Chapter 4

Probabilistic Routing in Opportunistic Ad Hoc Networks

Vangelis Angelakis, Niki Gazoni and Di Yuan

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/53997

1. Introduction

Routing packets in multi-hop ad hoc wireless networks poses a great challenge mainly due to the unreliability of the wireless links and the inherent interference of the wireless medium [13]. Due to these characteristics, traditional wired routing schemes that select the best path towards a destination and forward the packet to a specific next hop, have proven ill-suited for networks utilizing the wireless medium and relying on lossy broadcast links. Lately, a new routing paradigm, namely *opportunistic routing*, has been proposed to cope with unreliable transmissions by taking advantage of the broadcast nature and spatial diversity of the wireless medium [2, 6].

Opportunistic routing constitutes a new routing paradigm leveraging the nodes' ability to overhear a broadcast packet. Its core difference from traditional routing schemes in that forwarders can be selected from the group of recipients of the packet even after its transmission, hence there is no hard commitment to a predetermined path. This flexibility enables opportunistic routing to combine multiple weak links to create a reliable route, as well as to exploit unexpectedly long transmissions. The added forwarding reliability in transmission reduces the retransmission cost, which in turn improves the throughput and energy efficiency [10, 16]

Proposed opportunistic protocols demonstrate a lack of concrete understanding of the way key wireless networking primitives and design decisions affect the performance of an opportunistic routing scheme. As a result, it is unclear to which extent the improved performance of these protocols owes to their opportunistic design and to which extent it is affected by other design features that can also be applied to traditional routing schemes [19, 20, 22]. Opportunistic protocols which decide on forwarders in a centralized manner require the exchange of node coordination messages, leading to high overhead and increased resource

consumption. Furthermore they require global knowledge of the topology which makes them prone to poor performance in the event of misinformation.

On the other hand, localized probabilistic forwarding decision protocols, which have been designed mostly for use in sensor networks, have to trade high performance for robustness and simplicity. In fact some of them partly rely on flooding techniques, thus demonstrating a percolation behavior, which leads to unnecessary transmissions [23].

In designing of an opportunistic routing scheme, there are two key design decisions [10, 11]. Firstly, how node receiving a packet should decide to forward or not. Secondly, provided that a node has decided to relay a packet, based on some metric, when is the most appropriate time to do so. Here, using a simulation framework, we examine how the forwarding decisions and transmission timing affect performance and under which channel error conditions and topology density it is beneficial to use opportunistic routing instead of traditional routing. Furthermore, we develop an opportunistic forwarding scheme, whose parameters can be tuned to allow for low resource consumption and high delay performance, while being robust to misinformation. We provide evidence which confirms that the suggested protocol supports multiple flows in a network, a weakness of existing early demonstrated solutions in the literature. Finally, to evaluate the scheme, and the role of the wireless primitives in forwarding, we provide comparisons to single-path routing and two opportunistic routing protocols, SOAR [15] and Directed Transmission [12]. Our simulation results, under various channel error and misinformation conditions, demonstrate that the proposed routing protocol outperforms both SOAR which uses a centralized scheme for forwarding decision-making and Directed Transmission which, designed for sensor networks, is distributed using only local information.

1.1. Routing schemes in multi-hop wireless networks

1.1.1. Single-path routing

Initially, the routing techniques that have traditionally been used in wired networks were transported for use in multi-hop wireless networks as well. These routing protocols typically rely on choosing the best sequence of nodes between a source and destination, by some metric, and then forward each packet through that path, until something goes wrong, i.e. packets start getting lost. However, in the wireless domain, the performance of single path routing would often prove unsatisfactory due to the fact that, especially in multi-hop networks, link conditions are highly varying due to interference that such a single path may be far from optimal. Typically such protocols yield highly volatile routes incurring very high overhead costs in end-to-end communication. Most of the well-established routing protocols, such as DSR [7], AODV [14], and DSDV [4] fall into this category.

1.1.2. Multi-path routing

Towards improving routing performance, in multi-hop wireless networks, multi-path routing takes advantage of the potential of multiple alternative paths between a source-destina-

tion pair. The multiple paths established can be overlapping, edge-disjoint or node-disjoint with each other [6]. Data traffic is then split, according to some routing goal, along these multiple paths avoiding congestion and to using network resources with increased efficiently. Multi-path routing can yield a variety of benefits such as fault tolerance, increased bandwidth, or improved security. It is typically proposed in order to increase the reliability of data transmission or to provide load balancing [11].

1.1.3. Opportunistic routing

Opportunistic routing differs from the above traditional routing techniques in that it leverages the broadcast nature of wireless medium and the route is typically generated on the fly, i.e. the transmitting nodes may vary dependent on the actual packet broadcast transmission. As opposed to multi-path routing, it can remain "commitment-free" to the number of predetermined paths. Among the nodes that receive the packet, the one with the best conditions to the destination may be chosen to relay [3, 17, 21]. This can mitigate the effect of losses or increased overhead due to unreliable and unpredictable wireless links.

The key opportunistic routing benefits are the following two [15]:

i. Opportunistic routing can be used to combine multiple weak links to produce a virtual strong link.

As an example consider a node with five candidate relay neighbors and a distant destination and assume there is a 0.2 packet success rate to each of the neighbors, while each of them has a packet success rate of 1.0 to the destination. Under a traditional routing protocol, one of the five neighboring nodes would be selected as relay, resulting, on average, in five transmissions to send a packet from the source to the relay node, and then one more transmission from the relay node to the destination.

Opportunistic routing can consider the five neighboring nodes as one virtual link that can forward the packet to the destination. This opportunistic virtual link will then have a success rate of: $1 - (1 - 0.2)^5 = 0.672$. So, on average only 1/0.67=1.487 transmissions will be required to deliver a packet to *at least one* of the five intermediate nodes, and one more transmission is required for an intermediate node to forward. Hence, with 2.487 transmissions required on average end-to-end packet delivery through the opportunistic virtual link, achieves a throughput gain of x2.4 over traditional routing.

ii. Opportunistic routing can take advantage of "against-the-odds" successful transmissions, to achieve increased throughput.

In traditional routing protocols there typically is a trade-off reflected in the route selection metric between link quality and the spatial progress, in terms of distance from the destination, each transmission achieves. Consider a tandem of 4 nodes A, B, C, and D where the channel conditions from A to B is expected to be better than those of A to C, and so on due to proximity. Then indeed a transmission from A to B is more likely to be successful, albeit the information will physically progress less hence an added transmission to C may be required to guarantee, with some probability margin, reaching the final destination.

Not, in advance, committing to who will forward a packet to the final destination in such a case can prove beneficial if for some reason the channel conditions prove unexpectedly favorable to the long transmission from A to C. Hence a rule of thumb stating that: "Among the nodes that receive the packet, the one closest to the destination should forward" comes naturally in the case of opportunistic routing.

1.2. Related opportunistic routing works

Here, we introduce the major opportunistic routing protocols our work is based on, and/or is compared against.

1.2.1. Extreme Opportunistic Routing: ExOR

ExOR [1] is a routing protocol for wireless multi-hop networks that was implemented on the RoofNet testbed at the Massachusetts Institute of Technology. ExOR integrates routing and MAC protocols, still all packets are broadcast at layer 3. It improves routing performance by utilizing opportunistically success over unstable long-range links (see above).

After collecting a batch of packets, which is uniquely identified by a *BatchID*, the source broadcasts each packet in the batch, listing the forwarding nodes in a priority order in the packet's header. Priority is set by the ETX metric to the destination [8]. The ETX metric, i.e. essentially the expected number of transmissions necessary to forward a packet along a route, can be considered to map to a distance from the destination, hence, the lower the metric, the higher the priority. Only nodes that are "closer" to the destination than the source are included in the potential forwarder set. Each packet carries a bitmap, marking packets that have been received by the sending node or nodes with higher priorities. A forwarder transmits a packet only if no forwarder with higher priority has explicitly acknowledged receipt of it. ExOR has good routing performance.

It is unlikely that a forwarder will receive the entire batch correctly, so the nodes that have stored fragments of the batch will need to schedule their transmissions. To that end, each forwarder uses a forwarding timer that ExOR has set to five packet durations times the number of higher priority nodes in the forwarder list, and which is a prediction of the time at which the node should start forwarding packets from its packet buffer. After each schedule cycle, the batch maps need to be updated by means of negative acknowledgments and when the destination has received 90% of the batch, the rest of it is sent using traditional routing, because the overhead would be forbidding otherwise.

Due to the centralized coordination and scheduling that is needed between forwarders and the destination, ExOR incurs high overhead when the batch of packets to transmit is small as in bursty and short-lived flows, or the number of candidate forwarders is large.

1.2.2. The Simple Opportunistic Adaptive Routing protocol: SOAR

The SOAR protocol [15] has been proposed as an improvement to ExOR in order to support multiple flows. In SOAR the candidate forwarders are constrained to be on-or-near the

http://dx.doi.org/10.5772/53997

shortest path from source to destination. The cost metric is again ETX. A significant difference between ExOR and SOAR is that SOAR performs the routing decision process on a *per-packet* basis rather than on a batch. Finally, in SOAR the forwarder list is limited to 5 relays. Overall, SOAR incurs slightly less overhead than ExOR and restricts flows as close to the shortest path from source to destination as possible. It is considered crucial to avoid diverging paths, and select forwarders that can hear each other because overhearing is the only means to avoid duplicate transmissions.

