A New Multicast Opportunistic Routing Protocol for Wireless Mesh Networks

Amir Darehshoorzadeh and Llorenç Cerdà-Alabern

Computer Architecture Dep.
Univ. Politècnica de Catalunya
Barcelona, Spain
{amir,llorenc}@ac.upc.edu

Abstract. *Opportunistic Routing (OR) has been proposed to improve the efficiency of unicast protocols in wireless networks. In contrast to traditional routing, instead of preselecting a single specific node to be the next-hop forwarder, an ordered set of nodes (referred to as candidates) is selected as the next-hop potential forwarders. In this paper, we propose a new multicast routing protocol based on OR for wireless mesh networks, named Multicast OR Protocol (MORP). We compare our proposal with the well known ODMRP Multicast protocol. Our results show that Multicast-OR outperforms ODMRP, reducing the number of transmissions and increasing the packet delivery ratio.*

Keywords: Opportunistic routing; Multicast; Wireless

1 Introduction

Opportunistic Routing (OR) has been investigated in recent years as a way to increase the performance of unicast in multi-hop wireless networks. In OR, in contrast to traditional routing, instead of preselecting a single specific node to be the next-hop forwarder, an ordered set of nodes (referred to as candidates) is selected as the next-hop potential forwarders. More specifically, when the current node transmits a packet, all the candidates that receive the packet successfully will coordinate with each other to determine which one would actually forward the packet according to some criteria, while the other nodes will simply discard the packet. Previous research of OR mainly focused on developing various types of OR algorithms for unicast protocol and evaluating their performance. Multicast OR has received relatively few attention.

In this paper we propose a new multicast routing protocol based on OR. We will refer to our proposal as *Multicast Opportunistic Routing Protocol*, MORP. It opportunistically employs a set of forwarders to send a packet toward all destinations. The basic ideal of MORP is to form a candidates set to reach the destinations and based on the candidates which successfully receive the packet, selects a set of candidates as the forwarders to reach all destinations. Each forwarder is responsible for sending the packet to a subset of destinations. Indeed, based on the candidates that successfully receive the packet in each transmission,

MORP builds a multicast tree on the fly using OR and forwards the packet through the tree.

The mesh based protocols in multicast routing are more reliable than the tree based protocols. We have compared our new multicast opportunistic routing protocol (MORP) with the well known ODMRP multicast mesh protocol [13]. Our results show that Multicast-OR outperforms ODMRP, reducing the number of transmissions and increasing the packet delivery ratio.

The rest of the paper is organized as follows. Section 2 surveys the related work. In section 3 we describe ODMRP. Our proposal is described in section 4. In section 5 we show how our protocol works using an example. In section 6 both protocols are compared, and concluding remarks are given in section 7.

2 Related work

The majority of previous studies in opportunistic routing do not use it for multicast routing, and most of them are devoted to the selection of the candidates, the way of acknowledging packet reception and how to prevent, or at least reduce, duplicate transmissions.

Biswas and Morris proposed ExOR [2, 3], one of the first and most referenced OR protocols. In [18, 17] Zhong et al. proposed a new metric –*expected any-path transmission (EAX)*– that generalizes the single-path metric ETX [8] to an OR framework. They proposed a candidate selection and a prioritization rule based on it. They analyzed the efficacy of OR by using this metric and did a comparison using the link-level measurement trace of the Roofnet project [1]. In [9, 10] a distributed algorithm for computing minimum cost opportunistic routes, which is a generalization of the well-known Bellman-Ford algorithm, is presented. In [14] the key problem of how to optimally select the forwarder list is addressed, and an optimal algorithm (MTS) that minimizes the expected total number of transmissions is developed. In previous works we have proposed a discrete time Markov chain to analyze the performance that may be achieved using opportunistic routing [4], and in [5] we have studied the maximum performance that may be achieved using OR.

There are few works about using OR in multicast. MORE [6] is a MAC independent protocol that used both the idea of OR and network coding. It avoids duplicates transmission by randomly mixing packets before forwarding. It provides both unicast and multicast traffic. In [11] the source first creates the shortest path tree to reach all destinations based on the ETX of each link. Then the nodes not only receive packets from their father in the tree, but also can overhear packets from its sibling nodes. It uses random linear network coding to improve multicast efficiency and simplify node coordination. In [12] it is proposed an overlay multicast to adapt OR in wireless network.

