Mazumdar and Sairam *EURASIP Journal on Wireless Communications and Networking* 2013, **2013**:237
http://jwcn.eurasipjournals.com/content/2013/1/237

EURASIP Journal on
Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

TOAR: transmission-aware opportunistic *ad hoc* routing protocol

Arka Prokash Mazumdar and Ashok Singh Sairam*

Abstract

Opportunistic routing (OR) protocols for *ad hoc* networks basically consist of selecting a few forwarders between the source and destination and prioritizing their transmission. The performance of OR protocols depends on how these two steps are performed. The aim was to reduce the number of transmissions to deliver packets to the destination. In this paper, we first present a mathematical model to compute the total number of packets including duplicate packets generated by OR protocols. We use the model to analyse well-known OR protocols and understand the reason behind their increase in number of transmissions. Next, we propose an OR scheme *transmission-aware opportunistic ad hoc routing* (TOAR) protocol, which attempts to minimize retransmissions. Our proposed OR protocol uses tree structures to select forwarders and prioritize them. The use of tree structures helps in identifying *primary* forwarders which carry packets farthest to the destination during each transmission round. TOAR also helps in choosing *secondary* forwarders which will transmit packets missed out by the forwarder. The optimized selection of forwarders results in significant reduction in retransmissions, a smaller forwarder list set, and improvement in goodput.

1 Introduction

Ad hoc wireless networks are important communication techniques where wireless infrastructures cannot be deployed. This class of wireless network does not have any centralized control or fixed infrastructure, instead the nodes interconnect with each other through multihop routing paths. All the nodes have similar privileges, and they collectively participate in the routing procedures. A great number of wireless routing protocols based on traditional routing techniques [1-4] are proposed in the literature. Although these protocols attempt to optimize different metrics, in general, they select a path from source to destination prior to injecting packets to the network and then forward data through that predetermined route. The main characteristic of the wireless network is that it is broadcast in nature, i.e., all the nodes in the range of the transmitting node are able to listen to the packet even if the packet is not meant for them. OR protocols are designed to exploit this broadcast nature of wireless networks. In OR protocols, the source node rather than choosing a single path to the destination selects a number of potential forwarders between the source-destination pair and broadcasts the packet. Among the selected forwarders that received the packet successfully, the node closest to the destination is given the first chance to forward the packet. This procedure continues until the packet reaches the destination. It has already been demonstrated in literature that OR protocols produce significantly better results than traditional wireless protocols [5].

1.1 Problem outline and motivation

The two main steps in an OR protocol are (1) selection of the forwarder list and (2) prioritization among these forwarders. The efficiency of an OR protocol mainly depends upon these two steps. There may be numerous nodes available between the source and the destination. The job of the selection procedure is to decide which subset of the nodes will participate in the routing process that will lead to optimal result. One major issue with the selection process is the generation of duplicate packets due to improper selection strategy. The main goal of this paper was to understand the cause of duplicate transmissions and then propose a forwarder selection strategy that addresses the problem.

*Correspondence: ashok@iitp.ac.in
Department of Computer Science and Engineering, Indian Institute of Technology Patna, Patna 800013, India

The motivation behind studying duplicate transmissions in *ad hoc* OR protocols can be justified as follows. Most of the OR protocols available in literature use different forwarder selection criteria in an attempt to reduce retransmissions. Classical OR protocols such as ExOR [6] run a simulation and select those nodes as forwarders that can forward a certain percentage of packets. The more recent protocols select subsets of relays, compute the expected number of retransmissions for each subset, and choose the one that minimizes transmissions. However, none of these works did an in-depth study to find the cause of retransmissions. These protocols primarily concentrate on the global goal of reducing transmissions but do not attempt to understand the dynamics between the relays within a subset.

1.2 Contribution

In this paper, we systematically analyse the number of retransmissions that occur in OR protocols. This paper makes the following contributions. We first propose a mathematical model to analyse OR protocols in terms of the number of transmissions. The model considers the probability of duplicate packet generations. Using this model, we then analyse well-known OR protocols and try to understand the reason behind the generation of duplicate packets. Through quantitative analysis, we show that the main reason behind retransmissions is because adjacent forwarders are not in the range of one another. Finally, using this knowledge, we try to select forwarders from each hop distance to the destination, such that the communication between forwarders are more reliable as well as the set of forwarders reduce the overall number of transmissions. In particular, we show that there is at least 20% decrease in the number of retransmissions which subsequently results in improvement of goodput.

1.3 Paper outline

The rest of the paper is organized as follows. Section 2 presents a survey of related work. In Section 3, we discuss how duplicate packets can occur due to improper selection of forwarders. A mathematical model to compute transmissions in OR protocols is given in Section 4. Problem statement and our proposed OR protocol TOAR are given in Sections 5 and 6, respectively. In Section 7, the simulation results are presented. Finally, the concluding remarks are given in Section 8.

2 Related works

The first OR protocol was proposed by Biswas and Morris [6]. The protocol, named Extreme Opportunistic Routing (ExOR) protocol, was proposed for wireless mesh networks. ExOR selects the forwarders based on simulation with available nodes prioritized using a metric expected transmission cost (ETX) [7]. This metric computes the average number of retransmissions needed to forward a packet over a link using the packet delivery ratio of the link. ETX of a node in ExOR can be considered as the logical distance of the node from destination that is used to prioritize the nodes according to their positions. ExOR does not consider the link quality between adjacent forwarders which may result in these forwarders appearing on disjoint paths to the destination.

Another protocol MORE [8], which was designed as an improvement over ExOR, uses linear network coding for spatial reuse. This protocol, as it uses linear network coding technique, does not require strict coordination among the forwarders. However, the destination requires information regarding the number of packets encoded by the sender for successful decoding. This protocol also inherits the problems of ExOR.

Other OR protocols like EEOR [9] and OPRENU [10] use OR structures similar to ExOR but different routing metrics. GeRaF [11] is a location-based protocol where the source node uses the destination geographical location information to forward packets. It does not deterministically assign priorities to the forwarders, but at the time of transmission, the next higher priority node is chosen according to their location information. The problem with this approach is the protocol needs extra hardware and service supports to get the location information for routing.

Mao et al. proposed a global approach in their routing protocol EEOR [9], which selects route that is expected to use lowest energy among all the routes between source and destination to deliver packets. Kim et al. proposed a local approach in ORTR [12] for wireless sensor networks to deliver data under time constraints and efficient power consumption.

Zhong et al. proposed another metric, expected any-path transmission (EAX), that computes the expected number of transmissions to reach the destination given a certain ordered set of forwarders. This metric requires packet delivery ratios (PDR) between all pair of nodes, where these PDRs are assumed to be mutually independent. However, as the metric computes ETX for all possible candidate sets, the cost is high. Choice of the metric, being an important issue to select the forwarders, there is a challenge to find a metric or a set of metrics for the purpose for a given network. Some other metrics proposed for OR protocols are geographical location [11], remaining power level (RPL) [12], etc.

Resilient opportunistic routing protocol [13], proposed by Yuan et al., is a credit-based OR protocol for wireless mesh networks. There approach builds a forwarding mesh on the fly around the minimum cost path. Each data packet carries some extra credit, over the minimum cost from source to destination, so that the packet can afford to travel more paths. Upon receiving a packet, each candidate node forwards the packet, only if the node satisfies a threshold credit/cost ratio. The mesh of candidates from source to the destination is controlled by adjusting the extra credit assigned per packet. Forwarding the packets independently and building a forwarder set for each packet will incur high overheads [1]. The second issue with this approach is that the authors have not clearly mentioned how to assign credits to the packets, which is clearly a non-trivial issue as effective candidate set selection depends upon it. It is also not clear how the forwarding nodes coordinate among themselves or how the ACKs are propagated among the forwarders.

In 2009, Rozner et al. proposed another OR protocol SOAR [14] that tries to avoid duplicate transmissions. In this protocol, an intermediate node selects its next hop forwarders such that the cost from each of the nodes in this set to the destination is lower than that of the selector's. Moreover, the protocol ensures that each node is in the range of at least one node in the default (shortest) path. The protocol also bounds the number of forwarders to be selected. The protocol has some serious flaws. As the protocol follows the shortest path while building the candidate list, it might fail in scenarios where node availability around the shortest path is scarce. Another problem with the approach is it uses simple ETX as the metric to select the forwarders list. While ETX is a single-path metric, using this directly to find weights of the nodes will not reflect the improvements due to multiple next hop forwarders.

The work which is most related to ours is the Least-Cost Anypath Routing (LCAR) proposed by Dubois-Ferrire et al. [15]. This protocol chooses anypath and not the shortest path, in order to reduce retransmissions. A node selects its forwarder list (selector node) by considering all possible subsets of its neighbor set and then compute the relay cost from each of the subset to the destination. The total cost, from the selector node to the destination through a subset, is the sum of the relay set cost and the cost to the set from the selector. The subset which results in the minimum total cost of the selector is selected as its relay set. As for each selection, nodes in the relay set will have lower weights to the destination than the intermediate node; the method assures that there are no cycles. The main issue with this approach is that while selecting the relay set, it does not consider the connectivity between the member nodes of the set. They assume that the nodes in the neighbor set are in the range of one another which may not be always true. In such a case, it will result in divergent path which will defeat the basic aim. The second drawback of this approach is that while selecting the relay set, LCAR considers all possible combinations of the neighbour set. For n neighbors, the complexity of their approach will be $O(2^n)$ [16], whereas our approach has quadratic complexity. Finally as LCAR does not put a limit on the maximum number of nodes in the relay set, it may end up in selecting unnecessary large sets.

In terms of modeling OR protocols, very few work are present in literature. The authors in [17] proposed a model for OR protocols to compare them in terms of the number of retransmissions. Their model is based on discrete-time Markov chain (DTMC). The model assumes that acknowledgments for successful packet receptions are propagated to lower priority forwarders without any failure. As a consequence, their model omits duplicate packets generated during transmission. However, such an assumption is impractical, as the main problem with most OR protocols is duplicate transmissions due to failure in acknowledgment propagation.

3 Retransmissions in OR

The main goal of OR protocols is to decrease the number of transmissions needed to deliver packets to the destination. The imminent approach taken by OR for this motivation is to advance packets as close to the destination as possible in each retransmission, by exploiting all possible paths through the selected forwarders. In OR, packets are transmitted in batches. OR uses *source routing* to deliver packets to the destination. The source includes the list of forwarders ordered by their distance to the destination. The OR packet header format [18,19]) is shown in Figure 1. *Forwarding Set* is the list of forwarders. These forwarders buffer received packets.

At the end of the batch transmission, the highest priority forwarder broadcasts the buffered packets. OR protocols use a mechanism similar to *passive acknowledgments* to avoid individual acknowledgments and suppress duplicate retransmissions.

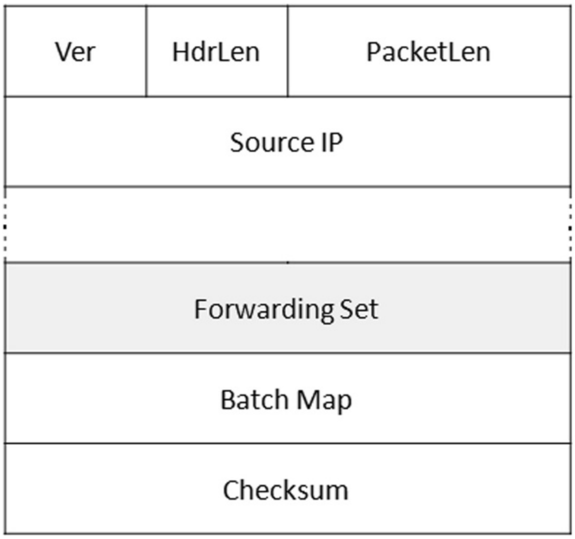


Figure 1 OR packet header format. The figure shows the various field of an OR packet.