To make the protocol reliable to ACK losses, selective ACKs are used. Each ACK packet contains the starting sequence number of out-of-order ACKs and a bit-map of out-of-order ACKs. In addition, SOAR uses a scheme of *piggy-backed acknowledgement* and *ACKs compression* to reduce the overhead of acknowledgments. When a node does not have much data to send, it should also send stand-alone ACKs to provide timely channel feedback.

However, the increase in throughput that is observed when SOAR is in use is partly due to its opportunistic scheme and partly due to its complex acknowledgment scheme, therefore making it unclear at what extent this protocol contributes to opportunistic routing [20].

1.2.3. Resilient Opportunistic Mesh Routing: ROMER

In ROMER [18], a credit mechanism is used to build a forwarding mesh topology on the fly centered on minimum cost paths. It aims to avoid the cost of retransmissions over persistently poor routes. Each packet is assigned a number of credits equal to the amount needed to reach the destination along the minimum cost path plus some extra slack allowing the packet to deviate slightly from this (shortest) path

Before forwarding a packet the transmitting node calculates the remaining credit that the packet would have if forwarded over a link and compares it to a threshold value. This scheme enables multiple forwarders to emerge near the source to ensuring route diversity and packet progress in view of lossy links. At the same time the scheme secures converge near the destination and remains close to the minimum cost path.

Greedy probabilistic forwarding is used to deliver the data packet along the instantaneously highest rate link with probability one and along other high-rated downstream links with a high probability value. Thus, it favors high quality links over low quality ones by selecting higher random forwarding probability. However, it does not account for resource conservation and catering to multiple flows.

1.2.4. Destination attractor and directed transmission

The Parametric probabilistic sensor network routing scheme of [15] proposes two protocols that forward a single packet with varying retransmission probability through a network of sensor nodes, focusing on simplicity and robustness to errors in distance estimation. The Destination Attractor assigns a higher retransmission probability to the packet, as it moves closer to the destination and reduces it, as the packet moves away from the destination. Distance check is performed by comparing the distance of the

source node in hops from the destination to the distance of the node currently holding the packet. The primary concern of this approach is to deliver as many copies of a single packet to the destination sensor as possible, without accounting for resource usage or delays. Directed Transmission improves probabilistic routing's performance by assigning an exponentially higher forwarding probability to nodes that are on the shortest path from the source to the destination and decreasing it as the packet strays from the shortest path. This leads to lower resource consumption than the Destination Attractor and can be tuned to resemble shortest path routing, when the misinformation is low enough. Both protocols are compared to shortest path routing as an ideal case.

2. Design considerations for a probabilistic routing scheme

As it has become apparent from our discussion thus far, in designing a probabilistic packet routing scheme for an opportunistic ad hoc network there are two key design decisions in the core of the forwarding procedure [5]:

a. *how* a node should decide whether or not to forward a packet it received and is not intended for him, and

b. *when* is the most appropriate time to transmit such a packet.

In what follows we develop a concise simulation framework to investigate how the forwarding decisions and transmission timing affect performance, and under which channel error conditions and topology density it is beneficial to use stochastic opportunistic routing instead of traditional routing techniques. With the insight gained, we develop a probabilistic packet forwarding procedure can be automatically tuned to allow for low resource consumption and delay-tolerant performance, while at the same time being robust to poor system information, which is a common case in ad hoc opportunistic networks. The procedure is embedded in a basic routing scheme, which is compared to single path routing and two reference opportunistic routing protocols, already discussed, SOAR and Directed Transmission. Simulation results under various channel error and errors in the metric calculation, demonstrate that our proposed protocol outperforms both SOAR, which uses a centralized forwarding decision scheme and Directed Transmission, which is highly distributed and designed for sensor networks.

2.1. Forwarding procedure principles & parameters

Apart from an initial neighbor discovery phase, routing decisions should require minimal information to be exchanged between nodes and no co-operation, to avoid imposing computational load on nodes with limited computational capacity and wasting bandwidth in exchanging control packets when bandwidth is limited. The forwarding procedure developed therefore had the key goals of simplicity and distributed decision-making. This is an obvious trade-off, as the simplest design solution would be flooding, a technique used in many sensor network opportunistically routing schemes (e.g. see [23] and the references therein).

http://dx.doi.org/10.5772/53997

The immense resource consumption of this scheme renders it completely impractical when amongst the targets of a routing protocol is e.g. to allow for multiple flows to exist simultaneously in the network – a goal not always necessary in event-detection sensor networks. Summarily the stochastic forwarding procedure proposed is intended as a flexible solution in routing applied in a variety of networks, where lossy areas are observed due to low link quality and that the topology is sufficiently dense to require opportunistic communication.

The major issues that need to be addressed in the design of the routing scheme that encompasses a stochastic forwarding procedure are (i) deciding which nodes should forward a packet, (ii) when to do so, (iii) how to acknowledge the reception of a packet. Candidate forwarders should be selected in order to improve the performance of the end-to-end flow. On the other hand the number of these forwarders should be limited to those that guarantee, with a high probability, the packets' progress towards their destination, in order to avoid excess resource consumption and allow better network resources utilization. Furthermore, the forwarding time differentiation between the candidates enables the mitigation of redundant transmissions, by limiting packet collisions. Lastly, since layer 3 broadcast packets typically do not implement link-layer acknowledgements, such a scheme should be devised. The destination could generate acknowledgements upon packet reception, which would be propagated back to the source, either opportunistically or via the shortest path. This would impose significant load on the network if the acknowledgements were "stand-alone" layer 2, or layer 3 control packets competing with the data packets and being susceptible to lossy links, which would cause significant delay until retransmission of a lost packet. However, the option of piggy-backing the acknowledgments onto data packets is limiting since it can be applied only when flows of symmetric direction exist between source and destination. Hop-by-hop acknowledgments provide robustness to the lossy links and mitigate delay on the other hand increase the overhead. Conversely, a passive acknowledgment scheme that utilizes overhearing of other nodes' transmissions, avoids imposing additional load to the network, contrary to explicit acknowledgment packets.

2.2. Forwarding cost metric

For the purpose of determining which nodes are the most suitable forwarding candidates for a particular packet, a routing cost metric needs to be used. For the framework presentation elucidation, the procedure presented here uses as a metric the physical distance between two nodes. However, any cost metric can be used to this end, (provided, the metric has well-defined minimum and maximum values). In order for a node to be able to calculate its routing metric from the destination of the packet, a neighbor discovery phase should take place between the nodes in network, before data packets can be exchanged. In our discussion we consider this to have taken place offline and do not assume it to be a part of the protocol, although it is an aspect that would increase the protocol overhead. Note that for the actual simulation comparisons the same assumption has been made for all protocols. Other metrics' calculation, as for example the ETX metric used in ExOr, can be incorporated in the specific protocol, with relative low design overhead. Note that such processes may be prone to misinformation due to the overly distributed nature of the network.

2.3. Forwarding probability function

In order to determine which nodes will forward a packet in a stochastic manner, a forwarding probability function is utilized. This function is the same for all nodes in the network. It's role is to map cost metrics to forwarding probabilities to each node that receives a packet. More specifically, this function is (a) non-increasing in the cost metric, i.e. lower routing cost values should not yield lower probability, and (b) bound, so that the minimum values it assigns are between zero and one. Given the cost metric assumed, our forwarding probability is assigned with respect to the distance of the receiving node from the packet destination and its relative position to the sender. Hence, when a node broadcasts a packet it has to include in the routing header its distance from the destination, so that the nodes that receive it, can calculate their forwarding probability for it. Obviously, out of the neighbors of the sender node, that is to say, the nodes that are within range of its broadcast, the neighbor that is closest to the destination should be assigned a probability equal to one, to ensure the progress of the packet towards the destination. In what follows we interchange the terms "routing cost metric" and distance.

a. a linear forwarding probability function: Initially, we consider a linear decreasing function with probability values from 1 to 0. It is simple to observe that such a function satisfies the above requirements. This forwarding probability p is expressed by:

$$p = \frac{1}{d_{\min} - d_{\max}}\left[d\left(p_{\max} - p_{\min}\right) + p_{\min}d_{\min} - p_{\max}d_{\max}\right]. \tag{1}$$

where d is the distance between the candidate forwarder and the destination, $p_{\max}$ is the forwarding probability associated to the nearest possible candidate (i.e., for $d = d_{\min}$), and $p_{\min}$ is the probability associated to the furthest possible candidate (i.e., for $d = d_{\max}$). It is straightforward to observe that $0 \le p \le 1$ for $d_{\min} \le d \le d_{\max}$.