3 ODMRP mechanism

The On Demand Multicast Routing Protocol (ODMRP) is a mesh based multicast protocol that establishes the routes and updates them by the source on-demand [13, 15]. While a source has data packets to send, it periodically broadcasts a *Join-Query* packet to the entire network. When a node receives a non-duplicate *Join-Query*, it stores the upstream node ID and rebroadcasts the packet. When the *Join-Query* reaches a multicast destination, the destination creates or updates the source entry in its *Member-Table*. While valid entries exist in the *Member-Table*, *Join-Table*s are broadcasted periodically to its neighbors.

When a node receives a *Join-Table*, it checks if its ID matches with the ID of the next node of one of the entries in the *Join-Table*. If it does, the node realizes that it is on the path to the source and thus is part of forwarding group. It then sets the *FG-Flag* and broadcasts its own *Join-Table*. The *Join-Table* is propagated by each forwarding group member until it reaches the multicast source. The forwarding group is a set of nodes in charge of forwarding multicast packets. A multicast receiver can also be a forwarding group node if it is on the path between a multicast source and another destination.

A multicast source can transmit packets to the destinations via the forwarding groups. When a node receives a data packet, it forwards the packet only if it is not duplicated and the *FG-Flag* for the multicast group of this node has not expired.

4 Multicast Opportunistic Routing Protocol

In this section we propose a new multicast routing protocol that we call *Multicast Opportunistic Routing Protocol*, MORP. Assume a network with N nodes and a multicast group M consisting of one source S and a set of destinations $\mathbf{D} = \{D_1, D_2, \cdots, D_n\}$. Denote $C^{j,D_i} = \{c_1, c_2, \cdots, c_n\}$ as the candidates set of node j to reach destination D_i (c_1 the highest priority, and c_n the least one), and $C^{j,\mathbf{D}}$ as the multicast candidates set of node j to reach the destinations in $\mathbf{D}$.

Before a transmission starts, each node in the network must compute C^{j,D_i} for each $D_i \in \mathbf{D}$, and store them in a *Candidate-Table*. This would be done using one of the candidates selection algorithms of the unicast opportunistic protocols that have been proposed in the literature (like ExOR [2]). Each time the source S wants to transmit a packet, the following three-way-handshaking is carried out: First the source inserts its multicast candidates set ($C^{S,\mathbf{D}}$), which is the union of all the candidates sets to reach $\mathbf{D}$: $\bigcup_{D_i \in \mathbf{D}} C^{S,D_i}$, in the data packet and transmits it. Each candidate which successfully receives the packet sends back an acknowledgment (ACK). After a period of time (T_{Ack}) the source checks if it received ACKs from enough candidates to reach all destinations. If no enough ACKs were received, it retransmits the packet. This is done up to a maximum number of retransmissions (MAX_{ReTx}). Then the source selects the candidates responsible to forward the packet (*forwarding set*), and chooses to which destinations each of them must care. This process is explained in section 4.1. We will refer to the set of destinations chosen for each forwarder c_j as

its *Bind-Destinations*, and denote it as $D^{'}_{c_j}$. Note that the *Bind-Destinations* for the source node is the multicast destinations set: $D'_S = \mathbf{D}$.

The source creates a set $\mathbf{F}$ with the IDs of the *forwarding set* and their *Bind-Destinations*. Then the source puts the set $\mathbf{F}$ in a control packet, that we will refer to as *ForwardingPacket*, and broadcasts it. Each candidate c_j that receives the *ForwardingPacket* and its ID is included in it, would forward the packet following the same rules as the source, except that its *Bind-Destinations* ($D^{'}_{c_j}$) will be used instead of $\mathbf{D}$. This process is continued until the packet reaches the destinations.

4.1 Candidate Coordination

Upon transmitting a packet, the node collects the received ACKs in an *Ack-Table*. When T_{Ack} expires, the *forwarding set* is computed as follows. If node j receives an ACK from one of the candidates in set C^{j,D_i} ($D_i \in \mathbf{D^{'}_j}$), it assumes that the packet can reach D_i, and the candidate with the highest priority to reach D_i which sent ACK is chosen as responsible to forward the packet toward D_i. Recall that the highest priority candidate is the candidate which has the least expected number of transmission to the destination. On the other hand, if j does not receive any ACK from candidates in C^{j,D_i} ($D_i \in \mathbf{D^{'}_j}$), it assumes that the packet can not reach D_i and retransmits the data packet. The node retransmits the data packet for at most MAX_{ReTx} times or until receiving ACKs from enough candidates to reach all destinations.