3.1 Batch map

As can be seen in Figure 1, each OR packet contain in its header, acknowledgments called batch map [18] for all packets in the batch. The aim of the batch map is to avoid sending individual acknowledgment per packet. The batch map entries indicate, for each individual packet, the last known highest priority forwarder(as visualized by the sender) that has buffered the packet successfully. The number of entries in the batch map, therefore, is equal to the batch size. Note that each entry in the batch map, the forwarder is encoded with its id, the position of the node in the *Forwarding Set* (refer to Figure 1). This optimization makes it feasible to embed a complete batch map in every OR packet. The space required for a 100-packet batch with 16 forwarders is $\log_2(16) * 100$ or 50 bytes.

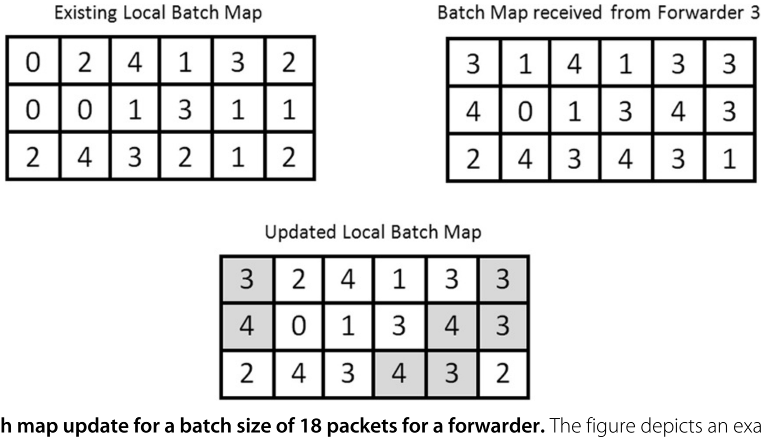
To keep track of packets, each forwarder keeps a copy of the batch map in its local memory. Whenever a forwarder receives OR packets, the batch map is updated. The batch map retrieved from the received OR packet's header is compared with the node's local batch map. If an entry in the local batch map shows a lower value, the value is updated with the one suggested by the received packet's batch map. This way, the local batch map indicates the highest known forwarders for each packet.

While forwarding packets, the node only forwards those buffered packets which were not acknowledged by higher priority forwarders. The local batch map of the node is inserted in the transmitted packets. Thus, a single transmitted packet from a forwarder carries entire information about packet positions gathered by that node.

Figure 2 shows an example of updating batch map for a batch size of 18 packets. The node receives a packet from the third forwarder. In this example, the value of the first entry in the local batch map is 0, whereas it is 3 in the received batch map. Therefore, the local batch map is updated with the value 3. There are a total of nine entries where the received batch map differs from the local one. However, for the 2nd and 18th entries, the value in the received batch map is smaller than the local one. This means that forwarder 3 has older information regarding those two packets, so the batch map is not updated for these two entries. All the other seven entries are updated in the local batch map, and they are shown in gray in the figure. It may be noted that, though the batch map came from forwarder 3, some updates show the packets buffered by forwarder 4. These updates are passively received from node 4 via node 3, as node 3 integrates the updates received from node 4 upon receiving a batch map from the node.

3.2 OR transmissions

The packet transmission process in OR is shown in Figure 3. The figure shows a forwarder list with linearly positioned forwarders v_0 to v_n prioritized according to their positions with v_n, the destination node, being the highest priority node. In OR, a forwarder will ideally transmit before any of the lower priority forwarders. Since each packet contains a batch map, this means that a node will come to know of the packets that have been buffered by nodes with higher priority. In our example when node v_0 transmits, v_3 will be the highest priority node able to receive those packets. After v_0 completes, v_3 will start retransmitting while other lower priority nodes, like v_2 and v_1, will try to listen to these transmissions to get the current packet positions. When node v_2's turn comes, it will not retransmit the packets already buffered by v_3, as it gathers the information about node v_3 by listening to its transmissions. In v_1's turn, it will not transmit the packets buffered by v_2 or v_3, even if it does not receive any packet from v_3 as node v_2's packets already include the information of node v_3.



Existing Local Batch Map

0	2	4	1	3	2
0	0	1	3	1	1
2	4	3	2	1	2

Batch Map received from Forwarder 3

3	1	4	1	3	3
4	0	1	3	4	3
2	4	3	4	3	1

Updated Local Batch Map

3	2	4	1	3	3
4	0	1	3	4	3
2	4	3	4	3	2

Figure 2 Example of a batch map update for a batch size of 18 packets for a forwarder. The figure depicts an example of a batch map updating procedure to demonstrate how it works.

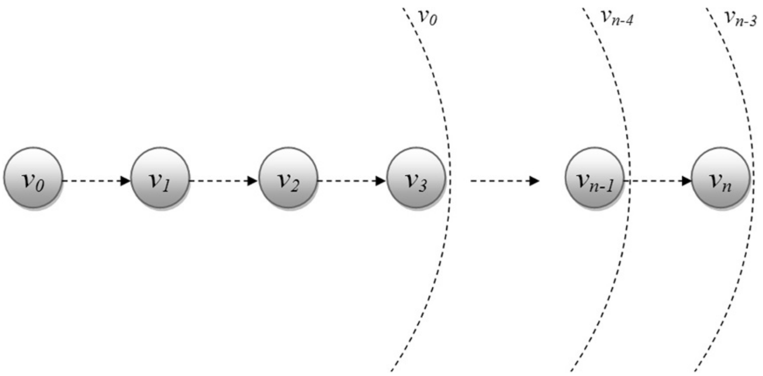


Figure 3 Ideal forwarder positions from node v_0 to v_n. The figure shows nodes v_0 to v_n linearly positioned according to their priorities, where nodes v_0 and v_n have the lowest and the highest priorities, respectively. The dotted arcs show the range of a certain node, for example, dotted arc labeled v_0 denotes the transmission range of node v_0.

This example show an ideal scenario, where all forwarders with successive priority are in the range of one another, i.e., they can listen to some or all the packets transmitted by next higher and lower priority forwarder nodes. But this may not be true, for real network scenarios where successive forwarders may fall in divergent path. For example in Figure 4, we assume a scenario where v_0 is the source and v_7 is the destination node. The figure also shows all the possible paths to the destination v_7. Let us assume an OR protocol similar to ExOR [6,18,20] that consider hop count and packet delivery probability together to select and prioritize the forwarders. In the scenario, there are two shortest paths, they are $\{v_0, v_3, v_6, v_7\}$ and $\{v_0, v_2, v_5, v_7\}$. The protocol, using the above selection criteria, will select both the paths and prepare a forwarders list $\{v_0, v_2, v_3, v_5, v_6, v_7\}$, where the nodes are prioritized by logical distances to the destination using PDRs. Here, the node pairs $\{v_2, v_3\}$, $\{v_3, v_5\}$, and $\{v_5, v_6\}$ in spite of being adjacent priority nodes, they are not in range of each other. The direct effect of this situation is that for each pair of nodes, the lower priority node may transmit packets just transmitted by the higher priority node, as the lower priority forwarder is not able to get updated batch map from the adjacent higher priority nodes.

4 Modeling retransmissions in OR

In this section, we present an analytical model to compute expected number of retransmissions for a given source destination pair. Such a model will allow comparison of OR protocols. In the recent past, there have been some attempts to model the OR paradigm and compute the expected number of transmissions.

In [17], the authors present a *discrete-time Markov chain* (DTMC) model that represents a discrete phase-type distribution in which the last state is strictly an absorbing state. In the proposed Markov model, each state represents a forwarder node. The states are sequenced as the positions of the forwarders in the ordered list, with destination being the absorbing state. The model, given the strict forwarder priorities and the packet delivery

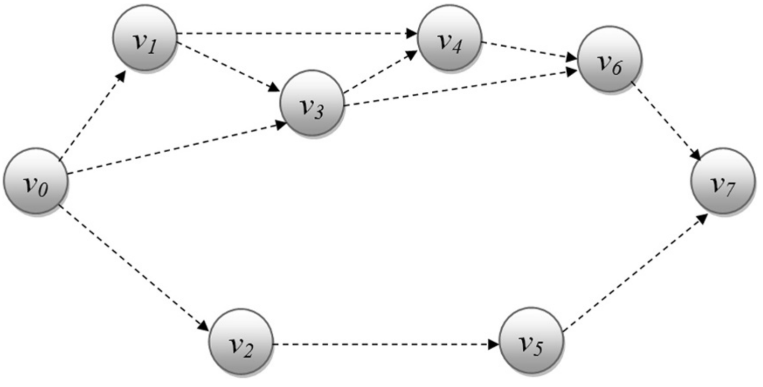


Figure 4 Simple network with all possible paths from node v_0 to v_7. The figure displays all possible paths from source node v_0 to destination node v_7. The dashed lines denote the existing links among the nodes that construct various paths to the destination.

probabilities, can compute retransmissions for a certain forwarders list using the following recursive equation:

$$E[X_i] = \frac{1 + \sum_{l=1}^{n_i} p_{il} E[X_l]}{1 - p_{ii}}, i \neq d. \tag{1}$$

Here, X_i is a random variable that represent the number of transitions from state i till total absorption occurs. The variable p_{il} denotes the probability of transition from state i to state l, with p_{ii} being the probability of remaining in the same state i. The above equation is applicable to all the states except the absorbing state (destination node). As the destination node does not transmit any data packet, the absorbing state holds the equation $E[X_d] = 0$. The main problem with this model is that, it assumes that all the nodes have knowledge of the packets received by the higher priority nodes, i.e., it does not consider duplicate retransmissions. Duplicate retransmissions occur in OR due to the failure in receiving updated acknowledgments by the lower priority forwarders. This mainly happens when the forwarder set is not strategically chosen. A lower priority node does not overhear transmission from the next higher priority forwarder.

In this section, we formulate an analytical framework for comparing OR protocols. In the remaining part of the paper, we call our model the binomial model, since it is mainly based on binomial distribution. The framework computes the expected number of retransmissions, including number of duplicate packets, given the prioritized forwarders list and the packet delivery probabilities between the forwarders.

4.1 The binomial model

Let $\overrightarrow{V} = \{v_0, v_1, v_2, \ldots, v_i, \ldots, v_m\}$ be a ordered set of forwarder list with v_0, the source node and v_m, the destination node, being the lowest and the highest priority forwarders, respectively. Let a_i^j denote the PDR between two nodes v_i and v_j. To make the mathematical model simple, we assume that all the links are symmetrical that means $a_i^j = a_j^i$. Moreover, we assume that the PDRs of the links are constant during the period of delivering the whole batch of packets. It is worth mentioning that the end result of using different path loss or shadowing models to represent a channel is difference in packet losses. The ratio of received and dropped packets reflects the effects of channel properties of the underlying channel. In this work, rather than modeling the channel directly, we use PDRs to estimate the channel qualities. Since PDRs can be easily measured, results of the mathematical models can be validated in a realistic as well as simulated environment.