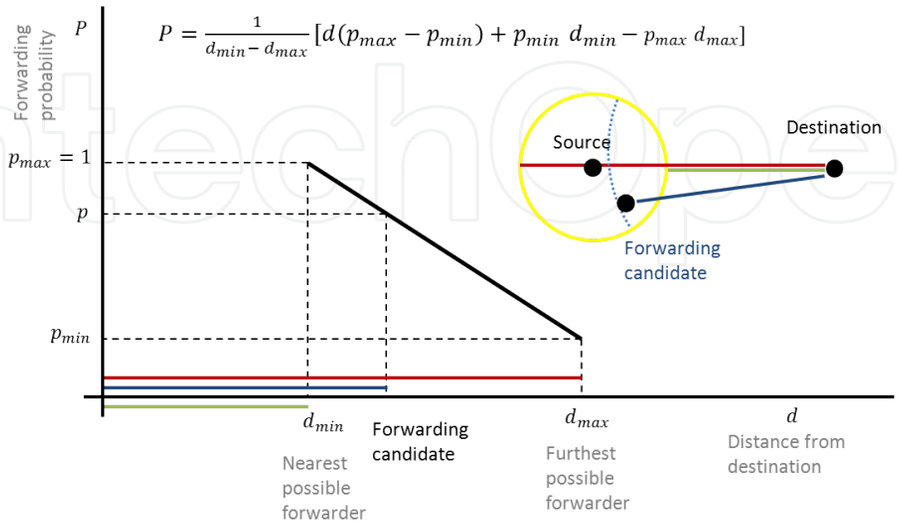

Figure 1. Forwarding probability as a linear function; inset: a node's broadcast radius.

This function provides differentiation between the forwarding probabilities of different nodes that receive the same packet in the broadcast range of the transmitter. However, it assigns a probability equal to 1 to exactly one node in each broadcast area, the node with $d_{\min}$ distance from the destination. Therefore, the packet's progress would heavily rely on that particular node. Moreover, if no node is to be found with this particular distance value in a topology, then there would be no certain forwarder for that packet in this broadcast area.

b. Piece-wise linear forwarding probability function: To deal with the case of no certain forwarders, the previous forwarding probability function can be modified to increase the number of potential forwarding candidates that can have high forwarding probability or equal to one. This can be achieved by using a piecewise function composed of an initial flat region where probability is one, followed by a decreasing linear function. The shape of the function is demonstrated in Figure 2. In this case, the forwarding probability is given by:

$$p = \min\left\{1, \frac{1}{d_{\min} - d_{\max}}\left[d\left(p_{\max} - p_{\min}\right) + p_{\min} d_{\min} - p_{\max} d_{\max}\right]\right\}. \quad (2)$$

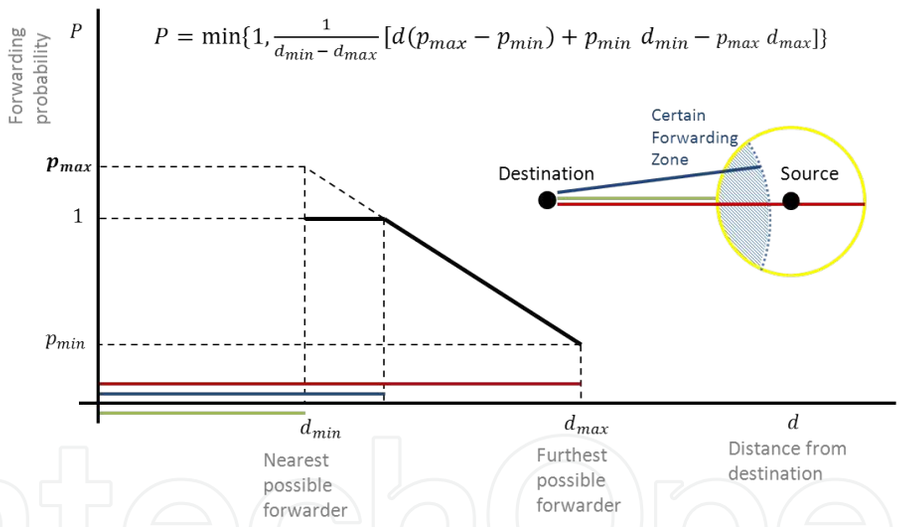

Figure 2. Piece-wise forwarding probability function and a node's broadcast radius.

This piecewise function is produced by setting $P_{\max} > 1$ and hence introducing more than one certain forwarders. To guarantee the packets' progress to the destination, at least one neighbor with forwarding probability equal to 1 is needed and this is ensured by the flat region. Ideally this would be the neighbor on the shortest path to destination.

b. Binary forwarding probability function: The final variant we examine for the forwarding probability function is a 0-1 function. Figure 3 illustrates the step-wise function. In this case, nodes are either assigned a forwarding probability equal to one or a probability equal to zero, that is they either forward the packet always or they never do. For-

warding is again based on the value of the metric and there is a threshold value of the metric, which when surpassed it is decided that the node should not forward. Yet again, the number of forwarders may be increased or decreased by adapting the threshold value of the metric. We describe this function by:

$$p = \begin{cases} 0, & d_{threshold} < d \le d_{\max}, \\ 1, & d_{\min} \le d \le d_{threshold}. \end{cases} \tag{3}$$

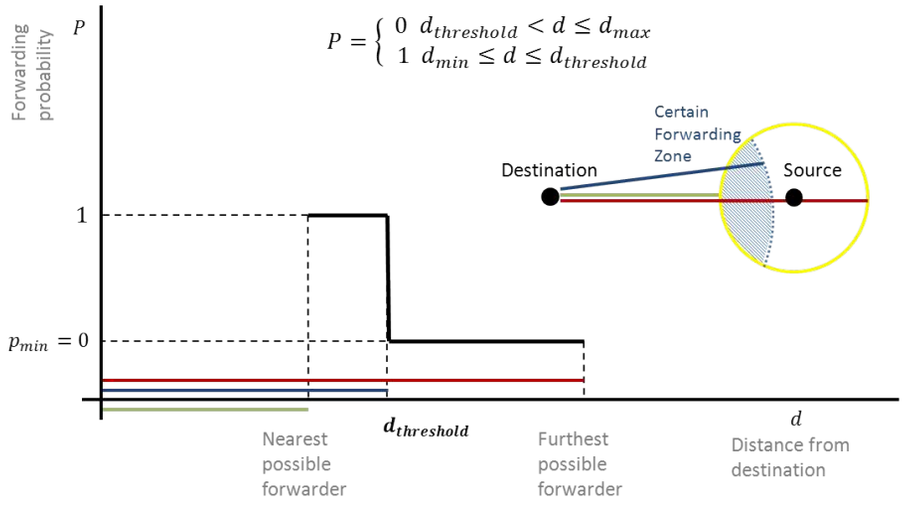

Figure 3. Step-wise forwarding probability function and a node's broadcast radius.

2.4. Back-off window differentiation

Having calculated its probability to forward a certain packet that it received, a node should proceed to decide when to do so. To this end, a randomized back-off mechanism is used, where each node calculates a window of back-off time slots and randomly selects a number of slots from that range. After this back-off timer has expired, the node will proceed to forward the packet with its predetermined probability. The focus here is to differentiate the back-off times of different nodes, focusing more on the ones with high forwarding probability. These highly-likely forwarders contribute significantly to the packet's progress towards the destination; therefore it is important to avoid collisions between their transmissions, making the packet move closer to the destination as fast as possible. To this end, back-off values are taken to be inversely proportional to a node's forwarding probability. However, the piece-wise and the step forwarding probability functions both include flat regions which can assign probability equal to 1 to more than one nodes which would result in them calculating the same back-off timers. For this purpose, the linear probability function (2) is used to calculate a base probability for each node which will then be used in order to calculate its back-off window *win*.

http://dx.doi.org/10.5772/53997

$$win = \left(win_{\max} - win_{\min}\right)\left(1 - p_{base}\right) + win_{\min}. \quad (4)$$

Where p_{base} is the base probability for that node and $win_{\min}$, $win_{\max}$ are values of minimum and maximum back-off that can be assigned, respectively.

2.5. Passive hob-by-hop acknowledgment and retransmission scheme

In order to acknowledge successful packet reception the forwarding procedure takes advantage of the broadcast nature of the wireless medium. After broadcasting a packet, a node can learn if at least one of its neighboring nodes received it by overhearing its neighbors' transmissions for a sort amount of time, thus also avoiding collisions. If a transmission of the last sent packet is overheard, then a node will drop the packet from its queue and continue to transmit the next. In case time goes by without overhearing any transmission of the last sent packet, then the node will retransmit it, as long as a maximum number of retransmissions has not been reached.

2.6. Multiple packets handling

Upon successful reception of a new non-expired packet, the node will have to calculate its forwarding probability for it, its back-off window for it and select a random back-off value from the range of the latter. A list with the packet and flow IDs of previously successfully forwarded packets can be kept to ensure that a node will not forward the same packet of a flow twice thus reducing redundant transmissions. After determining all of the above, a node will have to store individual packets according to the back-off timer that it has calculated for each of them and try to transmit them in time.

The manner in which the node will handle the various packets it has stored can be described as a system of multiple queues, each one containing packets for which the node has selected the same back-off value and each queue is a FIFO. After having successfully transmitted a packet and overheard its retransmission by a neighboring node, the node will look for the queue with the smallest back-off and pick the first packet from that. This ensures that a packet for which the node has a small back-off will have priority over one for which the node has a large back-off value. Taking into account that the back-off assignment favors the optimal forwarding candidates for a packet, this queuing policy allows the node to give priority to packets to whose progress it can contribute more.