If j finds that it is possible to reach all destinations in $\mathbf{D^{'}_j}$, then it sends the *ForwardingPacket* announcement. In *ForwardingPacket*, j determines which candidates would actually forward the packet, and to which destinations (*Bind-Destinations*). Note that for each destination only one candidate would be chosen to forward the packet, and the same candidate can be chosen to forward the packet to more than one destination. If the number of retransmissions of a data packet reaches MAX_{ReTx} and there are not enough ACKs to reach all destinations, then the *ForwardingPacket* for the reachable destinations would be sent.

5 An Example of MORP

Figure 1 shows an example of MORP. The destinations set is $\mathbf{D} = \{D_1, D_2, D_3\}$ and the source of this multicast group is S. The table under each node represents its *Candidate-Table* to reach the destinations.

The multicast candidates set for the source S to reach its *Bind-Destinations* set ($D^{'}_S$) is $C^{S,D^{'}_S} = \{B, D_2, A\}$. When S wants to send a packet, it puts its multicast candidates set ($C^{S,D^{'}_S}$) in the data packet and sends it. It sets the timer T_{ACK} and waits for the ACKs from the candidates that have received the packet successfully. Assume that only the candidates A and B receive the data. Since this packet is not duplicated for A and B and also their IDs are included in it, they will send back an ACK to the source.

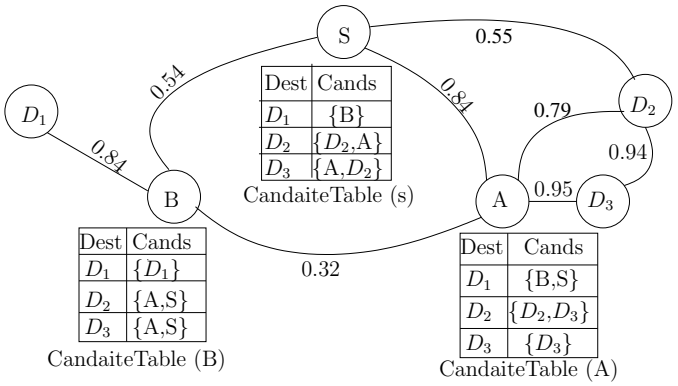

Fig. 1: Example of MORP.

When S receives ACKs from A and B, it stores their ID in its *ACK-Table*. When the T_{ACK} timer in the source expires, it uses the algorithm which has been described in section 4.1 to select and coordinate the forwarders. Since A and B sent ACK, S knows that for each destination at least one candidate received the data packet. Then S finds the best candidates that should forward the packet. The best candidates to reach D_1, D_2 and D_3 are B, A and A, respectively. Thus, S creates the *ForwardingPacket* with A and B as the forwarders and, $\{D_2, D_3\}$ and $\{D_1\}$ as the *Bind-Destinations* for A and B, respectively. Then S broadcasts the *ForwardingPacket*.

Upon receiving the *ForwardingPacket*, A and B know that they must forward the packet to $\{D_2, D_3\}$ and $\{D_1\}$, respectively. Let continue with node A. Node A creates its multicast candidates set for its *Bind-Destinations* ($\{D_2, D_3\}$), inserts the new multicast candidates set ($C^{A,D_A^{'}} = \{D_2, D_3\}$) in the data packet, starts the T_{ACK} timer and forwards the packet. Note that when this timer expires, A just checks the ACKs from the candidates for the destinations in $D_A^{'}$ (I.e. D_2 and D_3). Node B does the same for the destination D_1. Remark that one destination can act as a forwarder, if it is the best candidate for an other destination which sent ACK.

Note that when the data packets approach the destinations, the size of *Bind-Destinations* will decrease or remain unchanged. Thus, it is like MORP builds a tree on the fly, depending on the candidates that successfully receive the packet in each transmission.

6 Performance Evaluation

The simulation code has been implemented within the Global Mobile Simulation (GloMoSim) library [16]. In the simulation we have modeled a network with different number of nodes ($20 \leq N \leq 100$) placed randomly within a square with diagonal equal to 500 m. Each simulation runs for 300 seconds of simulation time. The IEEE 802.11 Distributed Coordination Function was used as the medium access control protocol and the channel capacity was 2 Mbps. Each point in the

plots was obtained averaging over 20 runs with different random node positions. The traffic generated by the source is Constant Bit Rate with 1 packet per second and 512 bytes of payload.