Suppose $n(v_i)$ denote the number of packets received and $\delta(v_i)$ is the number of packets to be transmitted by node v_i. The number of packets to be transmitted by a node is the difference between the number of packets it received from lower priority nodes and the number of packets for which acknowledgments are received from the higher priority nodes. Let $ack(v_i)$ denote the number of acknowledgments received by node v_i. We can define $\delta(v_i)$ as

$$\delta(v_i) = n(v_i) - ack(v_i). \tag{2}$$

In opportunistic routing, first turn to retransmit is given to the highest priority node. After this node finishes its transmission, the lower priority nodes are given chance one after another, on the basis of their priority with the source node being the last node to retransmit. This process is repeated till all packets are delivered to the destination node. We define *round* as the retransmission period starting from the highest priority forwarder to the end of transmission of the source node. To send a batch of packets, one or more number of rounds of retransmissions may be needed.

Let X_{ij} be a random variable, to denote the expected number of packets received by node v_j from the lower priority node v_i with PDR a_i^j. The number packets successfully received (X_{ij}) may be anything from 0 to $\delta(v_i)$. We use binomial distribution [21] to derive the expected number of packets received by node v_j from node v_i. The binomial distribution is the discrete probability distribution of the number of successes in a sequence of n independent yes/no experiments, each of which yields success with probability a. In our case, each packet transmitted by v_i is an experiment. It is a success with a probability a_i^j, if v_j receives a packet successfully. Thus, the expected number of packets received by node v_j from node v_i can be calculated as

$$E[X_{ij}] = \sum_{k=1}^{\delta(v_i)} k.b(k; \delta(v_i), a_i^j), \tag{3}$$

where $b(k; \delta(v_i), a_i^j)$ is the binomial distribution that gives the probability of receiving exactly k packets out of $\delta(v_i)$ transmitted. The total number of packets received by v_j from all the lower priority nodes can be computed as

$$n_t(v_j) = n_{t-1}(v_j) + \sum_{i=0}^{j-1} E[X_{ij}]\,. \tag{4}$$

The first term in Equation 4 denotes the total number of packets received till the last round. The second term denotes the number of packets received in the current round t.

In OR, receiving only one packet from a forwarder is enough to receive all the acknowledgments from that node. The probability of receiving at least one packet by

v_j, out of $\delta_t(v_{j+1})$ packets transmitted by v_{j+1}, can be computed as $\left\{1-(1-a_j^{j+1})^{\delta_t(v_{j+1})}\right\}$. Using this expression, the number of acknowledgments can be formulated as

$$ack_t(v_j) = \sum_{l=j+1}^{m} \left\{1-(1-a_j^{l})^{\delta_t(v_l)}\right\} n_t(v_l), \tag{5}$$

where $n_t(v_l)$ is the number of packets received by v_l in t^{th} round.

As failures are independent in wireless channel, the set of packets held by a higher priority node may not be a subset of the set of packets held by lower priority nodes. In other words, it means out of the total number of acknowledgments received by a node v_j (as shown in Equation 5) from its higher priority forwarders, some of the corresponding packets may not be in the buffer of v_j. Thus, to compute the number of packets to be transmitted by a forwarder, we need to compute the set difference of the total number of packets received and total number of acknowledgments received. Let us define for a node v_j, the set of acknowledgments corresponding to $ack_t(v_j)$, as $A_t(v_j)$. Similarly, let the set $N_t(v_j)$ denote the set of packets received corresponding to $n_t(v_j)$.

For all the nodes in the forwarder list, we can define the total number of retransmissions for a round t as

$$R_t = \sum_{i=0}^{m} |N_t(v_i) - A_t(v_i)|. \tag{6}$$

The above equation shows the sum of the transmissions performed by all the participating nodes in a certain round. Equation 6 can be extended to calculate total number of retransmissions needed to deliver a batch of packets as follows.

$$R_{\text{total}} = \sum_{t=1}^{\infty} R_t \tag{7}$$

Equation 7 can be used to compute the number of retransmissions for any OR protocol. The value for t is taken from 1 to ∞, as the number of rounds to be taken to deliver the packets is not known in prior. This will not affect the model by getting into an infinite loop as the simulation will stop, when after some rounds, the number of packets transmitted becomes zero and the value of R_{total} becomes constant. The values of the variables used in the equation can be computed using the Equations 4 and 5.

4.2 Validation of mathematical model

In order to test the accuracy of our mathematical model, we first implemented an OR protocol ExOR [6,18,20] in simulation framework and recorded the total number of transmissions for a particular network scenario.

4.2.1 Network model

We implement ExOR in the discrete event driven simulator EXata Cyber [22] and measure the results. The simulations are run with 100 nodes, randomly spread over an open area of 1,500 × 1,500 m^2. The path loss model used is log-normal shadowing model [23]. This shadowing model includes the path loss and the random shadowing effects of an wireless channel that may occur over a large number of measurement locations. The nodes are configured to use the 802.11b specification in broadcast mode. This mode does not use the RTS/CTS mechanism. The data rate is fixed to 4 Mbps, as bit rate selection algorithms are not compatible with any broadcast-based routing protocol. We take the batch size to be of 100 packets, where each of the packets contain 1 kbyte of data. We assume the nodes to be static and no background traffic.

Experiments were performed for different network scenarios, with nodes being deployed randomly. The source and destination pair were chosen randomly with different shortest path distance. In the experiments, the hop counts considered varied from 1 through 14. The outcomes from same shortest path distances are then averaged to obtain the results. During the experiment, we recorded the forwarder lists and the PDRs between each pair of nodes which are required as inputs to the mathematical models. These values are then fed into the mathematical models, *discrete-time Markov chain model* [17] and our proposed binomial model. The expected number of transmissions for both the models were computed.

4.2.2 PDR measurements

To measure packet delivery ratio, each node broadcast measurement packets at regular intervals, which are received by all nodes in its range. We use broadcast, as unicast packets have MAC layer retransmissions, which will falsely portray the link qualities to be better. Each of these packets contain a packet number such that receiving nodes can recognize the packets they missed out. To make the PDR measurement more accurate, the size of the measurement packets is same as that of data packets, which is 1 kbyte in our case. Another important part is bit rate. As the packet error rate depends upon the bit rate used, the bit rate is kept constant (at 4 Mbps) during both PDR measurement as well as data packet transmission. The PDRs are computed using 500 measurement packets. In general, the overhead of PDR measurements in OR protocols is comparable to link state updates used in traditional proactive protocols [24].

4.2.3 Results

In this work, we implemented ExOR, but using any other OR protocol would make no difference on the result, since we use the same PDRs and forwarder list used by the OR protocol in the mathematical models. Figure 5

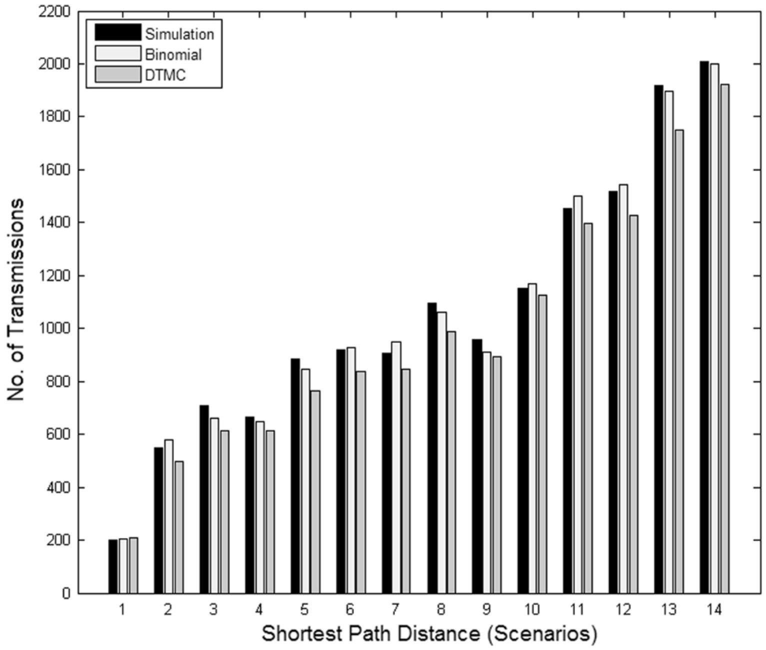


Figure 5 Expected number of transmissions for different shortest path distances. The figure represents the comparison results of ExOR protocol using simulation, binomial model, and the DTMC model for the comparison metric expected number of transmissions.

compares the three results obtained from simulation, the DTMC model, and binomial model. We can see from the figure that in general as the distance between source and destination increase, the number of transmissions also increase for all the three cases. From the figure, we can see that the binomial model gives more precise results over the DTMC model. The number of transmissions for the DTMC model is less than the simulation results. However, we also find scenarios where all the three results are similar, as in scenarios 1, 4, and 10. These are the cases that show scenarios where all the forwarders lie in a single path, and the inter-node communications (PDRs) are good. The explanation for these results is that the PDRs between adjacent forwarders are high, assuring the propagation of the acknowledgments to lower priority forwarders from higher priority forwarders. For this reason, as the knowledge of packet reception converges through all the forwarders, the chance of duplicate packet transmissions is very nominal.

The scenes where the nodes are not close to each other, that is, the PDRs between the nodes are not high (such as scenes 3, 7, 9, 14, etc. of Figure 5), we can see that the number of transmissions are significantly less in the case of the DTMC model. In this case, as the PDRs are less between the forwarders, acknowledgments may be lost. So, during instances where the acknowledgments do not reach the lower priority forwarders, duplicate retransmissions will occur. The DTMC model assumes that the acknowledgments are always propagated with a probability of 1; hence, it is unable to capture such duplicate transmissions. Therefore, for such cases, the number of transmissions predicted by the DTMC model is always less than the simulation results. On the other hand, the proposed binomial model considers the propagation of the acknowledgments through the forwarders, hence computation of the transmissions is more realistic.

Scenes 2, 6, 12, and 13 of Figure 5 depict those scenarios where the forwarder selection procedure selects nodes from divergent paths. In such scenarios, since two adjacent priority forwarders do not belong to a single path, the nodes are not in each others transmission range. This infers that information from the higher priority forwarder in one path is unlikely to propagate to the lower priority forwarders belonging to the other path. As a consequence, the lower priority nodes on the second path retransmit packets already transmitted by the other nodes in the first path, causing significant increase in the number of transmissions. The DTMC model do not consider such cases; hence, the number of transmission predicted by it is significantly less. We will look into more such scenarios in the following section. We find that our proposed binomial model closely follows the simulation results.

4.2.4 Duplicate packets

In Section 3, we explained how retransmissions can happen in opportunistic routing protocols. In this section, we provide experimental results to prove our claim. From the experiments conducted, we consider a scenario where the source (node 55) and destination (node 51) are two hops away, as shown in Figure 6. The paths to the destination formed by the selected forwarder set {55, 57, 12, 5, 15, 51} are shown with solid arrows. We can see that the pairs of adjacent priority forwarders {57, 12} and {5,15} are not in the communication range of one another. Initially, source node 55 forwards a batch of 100 packets which is received by both nodes 12 and 57. The forwarder nodes 12 and 57 receive 100 and 90 packets, respectively, and buffer them. Among the two, node 12 being the higher priority forwarder, it further forwards the packets but this transmission does not reach the lower priority node 57. Hence, node 57 again transmits the 90 packets in its buffer, resulting in 90 duplicate packets. Similarly, duplicate packets will again be generated for nodes 5 and 15, too.

