Reducing Data-Overhead of Mesh-based Ad hoc Multicast Routing Protocols by Steiner Tree Meshes

Pedro M. Ruiz
Dept. Information and Communications Engineering
University of Murcia
Fac. Informatica, Campus de Espinardo s/n.
E-30071 Espinardo, Murcia (Spain)
E-mail: pedrom@dif.um.es

Antonio F. Gomez-Skarmeta
Dept. Information and Communications Engineering
University of Murcia
E-30001, Aptdo. 4021, Murcia (Spain)
E-mail: skarmeta@dif.um.es

***Abstract*— We study the problem of reducing data overhead of mesh-based multicast ad hoc routing protocols by reducing the number of forwarding nodes. We show that minimizing the number of forwarding nodes is equivalent to the problem of finding the minimal cost multicast tree. In addition, we demonstrate the problem to be NP-complete by a transformation to the Steiner tree problem. We propose a distributed heuristic algorithm based on the epidemic propagation of the number of forwarding nodes. Our simulation results show that the proposed heuristic, when implemented into ODMRP, is able to offer similar performance results and a lower average latency while improving the forwarding efficiency in around a 40-50% with respect to the original ODMRP.**

I. INTRODUCTION

A mobile ad hoc network consists of a set of mobile nodes which are free to move and are interconnected through wireless interfaces. Nodes which are not able to communicate directly, use multihop paths using other intermediate nodes in the network as relays. So, a mobile ad hoc node can act both as a mobile router and a mobile host. Their completely distributed nature and their ability to operate without depending upon the deployment of any infrastructure, makes them ideal component of future mobile computing scenarios. These scenarios include among others emergency situations, battlefield assistance and search and rescue operations. Hence, the interest in mobile ad hoc networks is expected to increase in the future.

Multicast is one of the areas in mobile ad hoc networks which is to play a key role in future wireless networks. Key to this is the fact that most of the application scenarios for mobile ad hoc networks are strongly based on many-to-many interactions and they require a high degree of collaboration among terminals. Many services such as multimedia applications, service discovery and many other bandwidth-avid applications can strongly benefit from the underlying support of efficient multicast communications.

The problem of the efficient distribution of traffic from a set of senders to a group of receivers in a datagram network was already studied by Deering [1] in the late 80's. Several multicast routing protocols like DVMRP [2], MOSPF [3], CBT [4] and PIM [5]) have been proposed for IP multicast routing in fixed networks. However, these protocols are not able to perform well in highly mobile and topology changing scenarios such as ad hoc networks. The main reason is that they are based on multicast trees without any local link repairing mechanism. Thus, the cost in terms of the control overhead which is required to recompute the whole multicast tree whenever one of the links in the tree breaks, makes unreasonable their deployment in an ad hoc network.

Several multicast routing solutions specifically designed for ad hoc networks have been proposed in the literature [6]. In general, these protocols can be classified into two groups: tree-based and mesh-based approaches. Tree-based schemes construct a multicast tree from each of the sources to all the receivers. Examples of protocols following this approach are AMRIS [7], MAODV [8], LAM [9] and ADMR [10]. The main advantage of using a tree as the underlying forwarding structure is that the number of forwarding nodes tends to be reduced (although not optimized). However, a tree is very fragile when there is a high mobility in the network. Mesh-based approaches like ODMRP [11] and CAMP [12], by using additional links in their underlying forwarding structure, manage to deal with mobility very efficiently. The main drawback associated to the use of a mesh, is that the additional paths which are created can make an excessive consumption of network resources when sending data packets. Hybrid approaches ([14], [15]) try to combine the robustness of mesh-based ad hoc routing and the low overhead of tree-based protocols. Finally, there are stateless multicast protocols ([16], [17]) in which there is no need to maintain a forwarding state on the nodes. For instance, if the nodes to traverse are included in the data packets themselves.

The usual metrics used in the literature to assess the effectiveness of a multicast ad hoc routing protocol are usually the packet delivery ratio and the control overhead per data message delivered. Those are clearly good metrics to assess the performance and the efficiency respectively. However, in the author's opinion data overhead is also a very important factor to consider in the evaluation of multicast ad hoc routing protocols. Data overhead is related to the cost associated to the use of non-optimal multicast trees. Therefore, it has a strong impact on the overall scalability and network capacity which can be achieved with a particular routing protocol.

