameter sharing in large agent populations. As the number of agents grows, parameter sharing shows by far the most potential for scalability. It is impractical to train thousands of independent networks for each agent or to utilize a centralized approach whose input size would explode as the number of agents and the size of their observations grow larger. We can clearly see in **Table 1** that all works that provide experiments with large agent populations concur on that approach.

Another important conclusion from this study is the fact that, as scalability potential gets more prominent, stricter assumptions are made on the agents and on their environment. Specifically, the approach that manages to scale up to the largest agent population [57] assumes interchangeable agents. In order for these methods to find broad practical applicability, such assumptions have to be relaxed. In numerous practical applications, agents are not homogeneous, but more importantly, interchangeable agents are a more rare occurrence. Additions to the methods, in order to incorporate heterogeneous agents, would be of great value. An important point here is that parameter sharing assumes homogeneous agents. Further research and experimenting on the direction of techniques such as labeling agents, as proposed in the original parameter sharing publication [18], should be beneficial in that regard.

Finally, we can conclude that as the need for scalability becomes more prevalent, targeted and detailed communication becomes more challenging to achieve. For example, MF-Q [49], a method that shows great scalability capabilities, assumes that all agents affect the recipient of their messages equally. Local communication, on the other hand, affects the scalability of methods, while further studies on the use of attention and messages' combination mechanisms are necessary to prove the potential to operate in environments with thousands of agents.

Acknowledgements

This work is supported by the TAPAS project, which has received funding from the SESAR Joint Undertaking (JU) under grant agreement No 892358. In addition, this work was partially supported by the University of Piraeus Research Center (UPRC).

IntechOpen


IntechOpen

Author details

Theocharis Kravaris* and George A. Vouros
University of Piraeus, Piraeus, Greece

*Address all correspondence to: hariskrav.unipi@gmail.com

IntechOpen

© 2022 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. 

References

- [1] Kravaris T, Spatharis C, Bastas A, Vouros GA, Blekas K, Andrienko G, Andrienko N, Garcia JM. Resolving congestions in the air traffic management domain via multiagent reinforcement learning methods. 14 December 2019. arXiv preprint arXiv:1912.06860
- [2] Sutton RS, Barto AG. Reinforcement learning: An introduction. Cambridge, Massachusetts, USA: MIT Press; 13 November, 2018
- [3] Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, et al. Human-level control through deep reinforcement learning. *Nature*. 2015; **518**(7540):529-533
- [4] Tsitsiklis J, Van Roy B. Analysis of temporal-difference learning with function approximation. *Advances in Neural Information Processing Systems*. 1996;**9**:1075-1081
- [5] Wang Z, Schaul T, Hessel M, Hasselt H, Lanctot M, Freitas N. Dueling network architectures for deep reinforcement learning. In: *International Conference on Machine Learning*. New York, NY, USA: JMLR: WCP; 11 June 2016. pp. 1995-2003
- [6] Fortunato M, Azar MG, Piot B, Menick J, Osband I, Graves A, et al. Noisy networks for exploration. 30 June 2017. arXiv preprint arXiv:1706.10295
- [7] Mnih V, Badia AP, Mirza M, Graves A, Lillicrap T, Harley T, et al. Asynchronous methods for deep reinforcement learning. In: *International Conference on Machine Learning*. New York, NY, USA: JMLR: W&CP; 2016. pp. 1928-1937
- [8] Dabney W, Rowland M, Bellemare M, Munos R. Distributional reinforcement learning with quantile regression. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. California, USA: AAAI Press, Palo Alto; 2018
- [9] Hessel M, Modayil J, Van Hasselt H, Schaul T, Ostrovski G, Dabney W, et al. Rainbow: Combining improvements in deep reinforcement learning. In: *Thirty-second AAAI Conference on Artificial Intelligence*. California, USA: AAAI Press, Palo Alto; 2018
- [10] Van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double q-learning. In: *Thirty-second AAAI Conference on Artificial Intelligence*. California, USA: AAAI Press, Palo Alto; 2016
- [11] Schaul T, Quan J, Antonoglou I, Silver D. Prioritized Experience Replay. *InICLR (Poster)*. 1 January 2016
- [12] Hasselt H. Double Q-learning. *Advances in Neural Information Processing Systems*. 2010;**23**:2613-2621
- [13] Tampuu A, Matiisen T, Kodelja D, Kuzovkin I, Korjus K, Aru J, et al. Multiagent cooperation and competition with deep reinforcement learning. *PloS One*. 2017;**12**(4):e0172395
- [14] Qiu C, Hu Y, Chen Y, Zeng B. Deep deterministic policy gradient (DDPG)-based energy harvesting wireless communications. *IEEE Internet of Things Journal*. 2019;**6**(5): 8577-8588
- [15] Silver D, Lever G, Heess N, Degris T, Wierstra D, Riedmiller M. Deterministic policy gradient algorithms. In: *International Conference on Machine Learning*. New York, NY, USA: JMLR: W&CP; 2014. pp. 387-395