The number of packets forwarded by each forwarder is shown in Figure 7. We can see that the number of packets transmitted by nodes 57 and 12 are more than the batch size. From Figure 6, we can see that there are two parallel paths to the destination: 55 → 12 → 5 → 51 and 55 → 57 → 15 → 51. The PDR of the links {12, 5} and {57, 15} is low. As a result, packet loss on these links is high, and so, nodes 12 and 57 have to (re)transmit more number of packets. In Figure 7, we see that nodes 5 and 15 transmit almost equal number of packets. Node 15, being the higher priority node, starts transmitting first. However, since node 5 is not in the transmission range of node 15, it is unable to get the acknowledgments. Node 5, therefore, waits for five packet durations and starts transmitting, before node 15 finishes its turn. So, the packets transmitted by node 5 are due to the non-existent link between the two nodes and are purely duplicates.

The total number of transmissions for the above scenario, as obtained from simulation and as predicted by the binomial and DTMC model, is shown as scenario 2 in Figure 5. As explained earlier, the DTMC model does not consider the divergent paths into consideration; hence, it is not able to capture the retransmissions. Thus, the number of transmissions predicted by the DTMC model is significantly less than the simulation results. The binomial model considers the probability of receiving an acknowledgment as can be seen in Equation 5. Therefore, the number of transmissions predicted by the binomial model is very close to the simulation result. Such duplicate transmissions can be avoided if we ensure that adjacent forwarders are in the transmission range of one another.

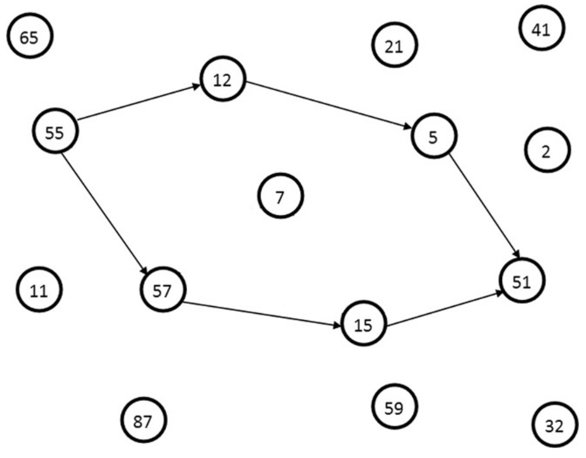


Figure 6 Scenario depicting duplicate transmissions. Depicts an example scenario to show how duplicate packets can be generated if the forwarders are selected from divergent paths. Here, the paths through the node sets {12, 5} and {57, 15} are mutually disjoint.

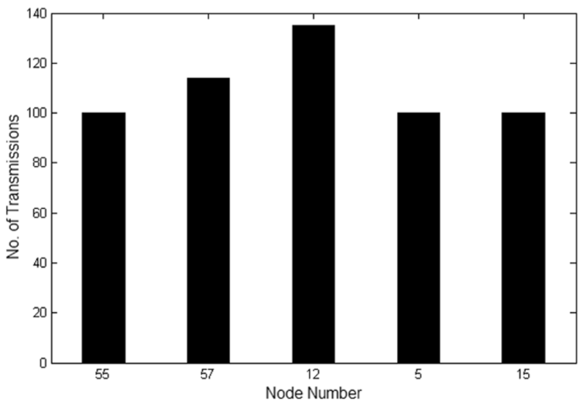


Figure 7 Number of transmissions per forwarder for ExOR. Shows the number of packets forwarded per forwarder obtained through simulation for the scenario shown in Figure 6.

5 Problem formulation and objectives

Let $(V, \preceq)$ be a totally ordered set where the set $V = \{v_0, v_1, \ldots, v_m\}$ is the set of forwarders. The set $\overrightarrow{V}$ is a totally ordered set, sorted according to the increasing orders of node priority with source node v_0 having the least priority and destination v_m having the highest priority. With a view to order the nodes, we define ℓ as a function that assigns weights to nodes and links, where $\ell[v]$ means the weight of node v and $\ell(u, v)$ means the weight of the link (u, v). The weight metric can be logical distances between nodes, hop counts to the destination, etc. The operator $\preceq$ is used to compare the priorities among two nodes in terms of their weights given by the weight function. A node u is said to be preferred over v if $\ell[u] \preceq \ell[v]$. We also define $N(v_i)$ as the set of higher priority nodes which are in the transmission range of v_i.

In this work, our aim is to compute the set $\overrightarrow{V}$ such that the number of retransmission is reduced. In Section 3, we have already shown that if the forwarders selected lie in divergent paths, then the number of retransmissions is increased due to duplicate retransmissions. If we ensure that all the adjacent forwarders are in the range of each other, then retransmissions will reduce. Mathematically this can be stated as,

Given: $v_j, v_k \in N(v_i)$,
where: $\ell\left[v_k\right] \preceq \ell\left[v_j\right] \preceq \ell\left[v_i\right]$
then: $\overrightarrow{\{v_j, v_k\}} \nsubseteq \overrightarrow{V}$
if: $v_k \notin N(v_j)$.

This means if we have a node v_i and two other nodes v_j and v_k in the transmission range of v_i, then both can appear as the forwarders of v_i, *iff* they are also in the transmission range of one another. Else, only one of them may appear as a forwarder.

6 Transmission-aware OR

In OR, there are three basic steps involved to select and prioritize the forwarders list.

1. *Candidate list selection*: From the set of nodes between source-destination pair, potential nodes, that is, nodes that can actually improve the performance of routing, are to be selected.
2. *Forwarder set selection*: From the set of candidate nodes, those nodes that will maximize the network objective are to be selected. At the end of this step, forwarders will be obtained.
3. *Prioritization of the forwarders*: The selected forwarders need to be prioritized which will decide their order of transmissions.

We encounter each step stated above and describe them individually. The main highlight of our proposed OR protocol is that, while selecting the forwarder list, we ensure that adjacent forwarders are in each others transmission range. Thus, we call our proposed approach transmission-aware OR *ad hoc* routing protocol (TOAR).

6.1 Candidate list selection

OR is basically a source-based routing protocol. The nodes periodically flood the link state of their neighbors using link state advertisement (LSA) packets. After the LSAs have been flooded, a source node will have map of the entire network. In order to route packet to a destination, the source node needs to find those intermediate routers which can forward packets toward the destination. These routers are then encoded in the packet headers as a *forwarder list*, and the packets are dispatched. The network map at the source node will have all possible paths to a destination. This is comparable to the topology table of EIGRP [25] which contain all routes to a destination. We view the entire set of paths from a source to destination as a tree rooted at the source and leave nodes as destination. We call such a tree as a *full path tree*.

Definition 1 (Full path tree). *A full path tree, for a given source destination node pair, is a tree consisting of all possible paths from source to the destination node. Source node is considered as the root, and all leaf nodes are the destination node.*

The full path tree can be trivially built by starting with the source node (v_s) as root. Nodes in the set of neighboring nodes of (v_s) are selected as child. The process is repeated recursively for each child node until the destination node is reached. However, the full path tree does not ensure loop-free paths. Figure 8 represents the full path tree for the network topology shown in Figure 4. In Figure 8, if we follow the right sub-tree through v_3, we can reach v_1. However, from v_1, there is a path to v_3, as can be seen in the left sub-tree rooted at v_0. Thus, the nodes v_3 and v_1 form a loop. Our aim was to remove such loops from the full path tree. Just like conventional routing algorithm [25] clears bad route from its topology table, we also need to discard the sub-paths that create loops. In Theorem 1, we give a necessary and sufficient condition to make the full path tree loop-free.

Theorem 1. *In a full path tree, if any two nodes do not appear in the sub-tree of one another, then the tree has loop free path.*

Proof. Consider two nodes v_i and v_j such that v_j appears in the sub-tree rooted at v_i. Further, we also assume that v_i appears in the sub-tree rooted at v_j. If we follow the first path, we can arrive at v_j, and since v_i appears in the sub-tree rooted at v_j, there will also be a path from v_j to v_i. Thus, the path between v_j and v_i forms a loop. □

A full path tree that satisfies Theorem 1 is called a *candidate tree*. It is so named because each node in the tree is a potential candidate for being a forwarder.

Definition 2 (Candidate tree). *A candidate tree for a given source destination pair is a full path tree where the paths do not contain any routing loop.*

Since our aim in this work was to reduce the number of retransmissions, we need to choose a metric for the weight function that measures efficiency in terms of successful packet delivery. One straight forward choice of such a weight function can be inverse of PDR, that is,

$$\ell \propto \frac{1}{\text{Packet delivery ratio}}. \tag{8}$$

We employ this weight function as our ℓ to assign weights to the nodes. In order to obtain the candidate tree from the full path tree, we apply Theorem 1 on the full path tree. The weight function ℓ we use here can be defined as $\ell[u] \preceq \ell[v] \Rightarrow \ell[u] \leq \ell[v]$, which means the lower the weight, the higher the priority. The algorithm to obtain the candidate tree is shown in Algorithm 1.

Algorithm 1 Algorithm to construct a Candidate tree

```
Input: full path tree $T$
Output: $M(v_0)$
 1: $h \leftarrow T.depth - 1$; $R \leftarrow \phi$
 2: //Initialize the leaf(destination) node
 3: $\ell\,[dest] \leftarrow 1$
 4: //Assign weight to nodes bottom-up
 5: while $h! = 0$ do
 6:     for all node $v$ in ($T$ at $height(h)$) do
 7:         //Compute weight of each node
 8:         for all node $u$ in $child(v)$ do
 9:             $\ell\,[v] \leftarrow min\{\ell\,[u] + \ell(v,u)\}$
10:         end for
11:         //Select nodes with weight less than parent
12:         for all node $u$ in $child(v)$ do
13:             if $\ell\,[u] \preceq \ell\,[v]$ then
14:                 $M(v) \leftarrow M(v) \oplus M(u)$
15:             end if
16:         end for
17:     end for
18:     $h = h - 1$
19: end while
```

The procedure starts from the leaf nodes and ends at the root node. Let T denote the full path tree and $T.depth$ be the depth of the full path tree T. The function $height(h)$ returns all nodes at *level h*. The source or root node (v_0) is at level 0. The weight of the leaf nodes will have the lowest weights as they are the destination nodes themselves. Let us also denote $M(v)$ as the candidate tree or a candidate sub-tree rooted at a node v. We define $\oplus$ as an operator that merges two sub-trees to create a new one. The algorithm iteratively adds the child nodes to the new tree which are logically closer to the destination than their parents, as given in line 13 of the algorithm.

6.2 Selection and prioritization of the forwarders

Once the candidate tree containing the candidate nodes is prepared, we are in a position to select the potential forwarders to build the forwarder list that will finally participate in packet forwarding. For this purpose, we need to further prune the candidate tree to get a more simplified tree containing only the forwarder nodes.