For ad hoc networks, most of the works in the literature devoted to the improvement of multipoint forwarding efficiency for routing protocols have been related to the particular case of flooding (i.e. the broadcast storm problem). Only a few papers study those mechanisms for multicast ad hoc routing. Lim and Kim [25] analyzed the problem of minimal multicast trees in ad hoc networks, but they defined several heuristics based on the minimum connected dominating set (MCDS) which are only valid for flooding. Lee and Kim [26] worked on a solution to reduce the overhead using a probabilistic approach. However, the overhead reductions where lower than the results we have obtained, and their fixed path selection probability makes their proposal unable to perform well under different network conditions.

We propose a new approach for the adaptive construction of the multicast forwarding mesh which reduces the number of forwarding nodes when there is enough reliability in the existing mesh. In existing mesh-based protocols the multicast mesh consists of the shortest path trees plus a number of backup links. To reduce the number of forwarding nodes the proposed scheme builds the forwarding mesh upon Steiner trees. Obviously, given the NP-completeness of the Steiner tree problem, we approximate the Steiner trees used in the mesh. For that approximation we propose a distributed heuristic based on the epidemic propagation of the number of forwarding nodes. Simulation results show that the proposed heuristic, when incorporated into the ODMRP mesh-based ad hoc routing protocol, yields around a 40-50% improvement in the forwarding efficiency without a noticeable impairment in the overall performance.

The remainder of the paper is organized as follows: section II discusses the problem of data overhead minimization. Section III describes our network model, and shows the validity of our problem formulation. The description of the proposed algorithm is given in section IV. In section V we explain our simulation results. Finally, section VI provides some discussion and conclusions.

II. DATA OVERHEAD IN AD HOC MULTICAST ROUTING

The goal of multicast routing protocols in ad hoc networks is finding a set of relay nodes so that data packets sent out by multicast sources can be delivered to multicast receivers. The paths defined by the union of all these nodes may resemble different forwarding structures such shortest path trees, shared trees, minimal steiner trees, acyclic meshes, etc. In general, the underlying forwarding structure is protocol-specific because it strongly depends on the path creation process implemented by that particular protocol. Forwarding nodes can be defined as those nodes which are selected by the routing protocol to be in the path between any source and any receiver. Note that even a source or a receiver can also be a forwarding node. Fig. 1 shows a multicast tree in which we identify forwarding nodes by a double circle. Wider lines represent the forwarding structure induced by the selected set of forwarding nodes.

There are two basic approaches to build multicast trees: shortest path trees and shared trees [18]. These trees are defined by the links between the forwarding nodes. For instance Fig. 1 shows the multicast tree induced by the forwarding nodes (in double circle). Given a multicast source s, a shortest path tree is formed by the aggregation of the shortest paths from any receiver r to s. The main advantage of this kind of trees is that each destination receives multicast data through its best route, which usually means that the latency from s to each r is also reduced. However, these trees are not optimal in terms of the overall number of forwarding nodes which are selected, incurring therefore in a higher data overhead.

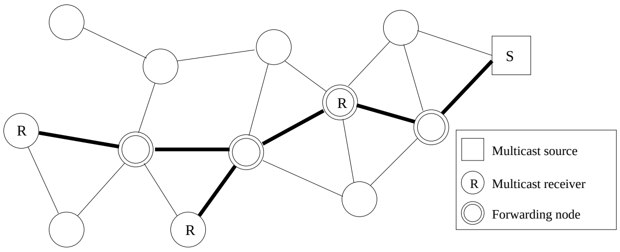


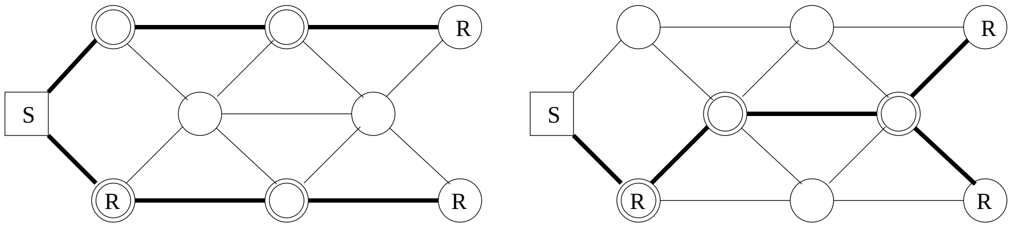
Fig. 1. Example of tree-based multicast forwarding structure