2.7. Routing scheme adjustable parameters

a. Maximum probability ($p_{\max}$): The forwarding probability function slope defines the difference in the probability to forward the packet between the neighbors of the node currently holding the packet. Specifically, the steeper that slope is the more the neighbors closer to the destination will be favored. By setting the maximum probability to a value higher than 1, the slope of the forwarding probability function can be tilted, thus increasing the flat segment, which leads to more "certain" forwarders. This feature pro-

vides the forwarding function the ability to adapt in situations where more forwarders with high probability are needed.

b. Acknowledgment delay: After a node broadcasts a packet, it will start overhearing its neighbors' transmission in order to verify that the packet it sent has been broadcasted by one of them. The amount of time it can wait in this overhearing mode without success, until it decides it has to retransmit the packet, is called acknowledgment delay. If this interval is too small, then the node might end up retransmitting a packet that is successfully received by the further hops, thus adding one redundant transmission. On the other hand, if it is too long and no transmission is overheard, then the packet's progress will be delayed.

c. Time to live (TTL): To ensure that the packets will not circulate in the network long after they have reached the destination, a mechanism that renders them obsolete is needed. For this purpose, each packet has a fixed number of "credits" which are spent each time it is broadcast. These credits can be time units or number of hops traversed, under the assumption that a time unit equals the time it takes for the packet to move one hope further. A node that receives a packet with an expired TTL will discard it without calculating any forwarding probability or back-off window for it.

3. Simulation framework assumptions and setup

To model the behavior of our proposed opportunistic routing scheme and gain insight in its parameters in order to be able to optimize them, we developed a MATLAB-based platform as a time-driven simulator. The model we consider is time-slotted one. Specifically, a time-slot refers to the duration a fixed size packet needs to be broadcast and received. To setup the multi-hop wireless network simulation environment, we made a series of assumptions, regarding system information availability, packet propagation and channel errors.

For simulation simplicity we assumed a protocol propagation model i.e. a transmission can only be received, under some probability, by all nodes within a broadcast radius from the source. The nodes that lie within a node's broadcast radius are referred to as neighbors of that node. In our initial implementation all nodes in the network shared the same packet reception probability and same broadcast radius. Individualizing them is a straightforward programming exercise.

The simulation scenarios we include in the next section take place in a grid topology, such that a node may have four, eight or twelve neighbors, depending on the transmission radius (Figure 4). The nodes have fixed positions, known a priori, as far as calculating metric values is concerned. However, more randomized topologies are be examined as well, by setting random nodes of the uniform grid topology as inactive. From an implementation perspective, this was performed by randomly selecting nodes other than the source or destination of the packets and fixing their packet reception probability to zero.

http://dx.doi.org/10.5772/53997

For the purpose of testing the performance of the suggested procedure in realistic conditions, the assumption that there is full and accurate knowledge of nodes' locations in the topology was relaxed. Regardless of the metric used, the possibility of erroneous estimations of the metric value should be taken into consideration when testing the performance of a routing scheme, in order to examine its robustness.

To this end, a noise parameter in the metric calculation was introduced in the simulation environment. We consider that the calculated metric is taken randomly in an interval centered on the real metric value and spans by a percentile which is a simulation parameter.

Experiments were conducted on a 40 x 40 node grid topology to measure delay, loss ratio and resource consumption for varying network densities and channel error conditions. Each experiment with a given set of parameters was repeated for 100 runs and the results presented in what follows are averages over the number of runs. Delay measurements were performed on the shortest path from source to destination and a classic single-path lowest-cost route has been simulated and used as the basis of comparisons.

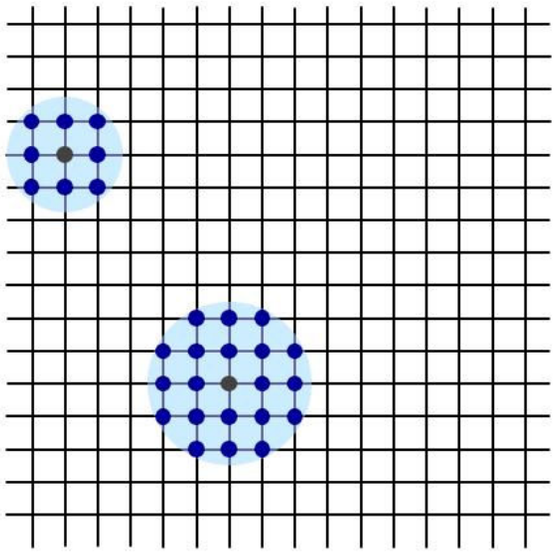

Figure 4. Grid topology with transmission radii for 8 and 12 neighbors.

4. Results

4.1. Tuning the schemes' adjustable parameters

4.1.1. The effect of the forwarding probability function

The forwarding probability function slope defines how great the difference in the probability to forward the packet will be between the neighbors of the node currently holding the

packet. Specifically, the steeper that slope is the more the neighbors closer to the destination will be favored. To ensure the packet's progressing to the destination, at least one neighbor with forwarding probability equal to 1 is needed; ideally this would be the neighbor on the shortest path to destination. This leads to a linear forwarding probability function. As the packet error ratio (PER) increases, more "certain" forwarders are needed, to make up for failed packet receptions, but the need arises to relax potential collisions among them.

By setting the maximum probability to a value higher than 1, we tilt the slope of the forwarding probability function, thus increasing the number of certain forwarders. Figure 5 illustrates how tilting the slope increases the number of certain forwarders in an 8 neighbor topology. Initially there is only one certain forwarder, the one that is closest to the destination. Two more forwarders are added that are closer to the destination than the node that transmitted the packet. If the slope is increased, two more forwarders are added, that are slightly further away from the destination, which will make the packet progress diverge sideways from the shortest path. Values of maximum probability in [1.4, 2] contribute to the packet's progress without diverging much from the shortest path. Furthermore, for a given PER value, having more than one certain forwarders yields lower delay.

This is verified by Figure 6 which illustrates how tilting the forwarding probability function's slope leads to lower delays, in the context of a 10-hop shortest path. It should be noted that only the maximum forwarding probability parameter is examined at this point; lower delay can be achieved by adjusting the maximum back-off window value as well, which in this case is variable, dependent on the forwarding probability, taking values in [1, 8]. Nonetheless, our probabilistic scheme outperforms single-path routing for PER values higher than 0.25.

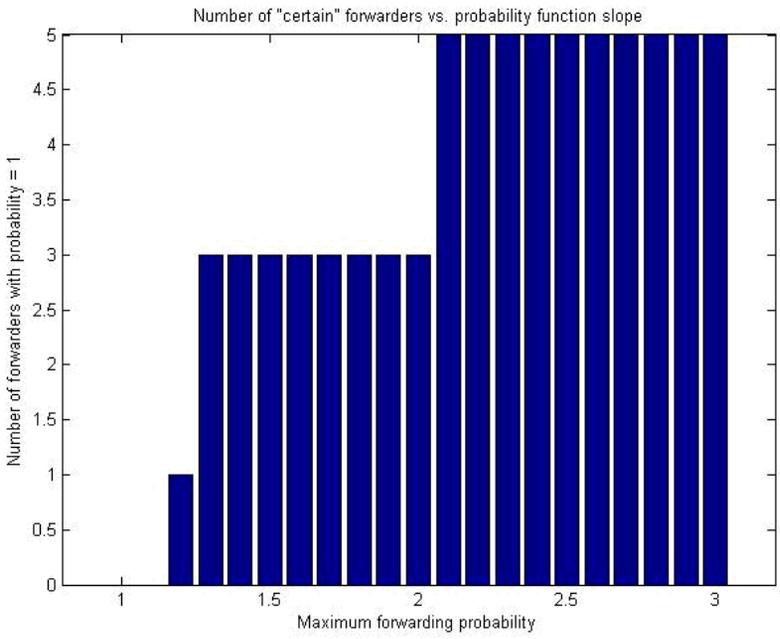

Figure 5. Increasing the number of certain forwarding nodes tilting the probability function slope.

http://dx.doi.org/10.5772/53997

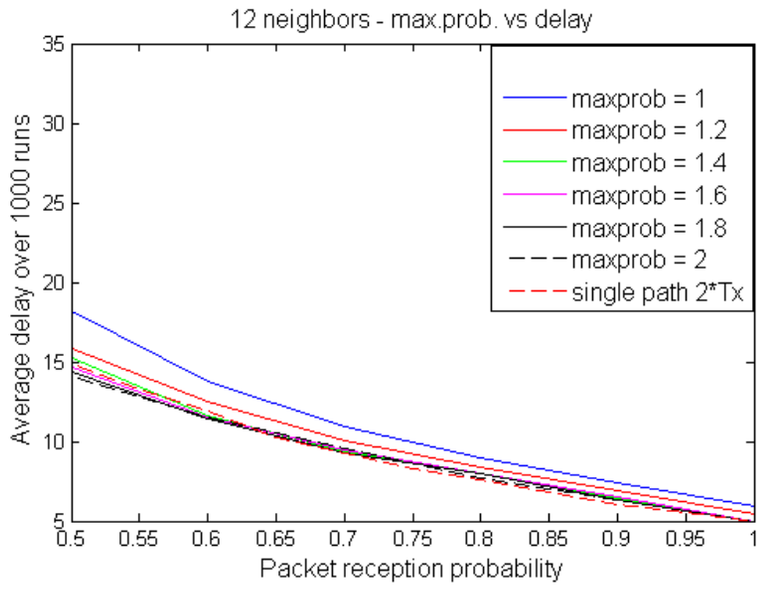

Figure 6. Tilting the slope to create a saturated, piece-wise probability function yields reduced delays.