In Section 4, we showed that if adjacent forwarders cannot listen to each others transmission, then the two forwarders are on separate paths. Such a scenario can increase the number of retransmissions. Our aim was to

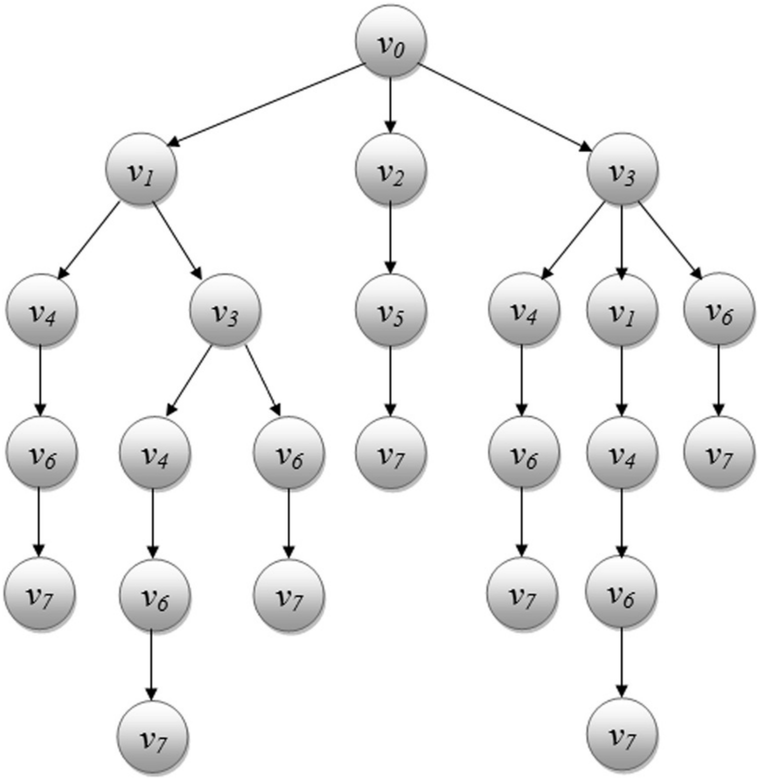


Figure 8 Full path tree for the scenario shown in Figure 4 with node v_0 as source and v_7 as destination. This figure shows the full path tree for the scenario shown in Figure 4 with node v_0 as source and v_7 as destination. All the possible paths are shown in the tree as the paths to the leaf nodes where all the leaf nodes denote the destination node.

select the forwarders, such that there is at least one common path which includes all of them. The candidate tree constructed using Theorem 1 will give us the shortest path to the destination. However, in OR, the aim was not to select the shortest path but to select forwarders that move packets closest to the destination. Therefore, we need to select the forwarders from each hop such that the number of transmissions needed is minimized. At each hop, if the best forwarder is unable to receive packets, then other forwarders that receive the packets must help it by retransmitting those packets. Nevertheless, in each hop, the highest priority forwarder, since it is closest to the destination among the other nodes in that hop, is clearly the best node to forward the packets. We call such nodes as primary forwarders. In [26], candidate nodes are also represented using OR tree, but such a tree does not allow lower priority nodes to be selected as backup forwarders.

Speaking in terms of the candidate tree, it means at each hop there must be exactly one primary forwarder, and zero or more secondary forwarders to backup transmission to that primary forwarder. We call such secondary forwarders as *backup forwarders*.

The number of backup forwarders to be selected per hop depends upon the improvement they offer in terms of weight for the current primary and its parent node duo.

Let v_i, v_j and v_k be three nodes with priorities given as $\ell\left[v_k\right] \preceq \ell\left[v_j\right] \preceq \ell\left[v_i\right]$. Let us also denote the delivery probability of a link (v_i, v_j) as a_i^j. If there exist a path $P_1 = v_i v_k$, then the forwarder list between v_i and v_k is $\{v_i, v_k\}$. If there exist a second path $P_2 = v_i v_j v_k$, then the following theorem holds a rule to select v_j as a backup forwarder.

Theorem 2. *Given three nodes v_i, v_j and v_k, if there exist two paths $P_1 = v_i v_k$ and $P_2 = v_i v_j v_k$ then adding v_j as a backup forwarder to the forwarder list $\{v_i, v_k\}$ will increase the delivery probability to v_k.*

Proof. Let the delivery probability of the links (v_i, v_j), (v_i, v_k), and (v_j, v_k) be a_i^j, a_i^k and a_j^k, respectively. Then, the PDR to node v_k such that it receives packet from v_i either through path P_1 or P_2 is

$$P_{v_j}\left(v_i, v_k\right) = 1 - (1 - a_i^k)(1 - a_i^j a_j^k), \tag{9}$$

where $P_{v_j}(v_i, v_k)$ is the delivery probability from v_i to v_k if the candidate node v_j is added as backup forwarder. The PDRs will be non-zero since there exists path between the nodes. Hence, from Equation 9, we can see that adding v_j in the forwarder list will increase the PDR to v_k by a factor equivalent to the second term. □

Our aim was to choose zero or more backup forwarders between each pair of adjacent primary forwarders such that the PDR improves. The weight function (ℓ) used for selecting candidate nodes will not work here as it represents the weight of a node through the shortest path. Though this weight function does not reflect the increment in probability of forwarding a packet in presence of backup forwarders, it is needed to prioritize the selected forwarders. Let us assume node v_i has two child nodes v_j and v_k, where v_k also appears as a child of v_j as shown in Figure 9. Assuming v_i and v_k are primary forwarders, then v_j can act as a backup forwarder. Including v_j as a (backup) forwarder will improve the PDR of v_i, or in other words, its weight will be decreased. However, if we keep on adding more forwarders between v_i and v_k, the weight of v_i may get decreased to such an extent that its weight becomes less than its child node. This is a violation of rule since, in a candidate tree, the child nodes should have less weight (higher priority) than their parents. Thus, we need to preserve the order of the candidate nodes with the help of the weight function ℓ.

To measure the improvement in PDR due to the introduction of backup forwarders, we introduce effective weights ℓ^{eff}. The effective weight of the path from v_i to v_k after incorporating v_j as a (backup) forwarder can be directly computed from Theorem 2 as shown below:

$$\begin{aligned} \ell^{eff}_{\{v_j\}}(v_i, v_k) &= \frac{1}{1 - (1 - \frac{1}{\ell(v_i,v_k)})(1 - \frac{1}{\ell(v_i,v_j)\ell(v_j,v_k)})} \\ &= \frac{1}{1 - C(1 - \frac{1}{\ell(v_i,v_j)\ell(v_j,v_k)})}. \end{aligned} \tag{10}$$

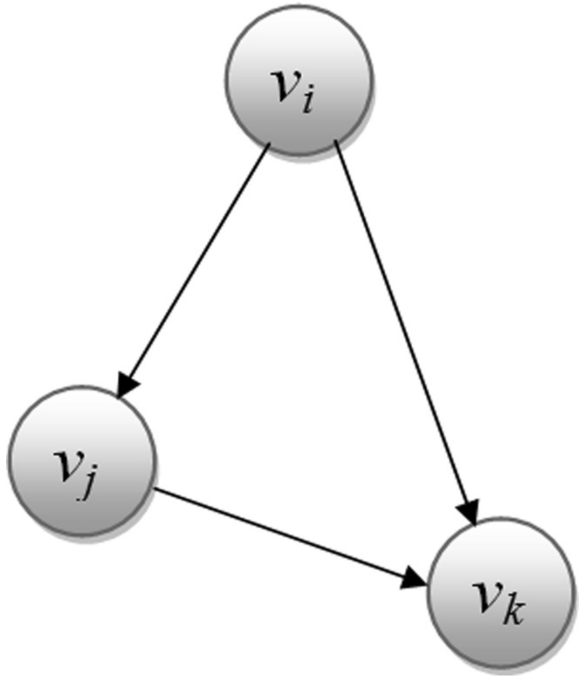


Figure 9 Example of primary and back-up forwarder selection. The figure depicts an example on how to select primary and backup forwarders from the candidate tree using Theorem 2. Here, node v_i tries to select node v_k as primary and node v_j as backup forwarders.

In Equation 10, since the primary forwarders v_i and v_k are fixed, we denote the term $(1 - \frac{1}{\ell(v_i,v_k)})$ with a constant C. The possible effective weight of node v_i using v_k as primary and v_j as a backup forwarder, $\ell^{eff}_{\{v_j\}}[v_i]$, can be calculated in the same manner as we computed the weights of the nodes.

$$\ell^{eff}_{\{v_j\}}[v_i] = \ell^{eff}_{\{v_j\}}(v_i, v_k) + \ell^{eff}[v_k]. \tag{11}$$

The actual effective weight of a node will be computed by considering all its child nodes and possible backup nodes. For example, in Figure 8, node v_i has two child nodes v_j and v_k. The effective weight of v_i will be computed by considering the two child nodes as possible primary forwarders with no backup nodes; or consider one as a primary forwarder and the other as a backup forwarder (if possible). Thus, in this example, there are three possible options for node v_i to choose from. Among these three possible weights, the one having the least value is selected as the effective weight of v_j as can be seen from the following equation.

$$\begin{aligned}\ell^{eff}[v_i] = \text{MIN}\Big\{&\ell^{eff}_{\{\phi\}}(v_i, v_k) + \ell^{eff}\left[v_k\right],\\ &\ell^{eff}_{\{v_j\}}(v_i, v_k) + \ell^{eff}\left[v_k\right],\\ &\ell^{eff}_{\{\phi\}}(v_i, v_j) + \ell^{eff}\left[v_j\right]\Big\}.\end{aligned} \tag{12}$$

The forwarder set of node v_i is then prepared by adding the node itself, the backup forwarder v_j and the current forwarder set. If the nodes v_j and v_k are selected as next hop forwarders then FWD(v_i), the forwarder set of v_i can be built as follows.

$$\text{FWD}(v_i) = \{v_i\} \cup \{v_j\} \cup \text{FWD}(v_k).$$

This forwarder set is then prioritized using the weight function ℓ that gives a proper ordering of the forwarders along the destination node, where highest priority node has the lowest weight.

The above procedure can be generalized for large candidate trees. In Equation 10, we assumed that only one node is available as backup forwarder. However, more than one node maybe available as backup forwarders. In such a case, we first order the nodes according to their weights. Each of them is then examined, by adding them as backup node starting with the one having the highest priority and check the improvement in the effective weight of the parent node. Let us assume R as such a set of backup forwarders where $R = \{v_{j_1} \ldots v_{j_n}\}$. Using Equation 10, the effective link weight $\ell^{eff}_R(v_i, v_k)$ for this set of backup forwarders can now be written as

$$\ell^{eff}_R(v_i, v_k) = \frac{1}{1 - C\sum_{m=1}^{n}(1 - \frac{1}{\ell(v_i,v_{jm})\ell(v_{jm},v_k)})}. \tag{13}$$

Using this effective link weight, the effective weight $\ell^{eff}_R[v_i]$ can be computed by using Equation 12. The computation of effective weight of the nodes and the forwarder set follow a bottom-up approach. The leaf nodes are assigned effective weight of 1. The effective weight of the next higher nodes are then computed using the above equation. The process continues recursively until we reach the node at height 0, i.e., the root node. At this point, we have the forwarder set for the root (source) node, and the process terminates.

As there may be zero or more backup nodes, the size of the final forwarder list mainly depends upon the backup nodes selected in the list. This leads us to the need of limiting the number of nodes selected as backup forwarders. A node is considered as a possible backup, only if it offers a certain amount of improvement in the effective weight. In this work, we add a node as a possible backup forwarder if it only offers a 10% improvement. This threshold value has been taken in line with ExOR, where a node is included as a forwarder, if the number of packets expected to be forwarded by that node is 10% or more. Size of the backup forwarder list can be managed by regulating the threshold value. This cut-off process ensures that, in network scenarios where nodes are dense, the number of backup nodes do not become unmanageable.