- [16] Lowe R, Wu YI, Tamar A, Harb J, Pieter Abbeel O, Mordatch I. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in Neural Information Processing Systems*. 2017;**30**:6379-6390
- [17] Rashid T, Samvelyan M, Schroeder C, Farquhar G, Foerster J, Whiteson S. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In: *International Conference on Machine Learning*. New York, NY, USA: JMLR: W&CP; 2018. pp. 4295-4304
- [18] Gupta JK, Egorov M, Kochenderfer M. Cooperative multi-agent control using deep reinforcement learning. In: *International Conference on Autonomous Agents and Multiagent Systems*. Cham, Switzerland: Springer International Publishing AG; 2017. pp. 66-83
- [19] Castaneda AO. *Deep Reinforcement Learning Variants of Multi-agent Learning Algorithms*. Edinburgh: School of Informatics, University of Edinburgh; 2016
- [20] Agogino AK, Tumer K. Analyzing and visualizing multiagent rewards in dynamic and stochastic domains. *Autonomous Agents and Multi-Agent Systems*. 2008;**17**(2):320-338
- [21] Devlin SM, Kudenko D. Dynamic potential-based reward shaping. In: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*. Cham, Switzerland: Springer International Publishing AG; 2012. pp. 433-440
- [22] Ng AY, Harada D, Russell S. Policy invariance under reward transformations: Theory and application to reward shaping. In: *ICML*. New York, NY, USA: JMLR: W&CP; 1999. pp. 278-287
- [23] Sunehag P, Lever G, Gruslys A, Czarnecki WM, Zambaldi VF, Jaderberg M, et al. Value-decomposition networks for cooperative multi-agent learning based on team reward. In: *AAMAS*. Cham, Switzerland: Springer International Publishing AG; 2018
- [24] Son K, Kim D, Kang WJ, Hostallero DE, Yi Y. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In: *International Conference on Machine Learning*. New York, NY, USA: JMLR: W&CP; 2019. pp. 5887-5896
- [25] Wang T, Wang J, Zheng C, Zhang C. Learning nearly decomposable value functions via communication minimization. In: *International Conference on Learning Representations*. 2019
- [26] Wang T, Dong H, Lesser V, Zhang C. ROMA: Multi-agent reinforcement learning with emergent roles. In: *International Conference on Machine Learning*. New York, NY, USA: JMLR: W&CP; 2020. pp. 9876-9886
- [27] Becht M, Gurzki T, Klarmann J, Muscholl M. ROPE: Role oriented programming environment for multiagent systems. In: *Proceedings Fourth IFCIS International Conference on Cooperative Information Systems*. CoopIS 99 (Cat. No. PR00384). New York, U.S.: IEEE; 1999. pp. 325-333
- [28] Stone P, Veloso M. Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork. *Artificial Intelligence*. 1999; **110**(2):241-273
- [29] Depke R, Heckel R, Küster JM. Roles in agent-oriented modeling. *International Journal of Software Engineering and Knowledge Engineering*. 2001;**11**(03):281-302