4.1.2. Back-off window differentiation

There were two general approaches to back-off window schemes with respect to the window range values. Firstly, in the fixed back-off window option, all transmitting nodes would randomly select their back-off values from the same range of window values. Secondly, when differentiating, each transmitting node randomly selects its back-off value from a different range of numbers. Specifically, the back-off window of a node will be randomly chosen between a smaller set of numbers, the larger its forwarding probability is, so as to reduce delays. Figure 7 below illustrates the delay performance of the fixed (7a) and differentiated back-off window schemes (7b), for increasing packet error rate values, in a scenario where the source is 10 hops away from the destination. There is a steeper increase in delay for the fixed back-off window scheme as the width of the back-off window interval increases, which renders fixed back-off values larger than 2 inefficient.

It can be observed that the lowest delay is measured for a back-off window of 1, which raises the question, why differentiate between nodes at all. The reason is that back-off differentiation also yields lower resource consumption.

To capture the effect a packet's transmission has on the network, we track the footprint its transmissions produce over time on the nodes as it is forwarded towards the destination, until all transmissions cease. The times each node has received the packet are averaged over the number of the different runs of the experiment.

For all results referring to footprints hereon the source node's coordinates are (10,20) and the destination is at (20,20). Figure 8 illustrates the footprint for a flow with back-off window equal to 1. As we increase the width of the back-off window the flooding is limited to an area around the shortest path. This is shown by the plots in Figure 9. It should be noted that

a linear forwarding probability function with maximum probability equal to 2 was used for these experiments. There is a trade-off between low delay performance and resource consumption which should be addressed by having each flow's specific requirements in mind. For example, in a network where only one flow is present at a time, a back-off window set to 1 would yield the lowest delay possible, whereas in the presence of multiple flows, a more conservative back-off scheme with the window interval set to [1,8] should be used.

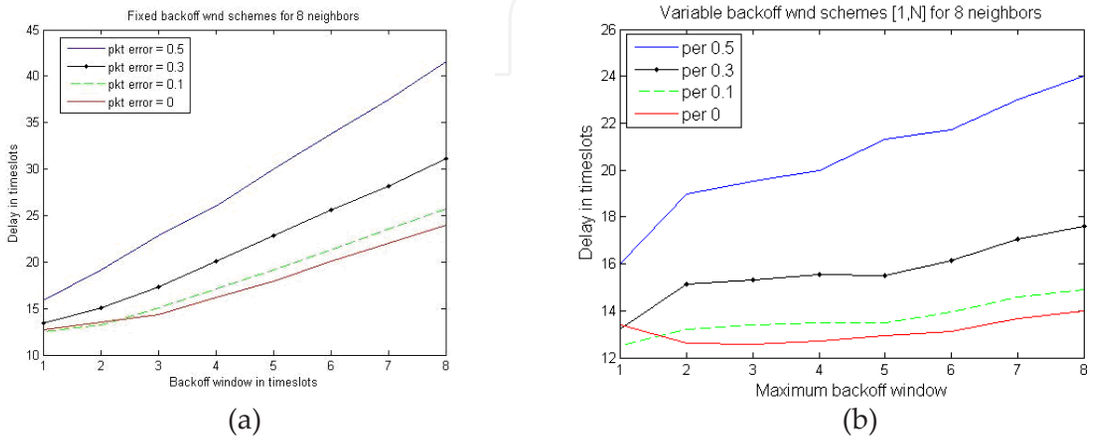

Figure 7. Delay in number of slots for a 10-hop shortest-path source-destination pair for the fixed (a) and variable (b) back-off schemes.

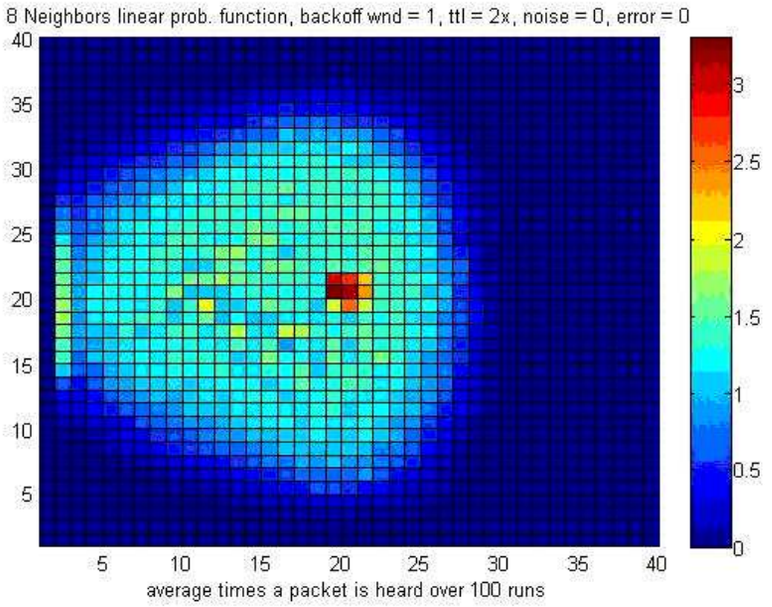

Figure 8. A back-off window of one time slot yields low delay compared to a variable backoff window but has a large footprint i.e. is resource inefficient.

http://dx.doi.org/10.5772/53997

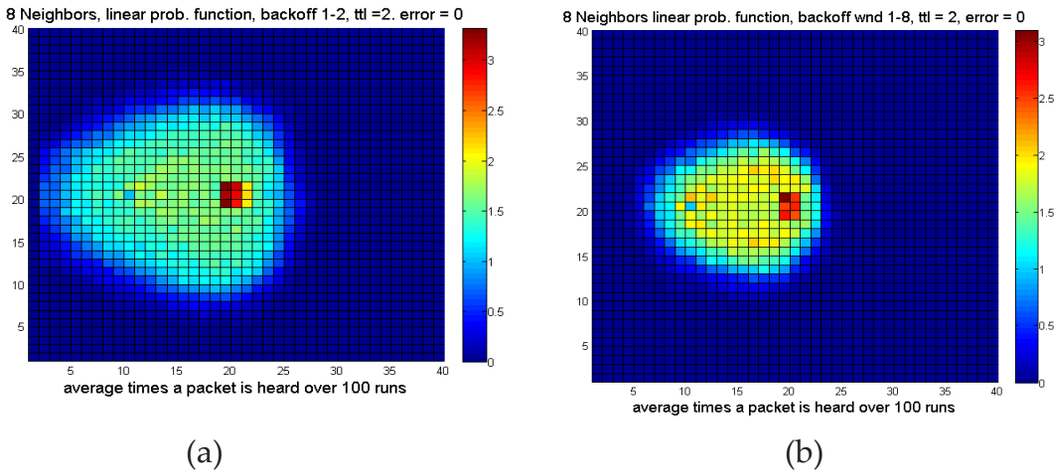

Figure 9. Increasing the back-off window range from [1,2] in (a) to [1,8] in (b) under a piece-wise probability function reduces the footprint causing less interference.

4.1.3. Metric miscalculation error

To evaluate the robustness of our scheme to mistakes in metric estimations, we examine the effect of introducing metric noise.

1. Linear probability function: To measure resource consumption for the linear probability function with maximum probability set to 2, we track the flow's course in a scenario where there is no channel error and the source is 10 hops away from the destination and use a balanced back-off window scheme in the interval [1, 4].

Figure 10 below illustrates how the linear forwarding probability scheme reacts to increasing metric miscalculation. It is noteworthy that the linear forwarding probability scheme is insensitive to metric noise up to 10% of the accurate metric's value and performs decently even at the presence of noise equal to 30% of the accurate metric's value. Having in mind that the metric values are randomly chosen from a uniform distribution in such wide intervals, it is obvious that metric noise equal to 0.3 already presents a scenario of extremely inaccurate metric estimation.

2. Step-wise probability function: We compare the performance of the linear probability function in the presence of noise to that of the step-wise probability function. For the purposes of these experiments the step was set to 0, such as the neighbors that have a smaller distance from destination, than the node currently holding the packet, will forward the packet. The source is again at (10,20) and the destination at (20,20) and the back-off scheme is again set to [1, 4] interval.

Figure 11 illustrates the step-wise function's performance under increasing metric noise conditions. For the same values of metric noise, the step-wise function consumes fewer resources than the linear function and it is also less affected by metric noise. This is justified by the fact that even when inaccurate estimations are made, only a few nodes will calculate their metric so that they fall below the threshold, whereas the majority of nodes will get the same probability as what they would get if there was no noise.