A second variant are the so-called shared trees. Shared trees try to reduce the cost of the multicast tree by reducing the number of links which are required to connect sources and receivers. This is done by selecting the links in the tree which are useful to a bigger number of receivers. Of course, in the resulting tree individual paths from sources to receivers might not be optimal.

The problem of finding a minimum cost multicast tree is well-known as the minimum Steiner tree problem. Karp [19] demonstrated by a transformation from the exact cover by 3-sets problem that this problem is NP-complete even when every link has the same cost. There are some heuristic algorithms to compute minimal Steiner trees. For instance, the MST algorithm ([20], [21]) provides a 2-approximation, and Zelikovsky [22] proposed an algorithm which obtains a 11/6-approximation. However, given the complexity of computing this kind of trees in a distributed way, most of the existing multicast routing protocols use shortest path trees, which can be easily computed in polynomial time.

Fig. 2 shows different multicast trees connecting the source S to the set of receivers R. As it is depicted in Fig. 2(a), the union of the shortest path trees result in 4 forwarding nodes and 6 links whereas the Steiner tree (see Fig. 2(b)) has 3 forwarding nodes and 5 links. We can see from the figure, how the Steiner tree tries to minimize the number of non-terminal nodes (i.e. nodes which are not either senders or receivers) which take part in the tree.

In our particular case, we are not interested in using the minimal cost multicast tree for routing. In fact, it would not work well with mobility because it does not have backup links. What we are interested is in using those minimal cost multicast trees as a basis to build a multicast mesh. The goal is to reduce the data overhead without loosing the resilience provided by backup links. We demonstrate in the next section that finding such a minimum cost tree can be done by finding a tree which minimizes the number of forwarding nodes when all the links



(a) Shortest Path Tree, 4 forw. nodes, 6 links

(b) Steiner Tree, 3 forw. nodes, 5 links

Fig. 2. A comparison of an SPT and an Steiner tree over the same ad hoc network

have the same cost. This is a valid assumption for an ad hoc network in which the routing protocol does not use an explicit link metric. Thus a forwarding mesh built upon Steiner trees is expected to have a lower data overhead than a forwarding mesh built upon shortest path trees.

III. NETWORK MODEL AND PROBLEM FORMULATION

A. Network model

We represent the ad hoc network as an undirected graph $G(V, E)$ where V is the set of vertices and E is the set of edges. We assume that the network is two dimensional (every node $v \in V$ is embedded in the plane) and mobile nodes are represented by vertices of the graph. Each node $v \in V$ has a transmission range r. Let $dist(v_1, v_2)$ be the distance between two vertices $v_1, v_2 \in V$. An edge between two nodes $v_1, v_2 \in V$ exists iif $dist(v_1, v_2) \leq r$ (i.e. v_1 and v_2 are able to communicate directly). In wireless mobile ad hoc networks some links may be unidirectional due to different transmission ranges. However, given that lower layers can detect and hide those unidirectional links to the network layer, we only consider bidirectional links. That is, $(v_1, v_2) \in E$ iif $(v_2, v_1) \in E$.