In the example shown in Figure 10, we show how the threshold value controls the number of forwarders selected in the forwarder list. The figure shows a scenario where source v_0 and destination v_5 have four equally good candidate nodes. The PDRs of the links are marked alongside in the figure. Further, we assume that the PDRs of all the links between these candidate nodes are 1. As all the candidate nodes have equal weight of 2 $(\frac{1}{0.5})$, v_0 may select any of them as the primary forwarder. Let v_1 be the selected primary forwarder, weight of v_0 becomes 5 (2+2+1). Now, the source node will try to reduce its effective weight by adding backup nodes. If we add node v_2 to the list, the new weight of the link v_0 to node v_1 will become 1.3333, $(\frac{1}{1-(1-0.5)(1-0.5*1)} = \frac{1}{0.75})$. Since there is an improvement of more than 10%, v_2 will be added as a backup forwarder. Similarly, adding v_3 as a backup node will further reduce the weight to 1.1428 $(\frac{1}{1-(1-0.75)(1-0.5*1)} = \frac{1}{0.875})$. At this point, if we consider adding v_4 as a backup node, the new reduced weight of link v_0 to v_1 will be 1.0667. Adding v_4 does not give an improvement by 10%; hence, it is not eligible to become a backup forwarder and the process of backup forwarder selection terminates.

Algorithm 2 shows the procedure for forwarder list selection. The function *sort*(*child*(v)) sorts the child nodes of node v in increasing order of their weight ℓ. The variable *Thresh* is the minimum improvement a forwarder set must contribute to the weight to replace the previous set. This procedure computes

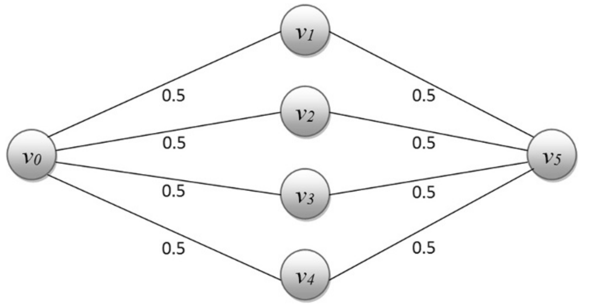


Figure 10 A sample scenario with four candidate nodes between source v_0 and destination v_5. The figure shows a sample network where source v_0 and destination v_5 have four candidate nodes between them, namely v_1, v_2, v_3, and v_4. The scenario is used to demonstrate backup forwarder selection procedure.

the forwarder list for each node to the destination in the tree and finally returns the root (source) node's forwarder list as FWD(v_0). Let us now check the worst case time complexity of selecting the next hop forwarders for node v_i. Let x be the size of the set $N(v_i)$ (refer to Section 5), that is, $|N(v_i)| = x$. As the algorithm searches for each node v_j in the set $N(v_i)$ ($\forall v_j \in N(v_i), j = 1 \ldots x$) and tries to add all the lower priority nodes in the set one by one ($v_{j-1}, v_{j-2}, \ldots, v_1$) in the sorted order the worst case time complexity of the process becomes $O(x^2)$.

Algorithm 2 Algorithm to create forwarder list

```
Input: $M(v_0)$ //Candidate-tree
Output: $FWD(v_0)$ //Forwarder List
$h \leftarrow M(v_0).depth - 1$
//Initialize leaf(destination) nodes
$FWD(dest) \leftarrow \{dest\}$
$\ell^{eff}\,[dest] \leftarrow 1$
//Compute effective weight bottom-up
while height $h! = 0$ do
    for all node $v$ at height $h$ do
        $S \leftarrow sort(child(v))$
        for all node $w$ in $S$ do
            R $\leftarrow$ Backup Forwarders of $w$
            $\ell_R^{eff}[v] = \ell_R^{eff}(v, w) + \ell^{eff}[w]$
            if $\ell^{eff}[v] - \ell_R^{eff}[v] > Thresh$ then
                $\ell^{eff}[v] = \ell_R^{eff}[v]$
                $FWD(v) = \{v\} \cup R \cup FWD(w)$
            end if
        end for
    end for
    $h = h - 1$
end while
```

6.3 Example: creating forwarder list

Continuing with our example, the candidate tree shown in Figure 11 is obtained by applying Algorithm 1 to the full path tree of Figure 8. Next, we show how Algorithm 2 can be applied to this candidate tree to create the forwarder list. The computed effective weights for the nodes of the candidate tree is shown in Figure 12. In this figure, all the nodes are labeled with their calculated *effective weights* except the leaves for which the *weights* are always 1. The *weights* of the links are also given. In the case of a node with a single child, the *effective weight* is same as *weight*, and the computation is straight forward. As we start constructing the forwarder list bottom-up, the forwarder lists of each intermediate node is shown in braces. Consider the right sub-tree rooted at v_3. Node v_7, being the leaf element, has effective weight 1, and node v_6, as it has only one child, computes its weight as 2.3. In similar manner, node v_4 (child of node v_3) calculates its effective weight as 3.5. Node v_3 has two children v_4 and v_6. Although the child v_6 has lower weight, node v_3 has two routes to v_6, one directly and the other through v_4. The algorithm will try to couple v_4 with v_6 which will reduce the weight to node v_6 from v_3 to

$$\frac{1}{1-(1-\frac{1}{1.3})(1-\frac{1}{1.3*1.2})} \approx 1.09,$$

resulting in the *weight* of v_3 as 3.39 (2.3 + 1.09). If node v_3 had considered node v_6 or v_4 separately, its *weight* would have been either 3.6 or 4.8. Since the algorithm considers both the node as forwarders, the resulting forwarder list of node v_3 is computed as $\{v_3, v_4, v_6, v_7\}$. Continuing in a similar manner, the three subtrees of the the source node (v_0) rooted at v_1, v_2, and v_3 get the *effective weight* 4.49, 3.5, and 3.39, respectively. The sub-tree rooted at v_3 has the lowest effective weight, and there are two paths to v_3 from v_0. The weight of the link to node v_3 through v_1 is computed as

$$\frac{1}{1-(1-\frac{1}{1.3})(1-\frac{1}{1.2*1.1})} \approx 1.06.$$

The effective weight of v_0, considering the sub-tree v_3 as its forwarder and the path to v_3 through v_1, is 3.39 + 1.06 = 4.45. Thus, the forwarder list of the source node becomes $\{v_0, v_1, v_3, v_4, v_6, v_7\}$. Note that, if we had used the conventional single path metric (e.g., ETX) to build the forwarder list, then node v_0 would have selected the the sub-tree rooted at node v_2 which has the minimum ETX weight 3.5 (node v_1 and v_3 has the weights 4.7 and 3.6, respectively). As this path does not offer much node availability, selecting the path will not be a very good choice.

7 Performance evaluation

For OR protocols to be useful, the average number of transmissions per packet must be lower as compared to conventional shortest path routing protocols. The experiments have been performed keeping two objectives in mind. First, we wanted to prove the performance improvements of OR protocols in general as compared to traditional shortest path first (SPF) routing protocols. The SPF protocol that we consider is dynamic source routing (DSR) [1]. Second, we wanted to show that our forwarder selection procedure (TOAR) results in overall improvement as compared to existing OR protocol like ExOR. The protocols were implemented in EXata Cyber simulation framework [22].

7.1 Network description

We used ExOR as the base OR protocol for our study to compare our protocol. We implemented ExOR, TOAR, and a traditional shortest path protocol DSR [1] in EXata Cyber network simulator for comparison and analysis.

As described in Section 4.2, we take 100 nodes that are scattered over an open play ground of 1,500 × 1,500 m^2 area. The path loss model considered is log-normal shadowing model. The nodes are assumed to be using 802.11b specification in broadcast mode. A fixed data rate of 4 Mbps is considered.

In our implementation, each node broadcasts a probe packet per second. Using the information of the number of probes successfully received over time, the packet delivery ratios of the links are measured. As these PDR information need to converge over the network for successful forwarder selection, we allow 500 s of warm-up time. After the warm-up time, the source node tries to send 100 kbytes of data to the destination as a batch of 100 packets, each of which contain 1 kbyte of data.

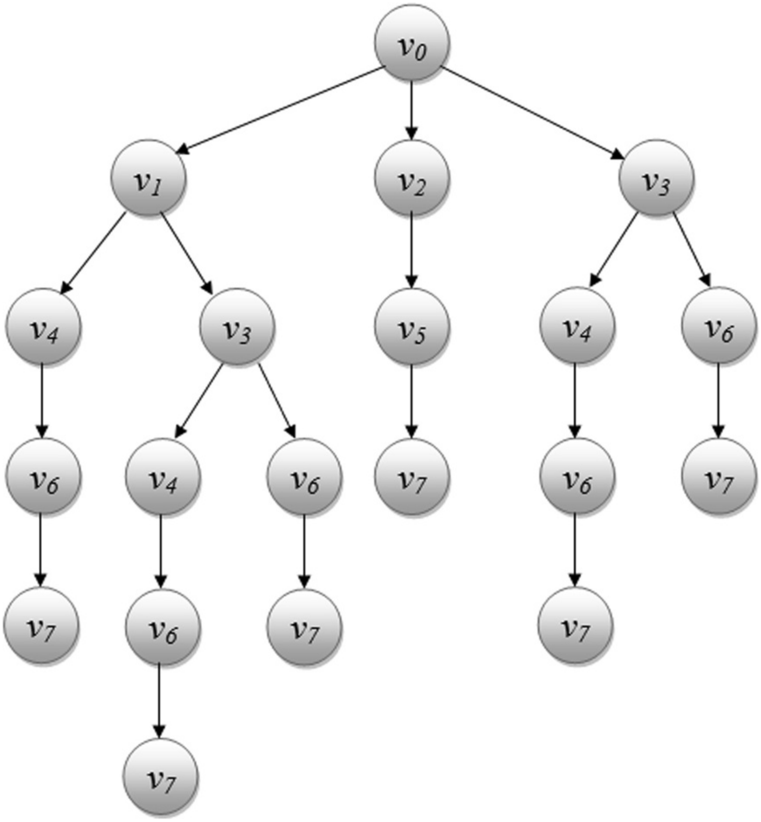


Figure 11 Candidate tree for the scenario shown in Figure 4 with node v_0 as source and v_7 as destination. The figure shows the candidate tree obtained from the full path tree in Figure 8 by applying Algorithm 1. The tree contains only the paths that do not form a routing loop.

The measurements are taken at the time of data transmission which are compared and analysed next.

7.2 Measurements

The experiments were performed for different network scenarios. The performance metrics considered were *number of transmissions required*, *size of forwarders list*, *average number of next hop nodes per forwarder*, and *goodput*. Our choice for these metrics is explained in the sections below.

7.2.1 Number of transmissions

We computed the number of transmissions, for different source-destination pairs, with different shortest path distances between them. In Figure 13, we show the result for the protocols ExOR, TOAR, and DSR. The reason for choosing DSR was that we want to compare the efficiency of OR and shortest path first routing (SPF) protocols. Moreover, DSR-like OR protocols is a source routing protocol.

From the figure, we see that in the case of single-hop distances, the number of transmissions in all the three protocols are similar, and as the distance increases, the OR protocols outperform DSR. In general, we find that using OR protocols, there is on average a 32% reduction in the number of transmissions. In DSR, the intermediate nodes forward packets only to the next forwarder as listed in the shortest path. In such cases, those nodes that are not the next hop node discard the packets even if it was successfully received. This is true for all SPF routing protocols. Hence, these protocols are unable to take advantage of the stochastic nature of wireless networks, which is considered in OR protocols to improve performance. Another drawback in SPF is that when a packet is received in error by an intermediate node, the packet is retransmitted from the source increasing the number of transmissions. In OR protocols, the lost packet is retransmitted by the forwarder closest to the destination that buffered the packet successfully.