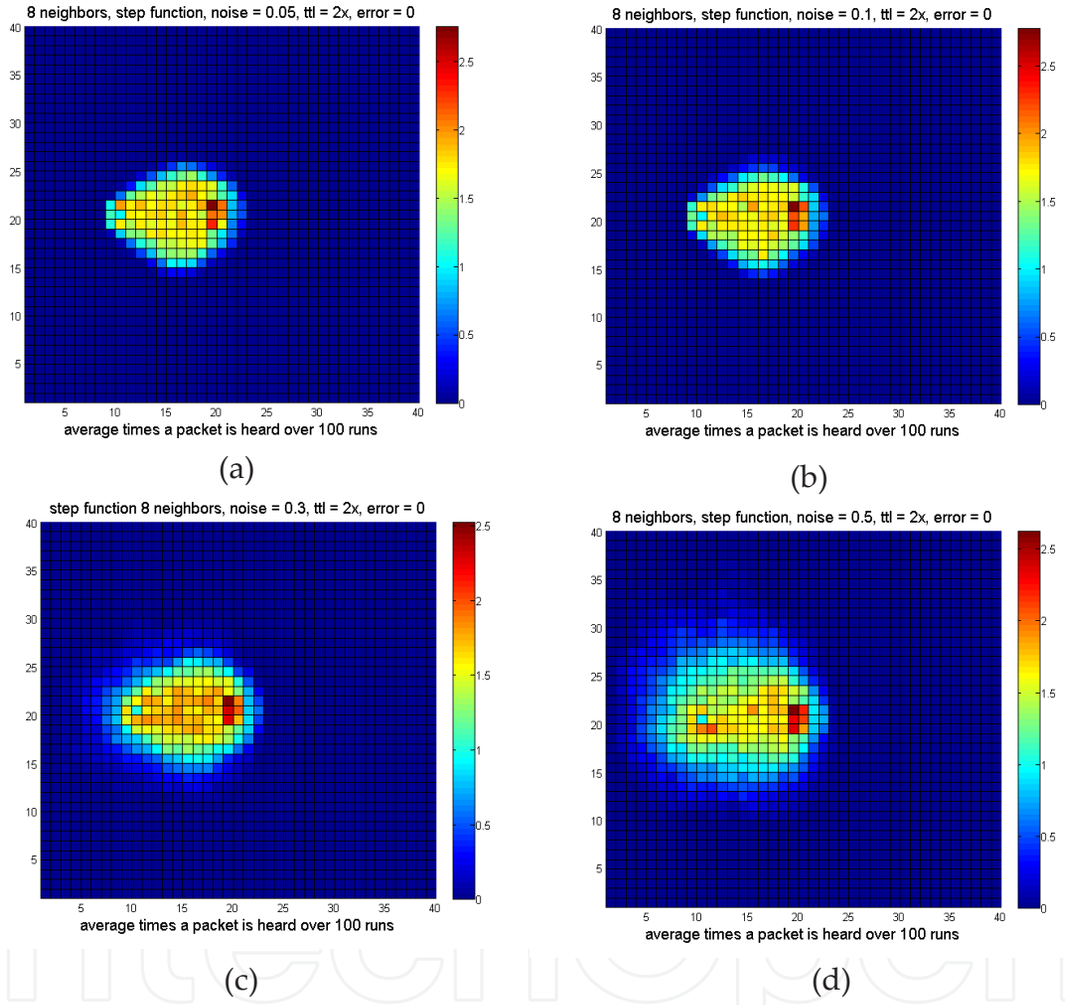

Figure 10. Increasing the metric miscalculation factor from 0.05 (a), 0.1 (b), 0.3 (c), to 0.5 (d) slightly affects the resource efficiency for an 8-neighbor topology under a piece-wise probability function.

http://dx.doi.org/10.5772/53997

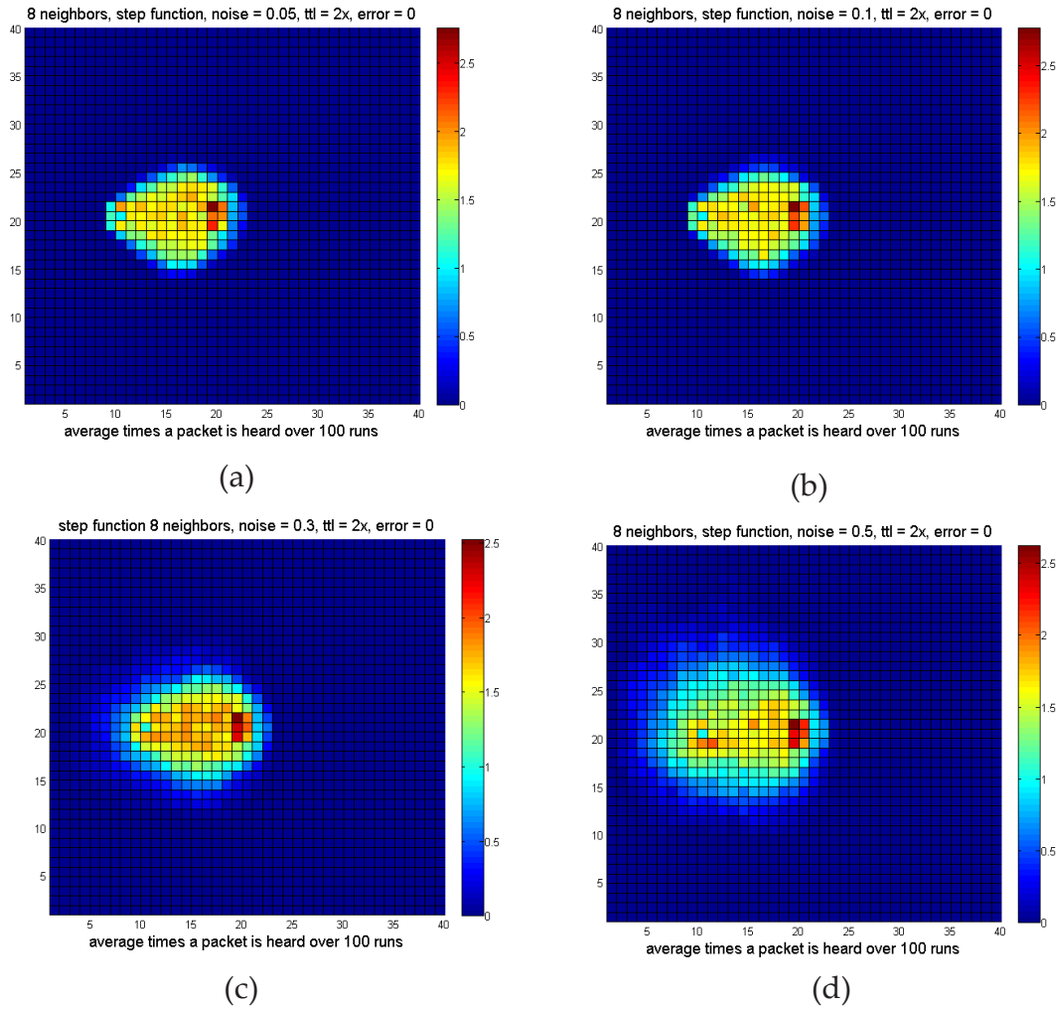

Figure 11. Increasing the metric miscalculation factor from 0.05 (a), 0.1 (b), 0.3 (c), to 0.5 (d) slightly affects the resource efficiency less than in figure 10 when using a step-wise probability function.

4.1.4. Multiple flow support

It is of interest to indicate that the routing scheme we have devised thus far is able to support the interaction of different flows in the network and examine their behavior. The scenarios in the following figures were chosen with respect to the most common cases presented in wireless mesh and sensor networks. Figure 12 illustrates the scenario of two nodes sending packets to the same destination node, whereas Figure 13 is the reverse, which can be considered as a "downlink" case. Both provide evidence that the system supports multiple flows and that the scheme behaves as expected both in terms of packet error, as well as in terms of back-off delay.

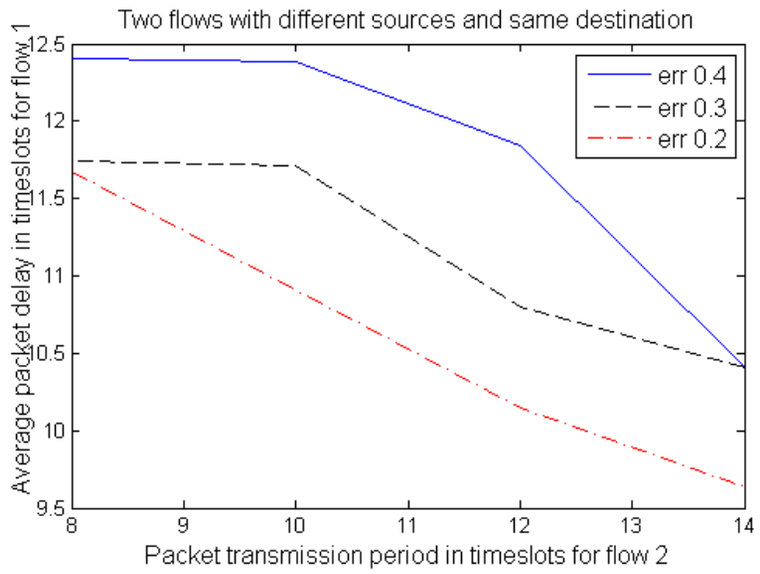

Figure 12. Delay performance of a 50 packet flow while competing with another for the same destination.