The number of multicast groups and sources is set to one in all scenarios. Destination nodes are chosen randomly. We have used different number of destinations ($2 \leq NumDest \leq 10$). Members join the multicast group at the start of the simulation and remain throughout the simulation. We have used ExOR as the candidate selection algorithm, fixing the maximum number of candidates $ncand = 2$ (see [7] for details).

For a more realistic simulation, we have used the shadowing propagation model with parameters $\beta = 2.7$ and $\sigma_{dB} = 6$ dBs. With these parameters the the link delivery probability is approximately 40 % at the distance of 150 m (see our previous works [4, 5, 7] for details).

To evaluate the performance of MORP, we compare it with ODMRP. We have changed the way that ODMRP creates the routes to adapt it with the shadowing propagation model. We have evaluated both protocols as a function of number of nodes and number of destinations. The measures of interest are:

- Data Delivery Ratio: The ratio of data packets received by destinations to the number of data packets sent by the source.
- Forwarding overhead: Total number of data packets transmitted over the total number of received packets.
- Control Packets overhead: The ratio of total number of control packets transmitted to the total data packets delivered.
- End-to-End delay: Average end-to-end delay of all data packets received by the destinations.

6.1 Data Delivery Ratio

Figure 2 shows the packet delivery ratio varying the number of nodes, but maintaining the diagonal of the area of the network equal to $D = 500$ m. For each point in the figure we have added error bars at 95% confidence interval. In this figure the number of destinations have been set to 5 ($NumDest = 5$). The results are shown varying the maximum number of retransmission (MAX_{ReTx}) of MORP. The legend MORP-ExOR(n) in figure 2 refers to MORP with $MAX_{ReTx} = n$.

We can see that using MORP with any number of MAX_{ReTx} outperforms ODMRP. Even if MORP does not retransmit any data packet (MORP-ExOR(1)), it achieves about 70% packet delivery ratio, while for ODMRP the delivery ratio is about 60%. This can be explained because the construction of the routes in ODMRP are subject to the random losses that may have the *Join-Query* packets. On the other hand, routes in MORP depends on the selection of the candidates sets, which is done taking into account the delivery probability of the links.

It is obvious that the more retransmissions are allowed in MORP, the higher will be the delivery ratio of the data packets. We can see that the differences

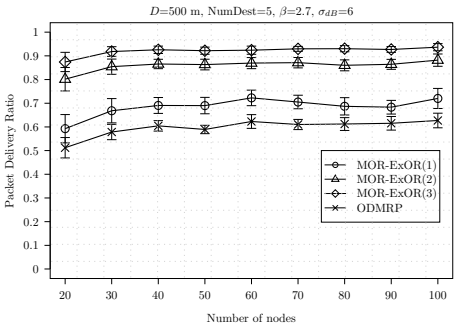

Fig. 2: Delivery ratio for 5 destinations

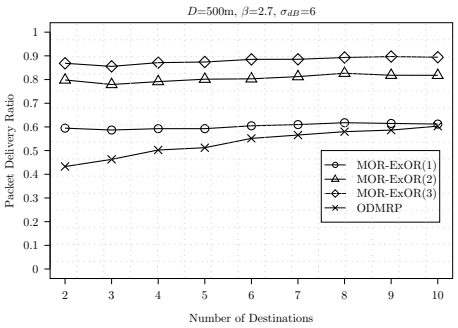

Fig. 3: Delivery ratio with N=20 nodes varying the number of destinations

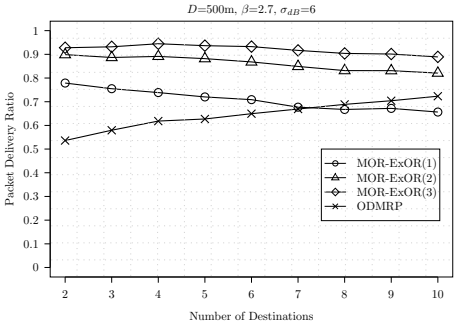

Fig. 4: Delivery ratio with N=100 nodes varying the number of destinations

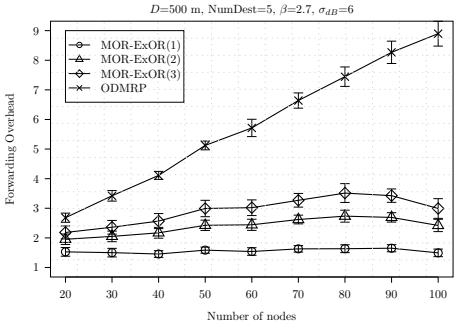

Fig. 5: Forwarding overhead for 5 destinations

between MORP-ExOR(1) and two other experiments (MORP-ExOR(2) and MORP-ExOR(3)) are about 17% and 23%, respectively.