- [30] Ferber J, Gutknecht O, Michel F. From agents to organizations: An organizational view of multi-agent systems. In: International Workshop on Agent-oriented Software Engineering. Cham, Switzerland: Springer International Publishing AG; 2003. pp. 214-230
- [31] Odell J, Nodine M, Levy R. A metamodel for agents, roles, and groups. In: International Workshop on Agent-oriented Software Engineering. Cham, Switzerland: Springer International Publishing AG; 2004. pp. 78-92
- [32] Cossentino M, Hilaire V, Molesini A, Seidita V, editors. Handbook on Agent-oriented Design Processes. Berlin Heidelberg: Springer; 2014
- [33] Lhaksmana KM, Murakami Y, Ishida T. Role-based modeling for designing agent behavior in self-organizing multi-agent systems. International Journal of Software Engineering and Knowledge Engineering. 2018;28:79-96
- [34] Wang J, Ren Z, Liu T, Yu Y, Zhang C. QPLEX: Duplex dueling multi-agent Q-learning. In: International Conference on Learning Representations. 2020
- [35] Samvelyan M, Rashid T, de Witt C, Farquhar G, Nardelli N, Rudner TG, et al. The starcraft multi-agent challenge. In: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems. Cham, Switzerland: Springer International Publishing AG; 2019. pp. 2186-2188
- [36] Foerster J, Farquhar G, Afouras T, Nardelli N, Whiteson S. Counterfactual multi-agent policy gradients. In: Proceedings of the AAAI Conference on Artificial Intelligence. California, USA: AAAI Press, Palo Alto; 2018
- [37] Foerster J, Assael IA, De Freitas N, Whiteson S. Learning to communicate with deep multi-agent reinforcement learning. Advances in Neural Information Processing Systems. 2016;29:2137-2145
- [38] Sukhbaatar S, Fergus R. Learning multiagent communication with backpropagation. Advances in Neural Information Processing Systems. 2016;29:2244-2252
- [39] Hoshen Y. Vain: Attentional multi-agent predictive modeling. Advances in Neural Information Processing Systems. 2017;30:2701-2711
- [40] Peng P, Wen Y, Yang Y, Yuan Q, Tang Z, Long H, Wang J. Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games. 29 March 2017. arXiv preprint arXiv:1703.10069
- [41] Schuster M, Paliwal KK. Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing. 1997;45(11):2673-2681
- [42] Jiang J, Lu Z. Learning attentional communication for multi-agent cooperation. Advances in Neural Information Processing Systems. 2018; 31:7254-7264
- [43] Das A, Gervet T, Romoff J, Batra D, Parikh D, Rabbat M, et al. Tarmac: Targeted multi-agent communication. In: International Conference on Machine Learning. New York, NY, USA: JMLR: W&CP; 2019. pp. 1538-1546
- [44] Tan T, Bao F, Deng Y, Jin A, Dai Q, Wang J. Cooperative deep reinforcement learning for large-scale traffic grid signal control. IEEE Transactions on Cybernetics. 2019;50(6):2687-2700
- [45] Marden JR, Arslan G, Shamma JS. Cooperative control and potential games.

- IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). 2009; 39(6):1393-1407
- [46] Rădulescu R, Legrand M, Efthymiadis K, Roijers DM, Nowé A. Deep multi-agent reinforcement learning in a homogeneous open population. In: Benelux Conference on Artificial Intelligence. New York, NY, USA: JMLR: W&CP; 2018. pp. 90-105
- [47] Leibo JZ, Pérolat J, Hughes E, Wheelwright S, Marblestone AH, Duéñez-Guzmán EA, et al. Malthusian Reinforcement Learning. In: AAMAS. Cham, Switzerland: Springer International Publishing AG; 2019
- [48] Espeholt L, Soyer H, Munos R, Simonyan K, Mnih V, Ward T, et al. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In: ICML. New York, NY, USA: JMLR: W&CP; 2018
- [49] Yang Y, Luo R, Li M, Zhou M, Zhang W, Wang J. Mean field multi-agent reinforcement learning. In: International Conference on Machine Learning. New York, NY, USA: JMLR: W&CP; 2018. pp. 5571-5580
- [50] You J, Liu B, Ying Z, Pande V, Leskovec J. Graph convolutional policy network for goal-directed molecular graph generation. *Advances in Neural Information Processing Systems*. 2018; 31:6410-6421
- [51] Rendle S, Schmidt-Thieme L. Pairwise interaction tensor factorization for personalized tag recommendation. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining. New York City, New York, USA: ACM. 2010; pp. 81-90
- [52] Rendle S. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*. 2012;3(3):1-22
- [53] Zhou M, Chen Y, Wen Y, Yang Y, Su Y, Zhang W, et al. Factorized q-learning for large-scale multi-agent systems. In: Proceedings of the First International Conference on Distributed Artificial Intelligence. New York City, New York, USA: ACM; 2019. pp. 1-7
- [54] Wei H, Xu N, Zhang H, Zheng G, Zang X, Chen C, et al. Colight: Learning network-level cooperation for traffic signal control. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management. New York, NY, United States: Association for Computing Machinery; 2019. pp. 1913-1922
- [55] Lin K, Zhao R, Xu Z, Zhou J. Efficient large-scale fleet management via multi-agent deep reinforcement learning. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. New York, NY, United States: Association for Computing Machinery; 2018. pp. 1774-1783
- [56] Nguyen DT, Kumar A, Lau HC. Collective multiagent sequential decision making under uncertainty. In: Thirty-First AAAI Conference on Artificial Intelligence. California, USA: AAAI Press, Palo Alto; 2017
- [57] Nguyen DT, Kumar A, Lau HC. Policy gradient with value function approximation for collective multiagent planning. *Advances in Neural Information Processing Systems*. 2017; 30:4319-4329
- [58] Nguyen DT, Kumar A, Lau HC. Credit assignment for collective multiagent RL with global rewards. *Advances in Neural Information Processing Systems*. 2018;31:8102-8113