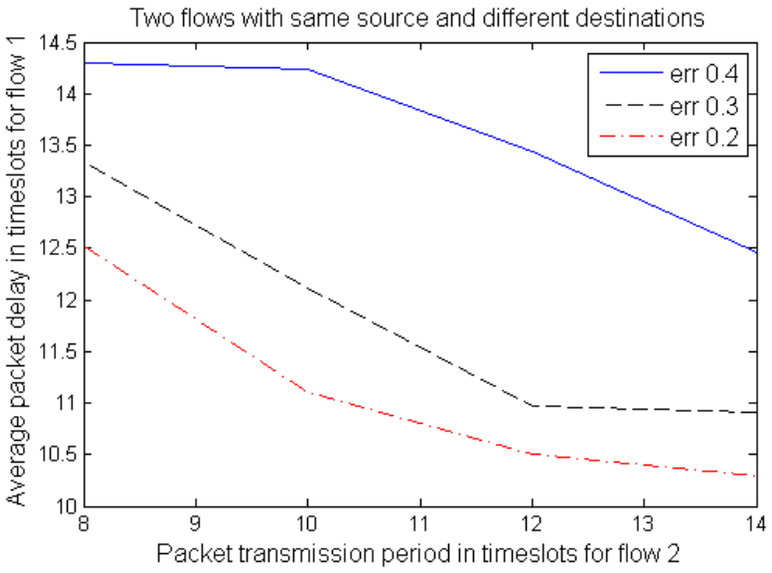

Figure 13. Delay performance of a 50 packet flow while competing with another flow from the same source.

http://dx.doi.org/10.5772/53997

4.2. Performance comparison

4.2.1. Comparing against SOAR

For our comparative simulations, SOAR's algorithm for packet forwarding decisions was used, combined with the proposed passive acknowledgment scheme, in order to test the performance of its opportunistic features. SOAR initially uses ETX as a metric in order to decide on the cost of forwarding, however, for comparison purposes, hop distance was used for both protocols. SOAR behaves similarly to shortest path routing in no-error conditions, constraining the flow along the shortest path from the source to the destination, as shown in Figure 14. When metric miscalculation is present, SOAR's delay increases signifantly, as opposed to the proposed scheme's performance which is unaffected, as shown in Figure 15. This can be explained by the quasi-deterministic forwarding scheme used by SOAR. If metric miscalculation occurs at the source (who creates the list of forwarders), the error will propagate along with the list, since it is included in any sent packets. Therefore consequent calculations based on this list will be influenced by even a single miscalculation error.

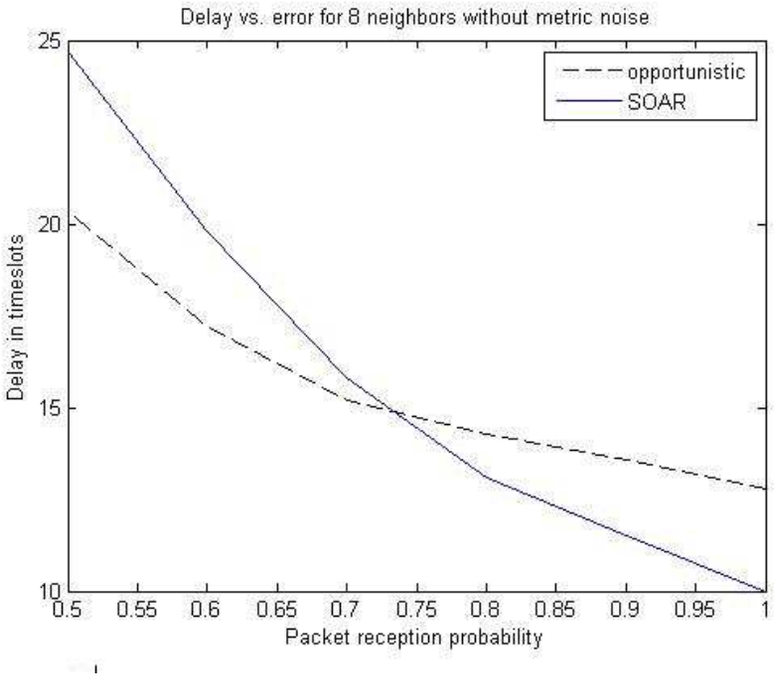

Figure 14. The proposed scheme (dashed) yields lower delays than SOAR (solid) for packet error probability over 0.25.

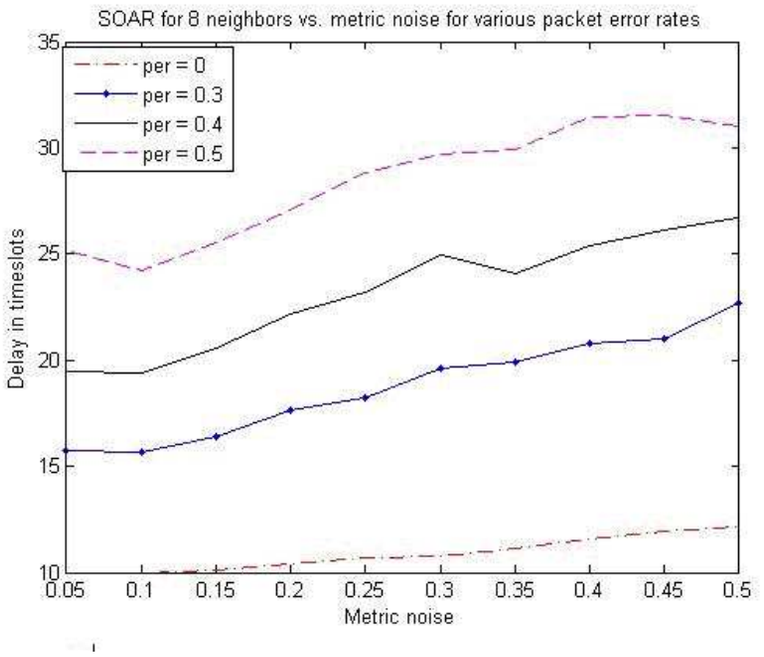

Figure 15. SOAR's delay performance degrades as metric miscalculation is increased.

4.2.2. Comparing against direct transmission

Directed Transmission is a parametric probabilistic routing protocol focuses on design simplicity, distributed routing decisions and robustness to metric miscalculation. It should be noted that Directed Transmission does not account for losses due to poor link quality, so the two protocols were compared in no-channel error scenarios, where metric miscalculation was present. When error in metric calculation increases, directed transmission's use of resources does not increase significantly, as is the case with the scheme proposed. However, the spread of the flow on the grid is comparable to its equivalent in the proposed scheme when the piece-wise probability function was used and larger than its equivalent when the step-wise probability function was used. Furthermore directed transmission has a larger average number of transmissions needed to deliver a packet along the 10 hop path, regardless of metric noise. These are depicted in Figure 16. This demonstrates that a routing protocol can be simple enough as the one we proposed and at the same time conserve resources sufficiently to be applied in WSNs without suffering from low delay performance.