Figures 3 and 4 have been obtained respectively with a total number of nodes equal to $N = 20$ and $N = 100$, representing a low and high density network, and varying the number of destinations: $NumDest = 2, 3, ..., 10$.

Figure 4 shows that the delivery ratio of MORP-ExOR(1) for 8, 9 and 10 destinations is a bit less than ODMRP (about 2%, 3% and 5%, respectively). This comes from the fact that the more destinations are, the larger are the forwarding groups in ODMRP. Therefore the packet delivery ratio in ODMRP increases by increasing the number of destinations, however, at the cost of increasing too the forwarding overhead. Nevertheless, increasing MAX_{ReTx} in MORP to 2 and 3, increases the delivery ratio to 82% and 88%, respectively, which outperforms the 72% obtained with ODMRP.

6.2 Forwarding Overhead

In this section we compare the forwarding overhead of MORP and ODMRP. Recall that we define the forwarding overhead as the total number of data packet transmissions by any node, over the total number of packets received by any destination.

Figure 5 shows the forwarding overhead varying the number of nodes in the case of 5 destinations. ODMRP periodically floods a data packet together with a *Join-Query* packet. I.e., it piggybacks the *Join-Query* information on the data packet periodically to update the membership information. For this reason the forwarding overhead of ODMRP is dominated by the flooding packets. Therefore, the higher is the node density of the network, the higher is the ODMRP forwarding overhead. On the other hand, the forwarding overhead of MORP is rather insensitive to the network density. This is because using opportunistic routing, as in MORP, only some useful nodes are selected as candidates to forward the packets, and thus, the number of forwarders is limited. Obviously, the higher is the number of retransmissions of data packets allowed in MORP, the higher will be the forwarding overhead. Nevertheless, figure 5 shows that in all cases the forwarding overhead of MORP is less than in ODMRP.

Figures 6 and 7 show more results of the forwarding overhead, varying the number of destinations for a low and high dense network. These two figures depict that in a dense network MORP is less sensitive to the number of destination. This is because having more nodes in the network, allows the candidates selection algorithm to better choose candidates. On the other hand, as mentioned before, the forwarding overhead in ODMRP is closely related to the network density.

Note that in figure 6 the forwarding overhead for ODMRP with 10 destinations is about 1.27 and for MORP-ExOR(2) is about 1.16. Although these two values are close to each other, figure 3 shows that the delivery ratio of ODMRP for 10 destinations is about 60% while for MORP-ExOR(2) is about 81%, so, MORP outperforms ODMRP.

On the other hand, figure 7 shows that the forwarding overhead of ODMRP with 10 destinations is about 4.04, while in MORP-ExOR(1) it is about 0.90. As we have mentioned in section 6.1, in this case the delivery ratio in ODMRP was slightly larger than in MORP (see figure 4). We see now that this is at cost of having a forwarding overhead about 4.4 times larger than in MORP-ExOR(1).

6.3 Control Packets Overhead

In this section we compare the signaling overhead of MORP and ODMRP. To do so, we count as control packets for ODMRP the *Join-Query*, *Join-Table* and ACK packets, and for MORP the *ForwardingPacket* and ACK packets.

Figures 8 and 9 show the control overhead varying the number of destinations for low and high density networks, respectively. As the number of destinations increases, the control overhead decreases in both protocols. In MORP, the higher is the maximum number of retransmissions (MAX_{ReTx}), the higher is the number of ACKs and *ForwardingPacket*s. However, in figure 8 the control overhead

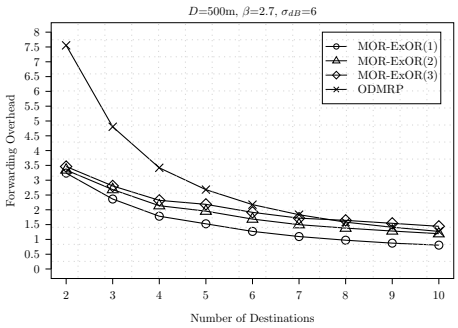

Fig. 6: Forwarding overhead varying the number of destinations. N=20.