B. Problem formulation

We are interested in finding for each source the multicast tree with the minimal cost. The multicast forwarding mesh will defined by the union of all those trees with minimal cost. We will formulate the problem in terms of the minimization of the number of forwarding nodes. As we demonstrate in theorem 3.1 it is an equivalent formulation. So, the problem of finding the multicast tree with the minimal number of forwarding nodes can be formulated as follows:

Let $C : T \rightarrow \mathbb{Z}^+$ be a function so that given a tree T, $C(T)$ is the number of forwarding nodes in T which are not sources or receivers. Given a graph $G = (V, E)$, a source node $s \in V$, a set of receivers $R \subset V$, and given V' defined as $V' = R \cup \{s\}$ so that $V' \subseteq V$, find a tree $T^* \subset G$ such that the following conditions are satisfied:

1) $T^* \supseteq V'$
2) $C(T^*)$ is minimum

From the condition of T^* being a tree it is obvious that it is connected, which combined with condition 1) establishes that T^* is a multicast tree. Condition 2) establishes the optimality of the tree. In theorem 3.1 we show that under this formulation T^* is the minimum cost multicast tree.

Theorem 3.1: Let $G = (V, E, \omega)$ be an undirected graph with nonnegative edge weights so that $\omega(e) = k, k > 0$ for every $e \in E$. Let $s \in V$ be a multicast source, $R \subset V$ be the set of receivers and let V' be defined as $V' = R \cup \{s\}$. Under these conditions the tree $T^* \subset G$ with the minimum number of forwarding nodes which contains all the nodes in V' is also the minimum cost tree containing the vertices in V'.

Proof: We will demonstrate that it cannot exist another tree $T^{'}$ having a lower cost than T^* so that $C(T^{'}) \geq C(T^*)$. Let assume that there is a tree $T^{'} \subset G$ which contains all the nodes in V' whose cost is lower than the cost of T^* having a greater number of forwarding nodes. Let $E_{T'}$ and E_{T^*} be the edge set of $T^{'}$ and T^* respectively. So, by the definition of the cost of a tree, for every $e^{'} \in T^{'}$ and $e^* \in T^*$:

$$\sum_{i=1}^{|E_{T'}|} \omega(e_i^{'}) < \sum_{i=1}^{|E_{T^*}|} \omega(e_i^*)$$

Provided that $\omega(e) = k, k > 0$ for every $e \in E$, the previous relation can be expressed as follows:

$$k|E_{T'}| < k|E_{T^*}| \Rightarrow |E_{T'}| < |E_{T^*}|$$

As both T' and T^* are trees, it is satisfied that $n = m - 1$ being n the number of edges and m the number of vertices. In addition, provided that the number of vertices in a tree T is, by definition of $C(T)$, $C(T) + |R| + 1$.

$$C(T^{'}) + |R| < C(T^*) + |R| \Rightarrow C(T^{'}) < C(T^*)$$

This contradicts our original assumption that $C(T^{'}) \geq C(T^*)$ and demonstrates the theorem. ■

In addition, in theorem 3.2 we demonstrate that the formulated problem is NP-complete, which fully justify the heuristic algorithm proposed in the next section.

Theorem 3.2: Under the conditions of theorem 3.1, the problem of finding a tree $T^* \subset G$ such that $T^* \supseteq V'$ and $C(T^*)$ is minimized is NP-complete.

Proof: Lets consider the particular case in our problem in which every edge has the same nonnegative weight k. Under that assumption, by theorem 3.1 the optimal solution to our problem T^* is also the solution to the problem of finding the minimal Steiner tree. The minimal Steiner tree problem is known to be NP-complete even if the weight of every edge is the same. Thus, our problem cannot be solved in polynomial time unless P=NP. ■

IV. PROPOSED ALGORITHM

Given the NP-completeness of the problem, in the next subsections we describe the proposed algorithm to approximate minimal multicast trees, we present the adaptive mesh creation process as well as our integration into the ODMRP protocol.

A. Steiner tree heuristic

Given the results from theorem 3.1, we can approximate minimal Steiner tree by trying to minimize the number of forwarding nodes required to connect each source and all the receivers. To achieve that, we propose a distributed counting process inspired on epidemic algorithms. The basic idea is that each ad hoc node will propagate during the path creation process information about the number of non-forwarding nodes in its path to the source. So, each time a source wants to discover or refresh a route initializes the counter to zero. Whenever an intermediate node propagates such a control message, it modifies the counter as follows:

- If node is not forwarder node then increment the counter by one.
- If node is forwarder node then do not increase the counter.
- If node is a receiver then set the counter to one if it is not forwarder or to zero otherwise.