Next, if we compare the results of our proposed protocol TOAR with ExOR, we see a performance improvement of around 20% to 30%. The main ground of performance improvement in our protocol is mainly due to the forwarder set selection procedure. ExOR simply selects the set of forwarders by running a simulation with the full set of candidates and select those nodes that forward at least 10% of the packets. As a result, there is a high probability that the protocol select nodes from different path that are mutually divergent. Hence, duplicate packets generated will be higher. On the other hand, TOAR ensures that a forwarder is in the communication range of its next higher and lower priority forwarders. Such a selection strategy improves the chances of successful propagation of acknowledgements to the lower priority forwarders. As a consequence, the expected number of duplicate packet transmissions is reduced. Moreover, this calculated selection procedure reduces the chance of improper forwarder

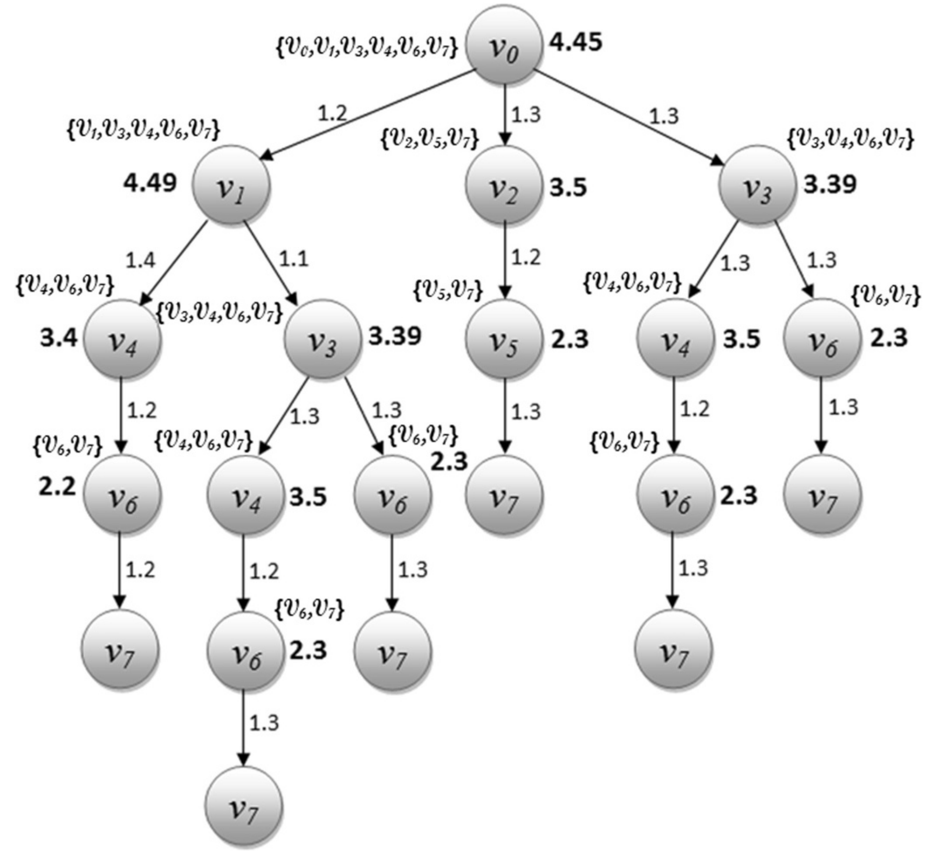


Figure 12 Effective weights and selection procedure for the Candidate tree in Figure 11. This figure shows the computation of effective weights and construction of forwarder lists for the candidate tree in Figure 11 using Algorithm 2. The computed effective weights as well as the constructed forwarder lists are marked along with the root and the intermediate nodes in the tree.

set selection leads to a smaller forwarder set size, as will be seen in the next section.

7.2.2 Number of selected forwarders

Another important metric for comparison of OR protocols is number of selected forwarders. A huge and tactically selected forwarder list may improve the performance, in terms of number of transmission required by increasing node availability for each forwarder. However, there are two disadvantages. Firstly, a large forwarder list will complicate the coordination among the nodes which will affect the throughput of the system. Secondly, OR protocols being source routing the packet header overhead will be high. The aim of OR protocols must be to select a small forwarder list yet maintain enough node availability for delivering the packets.

In Figure 14, we compare the forwarder list size for different source destination pairs. From the figure, we see that the number of forwarders selected by ExOR is generally higher than that of TOAR. On an average, TOAR selects about 25% less number of forwarders than ExOR. In a few cases, we observe that the selected number of forwarders in both the protocols are comparable. On further examination, we found that these are the cases where path availability is scarce and both the protocols are forced to select the same set of nodes. In the remaining cases where the number of forwarder list size is greater for ExOR, if we look at their corresponding number of transmissions in Figure 13, we also find it to be higher for ExOR. These observation prove our conjecture that ExOR selects nodes from different and disjoint paths.

7.2.3 Number of next hop nodes per forwarder

To evaluate and compare the quality of the different forwarder list, we use the comparison metric average out-degree, which is the average number of next hop forwarders available per forwarder. In OR, more number of next hop forwarders mean that the packets have more chance to move forward toward the destination at each transmission. A higher value in this evaluation will mean a better forwarder list with better node availability.

We depict the results of this comparison in Figure 15. We can observe from the figure that in the case of TOAR out-degree per forwarder is higher than ExOR for all the scenarios. This improvement is in spite of the fact that TOAR has a forwarder list of smaller size. Although ExOR has a larger forwarder list, it does not consider the link qualities between forwarders; hence, it has a smaller out-degree per forwarder.

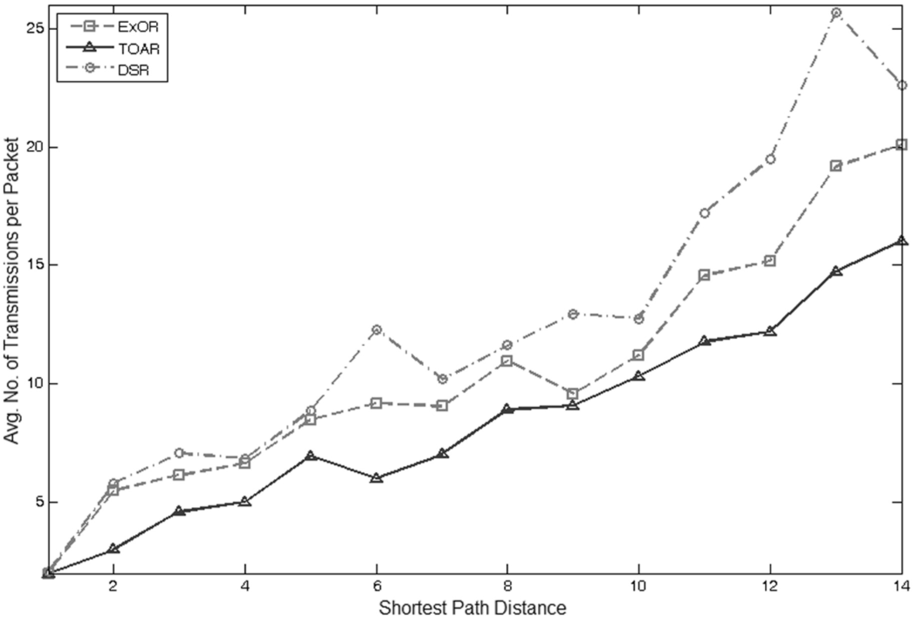


Figure 13 Expected number of transmissions for different shortest path distances and scenarios. The figure shows the simulation results of comparison of the three protocols ExOR, TOAR, and DSR in terms of expected number of transmissions. The figure is plotted for source-destination pairs with different shortest path distances (hop-counts).

7.2.4 Improvement in goodput

Throughput is an effective evaluation technique for most routing protocol. It measures the total number of bytes transferred including protocol overheads and retransmissions. However, in OR protocols, the overhead in the packet header (control information) is large and non-negligible. Therefore, a better way to compare OR protocols is to measure the number of data bits transferred per second (goodput).

The average goodputs of ExOR, TOAR, and DSR for the scenarios given in Figure 13 are shown in Figure 16. From the

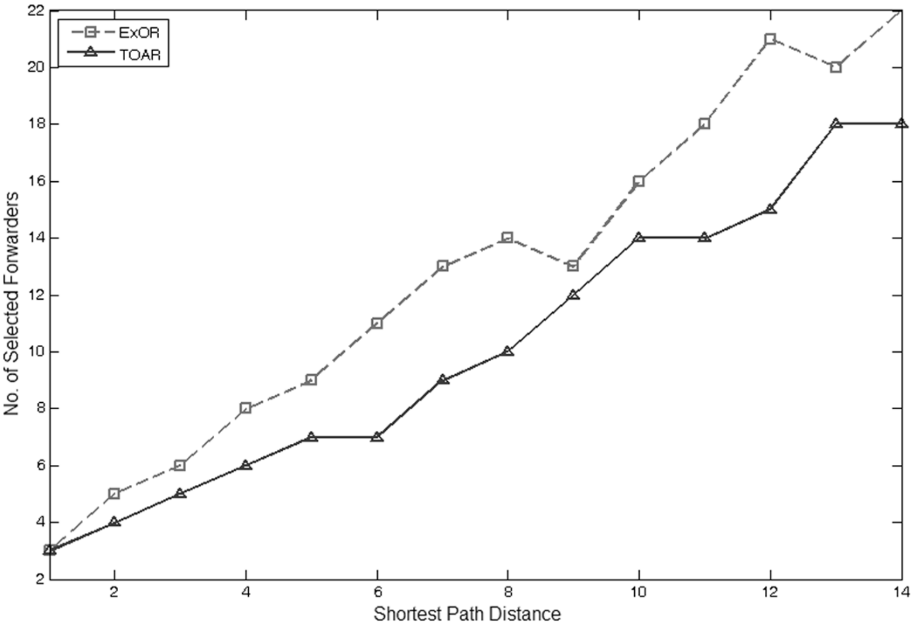


Figure 14 Number of selected forwarders for different shortest path distances and scenarios. This figure shows the simulation results of comparison of the two protocols ExOR and TOAR in terms of the number of selected forwarders. The figure is plotted for source-destination pairs with different shortest path distances (hop-counts).