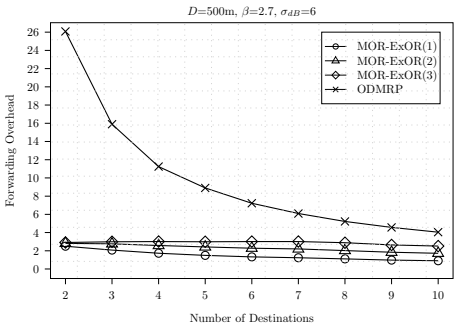

Fig. 7: Forwarding overhead varying the number of destinations. N=100.

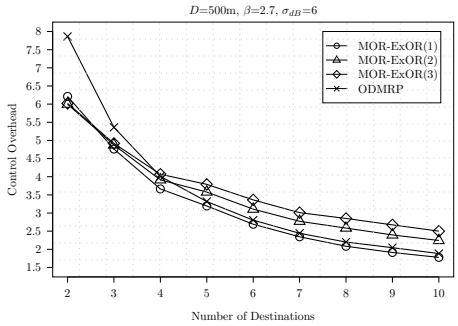

Fig. 8: Control overhead varying the number of destinations. N=20.

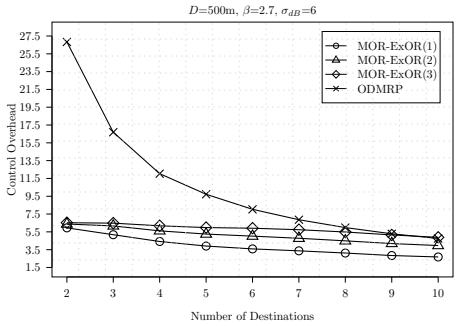

Fig. 9: Control overhead varying the number of destinations. N=100.

of MORP in the case of $MAX_{ReTx} = 2$ or 3 is only a bit higher than in ODMRP. On the other hand, its packet delivery ratio is much better than in ODMRP (see figure 3).

Additionally, figures 8 and 9 show that the control overhead of MORP with any number of retransmissions is much less sensitive to the number of destinations than in ODMRP.

6.4 End-To-End Delay

Figures 10 and 11 show the average end-to-end delay for different number of destinations with $N = 20$ and $N = 100$ nodes, respectively.

Recall that in MORP there is a three-way-handshaking each time a node transmits a data packet. Thus, as expected, the end-to-end delay in MORP is higher than in ODMRP. Comparing figures 10 and 11 we can see that end-to-end delay decreases in a dense network for ODMRP and MORP-ExOR(1). This is because in a dense network both protocols can find better routes. For MORP-ExOR(2) and MORP-ExOR(3), delays increase with the number of destinations.

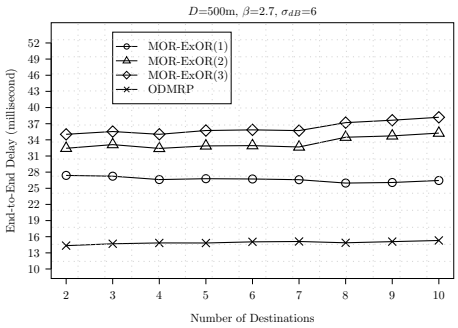

Fig. 10: End-To-End delay varying the number of destinations. N=20.

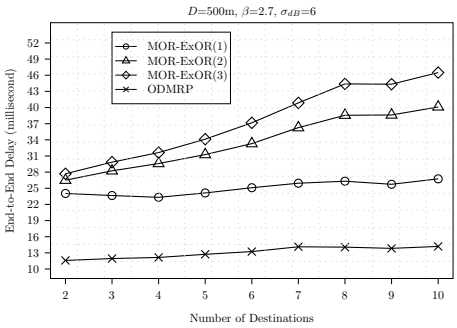

Fig. 11: End-To-End delay varying the number of destinations. N=100.

This comes from the fact that candidates set may be different to reach different destinations. Thus, more retransmissions are required until all necessary candidates receive the data packets.

7 Conclusion

In this paper we propose a new multicast protocol based on opportunistic routing that we call *Multicast Opportunistic Routing Protocol*, MORP. MORP uses a three-way-handshaking, where the node sending the data packet chooses the forwarders and a subset of destinations to which they have to send the data packets.