Thus, a control message with a lower counter value is associated to a route which produces a lower number of forwarding nodes. Then by selecting those routes, the resulting multicast tree becomes an approximation of a Steiner tree. The detailed algorithm is given in algorithm 1.

Algorithm 1 Epidemic propagation of FNCount

```
BestRR ← $null$
loop
  Receive route request packet RR
  if RR.seqno $>$ BestRR.seqno then
    BestRR ← RR
    if forwarder node $and$ not receiver then
      NewRR.FNCount ← RR.FNCount
    else if forwarder node $and$ receiver then
      NewRR.FNCount ← 0
    else if not forwarder $and$ receiver then
      NewRR.FNCount ← 1
    else
      NewRR.FNCount ← RR.FNCount+1
    end if
    Schedule sending of NewRR
  else if RR.seqno $=$ BestRR.seqno $and$ RR.FNCount $<$ BestRR.FNCount then
    if NewRR not sent out yet then
      BestRR ← RR
      if forwarder node $and$ not receiver then
        NewRR.FNCount ← RR.FNCount
      else if forwarder node $and$ receiver then
        NewRR.FNCount ← 0
      else if not forwarder $and$ receiver then
        NewRR.FNCount ← 1
      else
        NewRR.FNCount ← RR.FNCount+1
      end if
    end if
  end if
end loop
```

B. Adaptive mesh construction

As we explained before, algorithm 1 offers a low overhead route between a sender and a set of receivers by approximating a Steiner tree. However, due to the lack of redundancy of these trees, we are interested in using low cost multicast meshes built upon several of these Steiner trees.

Moreover, we are interested in adaptively controlling the redundancy which is introduced when creating the multicast mesh. In addition, we want the adaptive approach to be general enough and lightweight to compute so that the operation of the multicast routing protocol is not severely changed. This is why options like monitoring the neighbors with periodic beacons to estimate the network mobility have not been considered. These periodic beacons require additional overhead, and make substantial changes to the routing protocol.

An interesting aspect to consider is how the number of forwarding nodes is affected as the number of Steiner trees to join increases. For a single Steiner tree T_1, it is obvious that the number of forwarding nodes will be the minimum with no redundancy. If we consider a second Steiner tree T_2 and we build a mesh upon both of them denoted by $T_1 \oplus T_2$, then the number of forwarding nodes is $C(T_1) + C(T_2) - C(T_1 \cap T_2)$. The real number of forwarding nodes added by T_2 is then $C(T_2) - C(T_1 \cap T_2)$. In the general case of n trees $T_1 \dots T_n$, we find that the number of forwarding nodes added by the tree T_i is computed according to (1).

$$FN_i = C\left(T_i\right) - C\left(T_i \cap \left(T_1 \oplus \cdots \oplus T_{i-1}\right)\right) \qquad (1)$$

Provided the number of vertices ($|V|$) in the graph $G = (V, E)$ representing the whole ad hoc network is fixed, then as the number of Steiner trees in the union increases, the term $C(T_n \cap (T_1 \oplus \cdots \oplus T_{n-1}))$ also increases. This is because the probability that a node has been already considered in any of the previous Steiner trees is bigger. This can be easily shown by considering a Bernoulli experiment which is repeated as many times as the number of trees incorporated into the mesh. So, for smaller values of n we see that FN has a big increasing rate because the probability that other Steiner trees considered the same nodes in T_n is small. However, as n reaches some value n_1, the increasing rate of FN is reduced as n increases. This is because the probability of the event that forwarding nodes in T_n where already considered by any of the Steiner trees T_i for $i < n$ increases. This is also confirmed by the simulation results in the next section. So, for a higher number of sources (i.e. number of Steiner trees) the redundancy of the mesh is also bigger although the increasing rate is not linear. So, when the number of sources is big, almost no additional redundancy is required and when the number of sources is small, a bigger number of additional links might be required to cope with mobility.