http://dx.doi.org/10.5772/53997

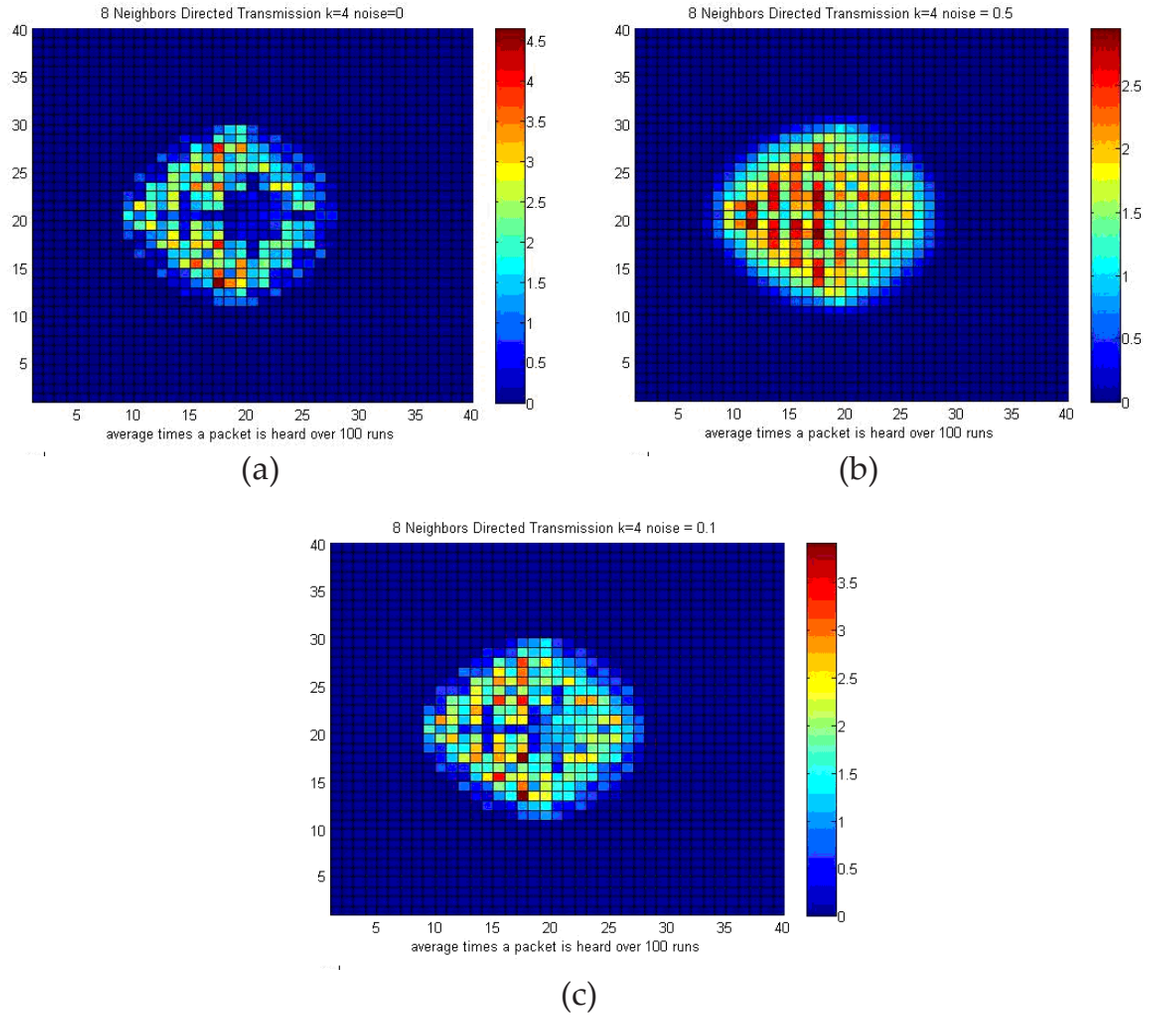

Figure 16. Directed Transmission appears equally robust to metric miscalculations as the proposed routing scheme, still compared to Figures 10 and 11 it requires on average more retransmissions to deliver a packet.

5. Conclusion

We have addressed the two key design decisions of designing an opportunistic routing scheme. How nodes should decide whether to forward or not and when is the most efficient time to do so. We gave indications using a simulation framework on how the forwarding decisions and transmission timing alone affect performance. This was done through a probabilistic forwarding scheme, whose parameters can be tuned to allow for low resource consumption and high delay performance, while being robust to misinformation. The overall routing protocol comprising the forwarding scheme along with the timing and the acknowledgement mechanism is stripped of complex routing mechanisms and so we examined which channel error conditions and topology density it is beneficial to use opportunistic routing instead of traditional routing.

Simulation results demonstrate that the suggested opportunistic scheme can outperform single path routing for error values larger than 15% -20%, for a well-selected slope of the forwarding probability function, with restrained use of resources. Furthermore, we have shown that the optimal manner of adapting to increasing error is to increase the number of forwarders by increasing the slope of the forwarding probability function. In particular, the number of certain forwarders has the most impact on performance and they need to be increased with respect to error conditions.

To reduce resource consumption, in terms of packet transmissions, utilizing a step-wise function is a sound approach, which proved robust to metric miscalculations. Finally, there is a tradeoff between differentiating each forwarder's back-off value to reduce resource consumption and reducing delay. Simulation results show that a variable back-off scheme that gives priority, by means of smaller back-off windows, to best forwarders according to their forwarding probability is preferable to a fixed back-off window for all forwarders.

Acknowledgements

Dr. Angelakis and Ms. Gazoni were with ICS-FORTH, Greece when conducting part of the research presented and acknowledge the aid of Dr. V. A. Siris and the support of the FP7 project EU-MESH.

We further acknowledge the support of the FP7-People-2007-3-1-IAPP-218309 project. This work has been also partially supported by CENIIT at Linköping University, the Swedish excellence center ELLIIT, and the Swedish Research Council (VR) through project B0581701.

We finally thank Mr. Lei Lei for his help in the final revision of the material.

Author details

Vangelis Angelakis[1], Niki Gazoni[2] and Di Yuan[1]

1 Department of Science and Technology, Linköping University, Norrköping, Sweden

2 Forthnet S.A., Systems Engineering and Design Department, Athens, Greece

References

[1] C. L. Barrett, S. J. Eidenbenz, L. Kroc, M. Marathe, and J. P. Smith. Parametric probabilistic sensor network routing. In proc. ACM WSNA '03, doi 10.1145/ 941350.941368, 2003.

[2] S. Biswas and R. Morris. Opportunistic routing in multi-hop wireless networks. In proc. SIGCOMM05, doi: 10.1145/972374.972387, 2005.

[3] J. Du, H. Liu, and P. Chen. Omr: An opportunistic multi-path reliable routing protocol in wireless sensor networks. In proc. IEEE ICPPW '07, doi: 10.1109/ICPPW. 2007.61, 2007.

[4] C. E. Perkins and P. Bhagwat (1994). Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers: ACM SIGCOMM Computer Communication Review, doi: 10.1145/190809.190336.

[5] N. Gazoni, V. Angelakis, V. Siris, and R. Bruno. A framework for opportunistic routing in multi-hop wireless networks. In proc. ACM PE-WASUN, doi: 10.1145/1868589.1868600, 2010.

[6] S. Jain and S. Das. Exploiting Path Diversity in the Link Layer in Wireless Ad Hoc Networks, in proc. IEEE WoWMoM '05, doi: 10.1016/j.adhoc.2007.07.002, 2005.

[7] D. B. Johnson, D. A. Maltz, and J. Broch (2001). Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks. In: Charles E. Perkins editor, Ad Hoc Networking, Addison-Wesley. pp. 139-172.

[8] De Couto D. S. J, Morris R., Aguayo D., and Bicket J. A high-throughput path metric for multi-hop wireless routing. In proc. ACM MobiCom '03, doi: 10.1145/938985.939000, 2003.

[9] Z. J. Haas, J. Y. Halpern, and L. Li (2006). Gossip-based ad hoc routing. IEEE/ACM Transactions on Networking (TON) doi:10.1109/TNET.2006.876186.

[10] L. Haitao, et al. (2009). Opportunistic routing for wireless ad hoc and sensor networks: Present and future directions. IEEE Communications Magazine, doi: 10.1109/ MCOM.2009.5350376.

[11] S. Mueller, R. Tsang, and D. Ghosal. (2004) Multipath routing in mobile ad hoc networks: Issues and challenges. In M. C. Calzarossa, E. Gelenbe, editors, Performance Tools and Applications to Networked Systems, Springer-Verlag pp. 209-234.

[12] M. S. Nassr et al. Scalable and Reliable Sensor Network Routing: Performance Study from Field Deployment. In proc. IEEE INFOCOM '07, doi: 10.1109/INFCOM.2007.84, 2007

[13] R. Rajaraman, (2002) Topology control and routing in ad hoc networks: A survey. ACM SIGACT News, doi>10.1145/564585.564602.

[14] E. M. Royer and C.E. Perkins. An implementation study of the aodv routing protocol. In proc IEEE WCNC, doi: 10.1109/WCNC.2000.904764, 2000.

[15] E. Rozner, J. Seshadri, Y. Mehta, and L. Qiu. SOAR: Simple Opportunistic Adaptive Routing Protocol for Wireless Mesh Networks. In IEEE Trans. On Mobile Computing 8(12): 1622-1635, 2009.

[16] R.C. Shah, S. Wietholter, A. Wolisz, J. M. Rabaey. When does opportunistic routing make sense? In Proc. IEEE Percom, doi: 10.1109/PERCOMW.2005.90, 2005.

[17] J. Wu, M. Lu, and F. Li. Utility-Based Opportunistic Routing in Multi-Hop Wireless Networks," Proc. IEEE ICDCS '08, doi: 10.1109/ICDCS.2008.90.

[18] Y. Yuan, H. Yang, S.Wong, S. Lu, and W. Arbaugh. Romer: Resilient opportunistic mesh routing for wireless mesh networks. In proc IEEE WiMesh workshop 2005, SECON, pp. 146-158, 2005.

[19] K. Zeng, W. Lou, and H. Zhai. On end-to-end throughput of opportunistic routing in multirate and multihop wireless networks. In proc. IEEE INFOCOM '08, doi: 10.1109/INFOCOM.2008.133 2008.

[20] Z. Zhong and S. Nelakuditi. On the efficacy of opportunistic routing. In SECON '07: In proceedings of the 4th IEEE International Conference on Sensor and Ad Hoc Communications and Networks, doi: 10.1109/SAHCN.2007.4292856, 2007.

[21] S. Chachulski et al., Trading Structure for Randomness in Wireless Opportunistic Routing. In proc. ACM SIGCOMM '07, doi: 10.1145/1282427.1282400 2007.

[22] P. Jacquet, T. Clausen, (2003) Optimized Link State Routing Protocol (OLSR), RFC 3626

[23] Y. Yan et al. Practical Coding-Aware Opportunistic Routing Mechanism for Wireless Mesh Networks. In proc. IEEE ICC2008 doi: 10.1109/ICC.2008.541,.2008.

[24] Z. Zhong et al., (2006) On Selection of Candidates for Opportunistic AnyPath Forwarding. ACM SIGMOBILE Mobile Computing and Communications Review doi>10.1145/1215976.1215978.