We have compared our protocol with the well known mesh-based multicast routing protocol called ODMRP. In our simulations we have measured the packet delivery ratio, forwarding overhead, control overhead and end-to-end delay of both protocols.

Simulations results show that MORP outperforms ODMRP in terms of packet delivery ratio and forwarding overhead. The end-to-end delay of MORP is higher than in ODMRP, but still acceptable for real time applications. We conclude that MORP, and opportunistic routing in general, is a convenient technique to be used in multicast protocols for wireless mesh networks.

Acknowledgments

This work was supported by the Spanish government and Generalitat de Catalunya through projects TIN2010-21378-C02-01 and 2009-SGR-1167, respectively, and by the European Commission through the NoE EuroNF.

References

1. MIT roofnet. http://pdos.csail.mit.edu/roofnet.

2. S. Biswas and R. Morris. Opportunistic routing in multi-hop wireless networks. *ACM SIGCOMM Computer Communication Review*, 34(1):69–74, 2004.
3. S. Biswas and R. Morris. ExOR: opportunistic multi-hop routing for wireless networks. *ACM SIGCOMM Computer Communication Review*, 35(4):133–144, 2005.
4. L. Cerdà-Alabern, V. Pla, and A. Darehshoorzadeh. On the performance modeling of opportunistic routing. In *MobiOpp '10: Second International Workshop on Mobile Opportunistic Networking*, pages 15–21, New York, NY, USA, 2010. ACM.
5. L. Cerdà-Alabern, V. Pla, and A. Darehshoorzadeh. On the maximum performance in opportunistic routing. In *IEEE WoWMoM 2010*, Montreal, Canada, 2010.
6. S. Chachulski, M. Jennings, S. Katti, and D. Katabi. Trading structure for randomness in wireless opportunistic routing. In *SIGCOMM*, pages 169–180, New York, NY, USA, 2007. ACM.
7. A. Darehshoorzadeh and L. Cerdà-Alabern. Candidate selection algorithms in opportunistic routing. In *PM2HW2N '10: 5th ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*, pages 48–54, New York, NY, USA, 2010. ACM.
8. D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. *Wireless Networks*, 11(4):419–434, 2005.
9. H. Dubois-Ferriére, M. Grossglauser, and M. Vetterli. Least-cost opportunistic routing. In *Allerton Conference on Communication, Control, and Computing*, 2007.
10. H. Dubois-Ferriére, M. Grossglauser, and M. Vetterli. Valuable detours: Least-cost anypath routing. *Networking, IEEE/ACM Transactions on*, 19(2):333 –346, april 2011.
11. D. Koutsonikolas, Y. Hu, and C.-C. Wang. Pacifier: High-throughput, reliable multicast without "crying babies" in wireless mesh networks. In *INFOCOM 2009, IEEE*, pages 2473 –2481, 2009.
12. T. Le and Y. Liu. Opportunistic overlay multicast in wireless networks. In *GLOBECOM*, pages 1–5, 2010.
13. S.-J. Lee, M. Gerla, and C.-C. Chiang. On-demand multicast routing protocol. In *Wireless Communications and Networking Conference, 1999. WCNC. 1999 IEEE*, pages 1298 –1302 vol.3, 1999.
14. Y. Li, W. Chen, and Z.-L. Zhang. Optimal forwarder list selection in opportunistic routing. In *Mobile Adhoc and Sensor Systems. MASS '09. IEEE 6th International Conference on*, pages 670 –675, oct. 2009.
15. M. Naderan-Tahan, A. Darehshoorzadeh, and M. Dehghan. Odmrp-lr: Odmrp with link failure detection and local recovery mechanism. In *Computer and Information Science, 2009. ICIS 2009. Eighth IEEE/ACIS International Conference on*, pages 818 –823, jun. 2009.
16. X. Zeng, R. Bagrodia, and M. Gerla. GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks. In *12th Workshop on Parallel and Distributed Simulation (PADS'98)*, pages 154–161. IEEE Computer Society, May 1998.
17. Z. Zhong and S. Nelakuditi. On the efficacy of opportunistic routing. In *SECON '07*, pages 441–450, June 2007.
18. Z. Zhong, J. Wang, S. Nelakuditi, and G.-H. Lu. On selection of candidates for opportunistic anypath forwarding. *SIGMOBILE Mob. Comput. Commun. Rev.*, 10(4):1–2, 2006.