The proposed mechanism to adaptively control the redundancy of the mesh uses a probabilistic path selection at every node. A node will select the shortest path with a probability p and the one minimizing the number of forwarding nodes with a probability $1 - p$. The key is to find a proper value of p so that an appropriate amount of redundancy is added. That is, that for a lower number of sources it reduces redundancy moderately as $|S|$ increases whereas reduces redundancy very fast redundancy for a greater number of sources. To achieve that behavior we have selected the value of p as shown in (2).

$$p = \frac{1}{1 + |S|^2} \tag{2}$$

C. Integration with ODMRP

For the integration with ODMRP we have had to consider three main aspects: the integration of algorithm 1 in ODMRP's control messages, the changes to the route selection process and the avoidance of cycles.

The integration with ODMRP's control messages has been almost immediate. The only change has been the introduction of a new field in JOIN_QUERY messages in which the $FGCount$ field is being updated as well as the addition of an "aggregation time" so that an ODMRP node can wait for a short period of time before propagating a JOIN_QUERY. This is done so that the node can receive other routes with lower cost than the source path tree. If no other route is received in that time then the shortest path route is propagated.

The route selection process has been changed to the probabilistically route selection explained before. That is, a node always select a fresher route rather than an old one. In case of two equally fresh routes with different cost it will select the one with the lower cost with a probability p according to (2).

Finally, as it may happen that two nodes select one each other as next hop to a source because both of them has the same cost (e.g. if both are receivers) then a tie resolution method has been proposed. In that case, the node with the lowest ID accepts being a forwarding node for the other node, and selects another neighbor towards the source. To detect that situation, a new field is added to the JOIN_QUERY message which includes the neighbor that the sender of the JOIN_QUERY message selected as its next hop towards the source. This was not a problem in the original specification of ODMRP because alternative routes (different from the shortest path) were detected as duplicate control packets and were not processed.

V. SIMULATION RESULTS

In order to assess the effectiveness of our proposed heuristic we have implemented it as part of a modified ODMRP protocol. We have simulated both the original version and the modified one of ODMRP in the Monarch extensions [24] to the NS-2 [23]. ODMRP has been selected as the mesh-based multicast routing protocol because it is very well-documented in the literature where it is shown to offer very good performance results in terms of packet delivery ratio.

A. Simulation methodology and scenarios

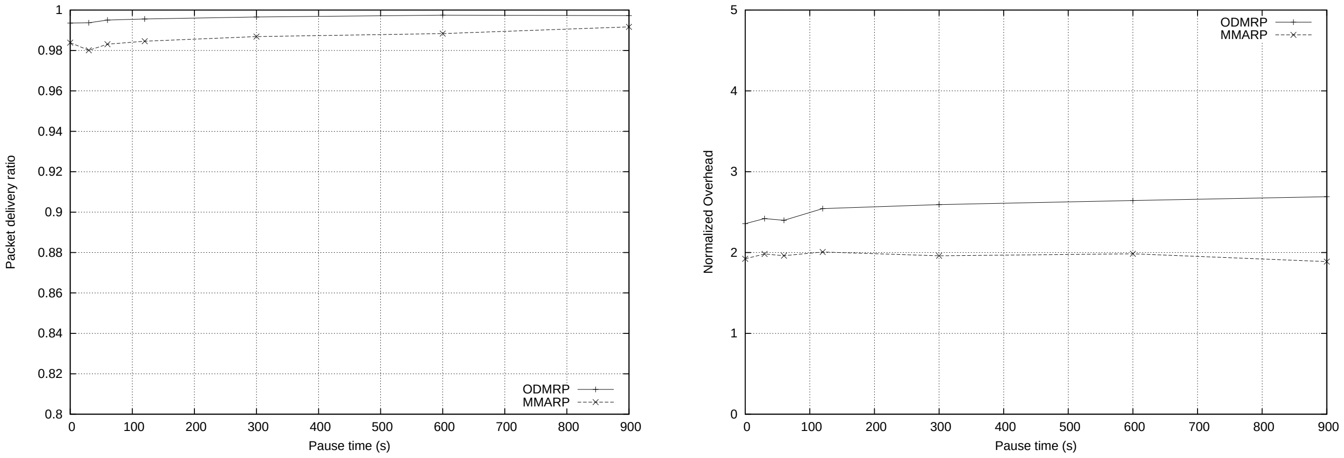
The simulated scenario consists of 100 mobile nodes randomly distributed over an area of 1200x800m^2. The radio channel capacity for each mobile node is 2Mb/s, using the IEEE 802.11b DCF MAC layer and a communication range of 250 m. Each of the approaches has been evaluated over the same pre-generated set of 210 scenarios with varying movement patterns and traffic loads. Mobile nodes move using a random waypoint model with changing pause times. Nodes start the simulation being static for *pause time* seconds. Then they pick up a random destination inside the simulation area and start moving to the destination at a speed uniformly distributed between 0 and 20 m/s (mean speed = 10m/s). After reaching its destination this behavior is continously repeated until the end of the simulation. Seven different pause times were used: 0, 30, 60, 120, 300, 600, and 900 seconds. A pause time of 0 seconds corresponds to a continuous motion whereas a pause time of 900 seconds corresponds to a static scenario. For each of these pause times 10 different scenarios where simulated. The results were obtained as the mean values over these 10 runs to guarantee a fair comparison among the alternatives.