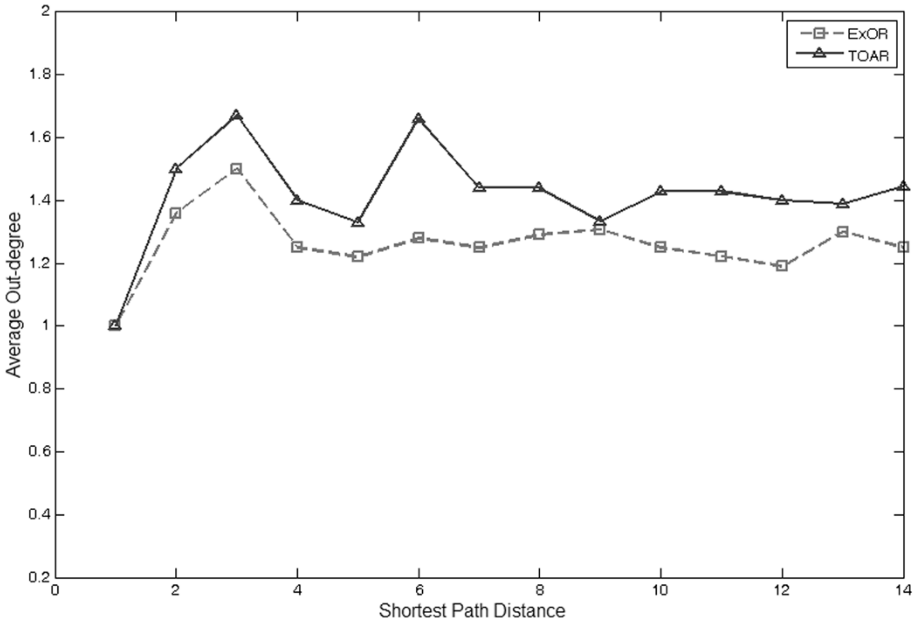


Figure 15 Average number of out-degree per forwarder for different shortest path distances and scenarios. This figure shows the simulation results of comparison of the two protocols ExOR and TOAR in terms of average number of out-degree per forwarder, that is, average number of next hop nodes available per forwarders. The figure is plotted for source-destination pairs with different shortest path distances (hop-counts).

figure, we can see that our protocol outperforms ExOR by 25% to 30% and DSR by 50% to 60%. The scheduling scheme used in TOAR is same as that of ExOR. The improvement in goodput is mainly due to two reasons. Firstly, as the number of transmissions needed to deliver packets is less in case of TOAR, the total time required for completion of delivery is also reduced. Secondly, due to our forwarder selection strategy, each pair of forwarders in the list has forward paths (path toward destination) between them. Thus, coordination between the forwarders is simpler.

In DSR, the main reason of decrease in goodput is due to the increase in the number of transmissions, as can be seen from

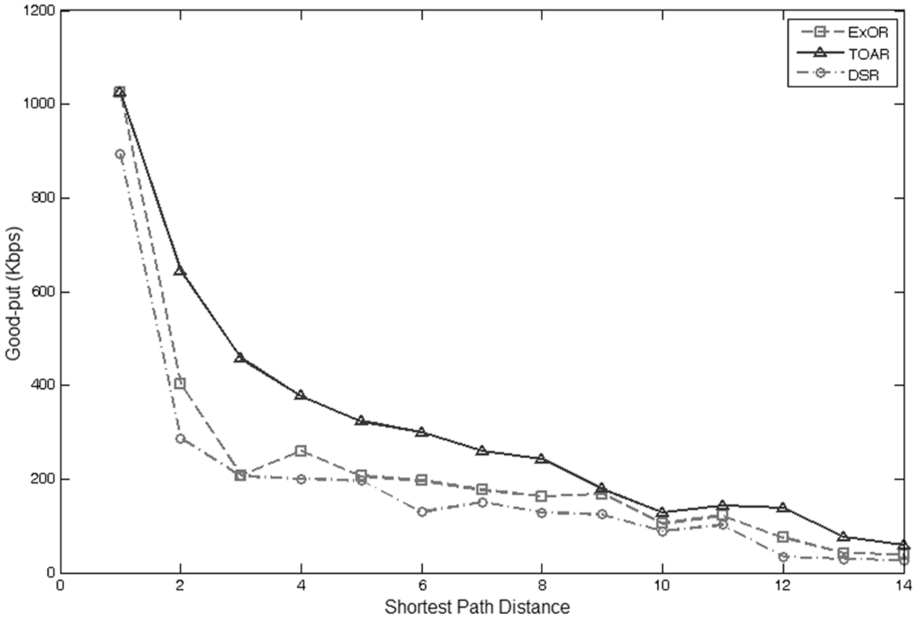


Figure 16 Goodputs of ExOR, TOAR, and DSR for different shortest path distances and scenarios. The figure shows the simulation results of comparison of the three protocols ExOR, TOAR, and DSR in terms of goodput. The figure is plotted for source-destination pairs with different shortest path distances (hop-counts).

Figure 13. More number of packet transmissions mean more time in packet delivery. Secondly, when a packet is not received by an intended intermediate node, the sender waits for a certain amount of time before retransmitting the packet again. This waiting time increases the delivery time further over the number of transmissions.

8 Conclusions

In this paper, we first proposed a mathematical model for OR protocol that can be used to study and analyse the protocols in terms of expected number of transmissions to deliver packets. Using this model, we studied classical OR protocols like ExOR and examined the cause of duplicate packet transmissions. We found that existing OR routing models assumed the protocol to be ideal in the sense that there were no duplicate packet transmissions. Our analysis indicated that duplicate packet transmissions occur if a lower priority forwarder cannot hear the transmission of its adjacent higher priority forwarder.

Based on our mathematical analysis, we proposed TOAR, transmission-aware opportunistic *ad hoc* routing protocol. The goal of our proposed OR protocol is to forward packets closest to the destination during each transmission as well as minimize the duplicate transmissions. The primary steps in TOAR are (1) select the potential nodes that can act as forwarders (candidate nodes); (2) from among the candidate nodes, select the actual forwarders such that network objective function is maximized, and (3) prioritize the forwarders. TOAR performs all these operations by constructing a *full path tree* and systematically pruning nodes of the tree. As our proposed model considers the number of duplicate packets generated, the results are more realistic. Empirical results show that TOAR significantly reduces the number of duplicate transmissions, and at the same time, the number of forwarders selected is small. The efficient selection of forwarders also results in better goodput of TOAR.

In *ad hoc* networks, since nodes can be mobile, in the future, we would like to consider node mobility during forwarder selection. *Ad hoc* networks are usually deployed in energy constrained scenarios, thus another important parameter that needs to considered is the power level of the nodes.

Competing interests

The authors declare that they have no competing interests.

Received: 14 December 2012 Accepted: 17 September 2013
Published: 25 September 2013

References

1. DB Johnson, DA Maltzand, Dynamic source routing in *ad hoc* wireless networks. Mobile Comput. **353**(5), 153–181 (1996)
2. CE Perkins, P Bhagwat, Highly dynamic destination-sequenced distance-vector routing DSDV for mobile computers. SIGCOMM Comput. Commun. Rev. **24**(4), 234–244 (1994)
3. CE Perkins, E Belding-Royer, *Ad hoc* on-demand distance vector (AODV) routing. IETF Netw. Working Group, RFC 3561 1994. http://www.ietf.org/rfc/rfc3561.txt. Accessed 10 Dec 2012
4. P T Clausen, P Jacquet, Optimized link state routing protocol (OLSR). IETF Netw. Working Group, RFC **3626** 2003. http://www.ietf.org/rfc/rfc3626.txt. Accessed 10 Dec 2012
5. AP Mazumdar, AS Sairam, in *Proceedings of the International Conference on Network, Communication and Computing*. Opportunistic routing: opportunities and challenges (New Delhi, 19–20 March 2011), pp. 191–195
6. S Biswas, R Morris, ExOR: opportunistic multi-hop routing for wireless networks. SIGCOMM Comput. Commun. Rev. **35**(4), 133–144 (2005)
7. DD Couto, D Aguayo, J Bicket, R Morris, in *Proceedings of the ACM 9th Annual International Conference on Mobile Computing and Networking (MobiCom '03)*. A high-throughput path metric for multi-hop wireless routing (San Diego, 14–19 Sept 2003), pp. 134–146
8. S Chachulski, M Jennings, S Katti, D Katabi, Trading structure for randomness in wireless opportunistic routing. SIGCOMM Comput. Commun. Rev. **37**(4), 169–180 (2007)
9. X Mao, S Tang, X Xu, XY Li, H Ma, Energy-efficient opportunistic routing in wireless sensor networks. IEEE Trans. Parallel Distributed Syst. **22**(11), 1934–1942 (2011)
10. J Wu, M Lu, F Li, in *Proceedings of the IEEE Computer Society 1st International Conference Simulation Tools and Techniques for Communications, Networks and Systems*. Utility-based opportunistic routing in multi-hop wireless networks (Marseille, 03–07 March 2008), pp. 470–477
11. M Zorzi, RR Rao, Geographic random forwarding (GeRaF) for *ad hoc* and sensor networks: multi-hop performance. IEEE Trans. Mobile Comput. **2**(4), 337–348 (2003)
12. J Kim, B Ravindran, in *Proceedings of the ACM Symposium on Applied Computing 2009 (SAC '09)*. Opportunistic real-time routing in multi-hop wireless sensor networks (Honolulu, 9–12 March 2009), pp. 2197–2201
13. Y Yuan, H Yang, S Wong, S Lu, W Arbaugh, in *Proceedings of the IEEE WiMesh 2005*. ROMER: Resilient opportunistic mesh routing for wireless mesh networks (Santa Clara, 26 Sept 2005)
14. E Rozner, J Seshadri, Y Mehta, L Qiu, SOAR: simple opportunistic adaptive routing protocol for wireless mesh networks. IEEE Trans. Mobile Comput. **8**(12), 1622–1635 (2009)
15. H Dubois-Ferriere, M Grossglauser, M Vetterli, Valuable detours: least-cost anypath routing. IEEE/ACM Trans. Netw. **19**(2), 333–346 (2011)
16. BR Alspach, P Hell, DJ Miller, *Algorithmic Aspects of Combinatorics*. (North-Holland, Amsterdam, 1978)
17. A Darehshoorzadeh, L Cerda-Alabern, V Pla, Modeling and comparison of candidate selection algorithms in opportunistic routing. Comput. Netw. **55**(13), 2886–2898 (2011)
18. SZ Biswas, Opportunistic routing in multi-hop wireless networks. PhD thesis. Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science 2005
19. AP Mazumdar, AS Sairam, in *Routing in Opportunistic Networks*. 1st edition, ed. by I Woungang, SK Dhurandher, A Anpalagan, and AV Vasilakos. On performance modeling of *ad hoc* opportunistic routing protocols (Springer, New York, 2013), pp. 223–250
20. S Biswas, R Morris, Opportunistic routing in multi-hop wireless networks. SIGCOMM Comput. Commun. Rev. **34**, 69–74 (2004)
21. W Feller, *An Introduction to Probability Theory and its Applications*. (John Wiley & Sons, Inc., Hoboken, 1968)
22. EXata + Cyber | SCALABLE Network Technologies. http://www.scalable-networks.com/content/exatacyber. Accessed 25 Sept 2013
23. AJ Coulson, AG Williamson, RG Vaughan, A statistical basis for lognormal shadowing effects in multipath fading channels. IEEE Trans. Commun. **46**(4), 494–502 (1998)
24. PPC Lee, V Misra, D Rubenstein, in *Proceedings of the 2nd international conference on Communication systems and Networks (COMSNETS'10)*. On the robustness of wireless opportunistic routing toward inaccurate link-level measurements (Bangalore, 5–9 Jan 2010), pp. 337–346
25. V Durairaj, P Kalla, in *Proceedings of the 8th International Conference on Theory and Applications of Satisfiability Testing (SAT 2005)*. Enhanced interior gateway routing protocol, Cisco white paper (St. Andrews, 19–23 June 2005), pp. 415–422
26. AP Mazumdar, AS Sairam, in *Proceedings of the TENCON 2011 - 2011 IEEE Region 10 Conference*. PBFS: a technique to select forwarders in Opportunistic Routing (Bali, 21–24 Nov 2011), pp. 149–153

doi:10.1186/1687-1499-2013-237
Cite this article as: Mazumdar and Sairam: **TOAR: transmission-aware opportunistic *ad hoc* routing protocol.** *EURASIP Journal on Wireless Communications and Networking* 2013 **2013**:237.