Nine different traffic loads where tested consisting of 1, 2 and 5 CBR sources for the same multicast group, and 5, 15 and 30 receivers. The duration of each simulation run is 900 seconds. Each of these CBR sources start sending data and receivers join the multicast group at an uniformly distributed time within the first 180 seconds of the simulation. Each of the sources generates 330 bytes data packets at a rate of 5 packets per second (13.2 Kb/s), which resembles a GSM audio communication.

B. Performance metrics

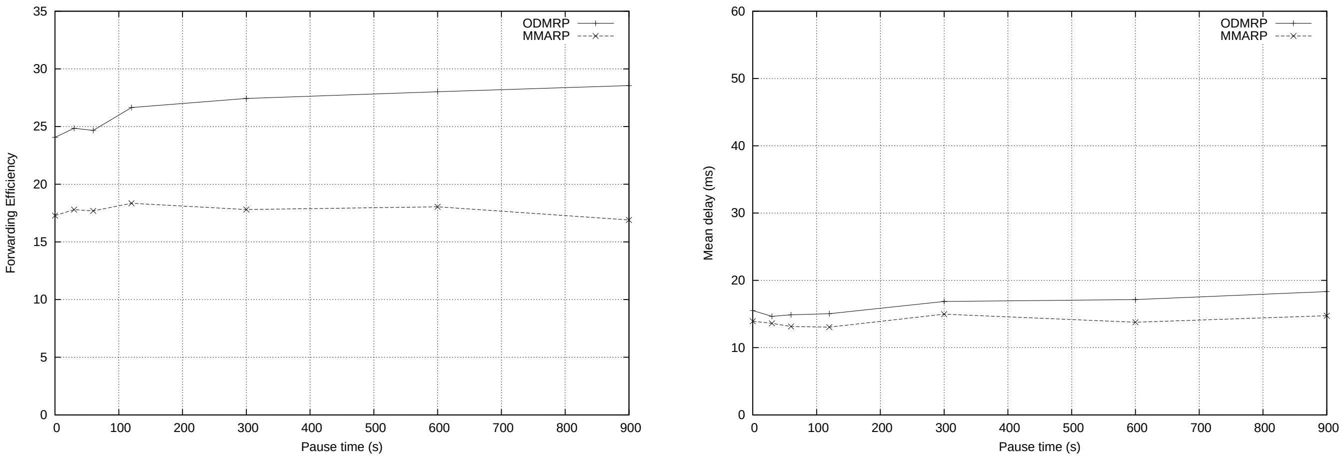
To assess the effectiveness of the different protocols, we have used the following performance metrics:

- Packet delivery ratio. Defined as the number of data packet successfully delivered over the number of data packets generated by the sources.
- Normalized packet overhead. Defined as the total number of control and data packets sent and forwarded normalized by the total number of packets successfully delivered across all the multicast receivers.
- Forwarding Efficiency. The mean number of times that a multicast data packet was forwarded by the routing protocol. This metric represents the efficiency of the underlying forwarding structure.



(a) Packet delivery ratio

(b) Normalized packet overhead



(c) Forwarding efficiency

(d) Latency

Fig. 3. Performance results for 1 source and 15 receivers

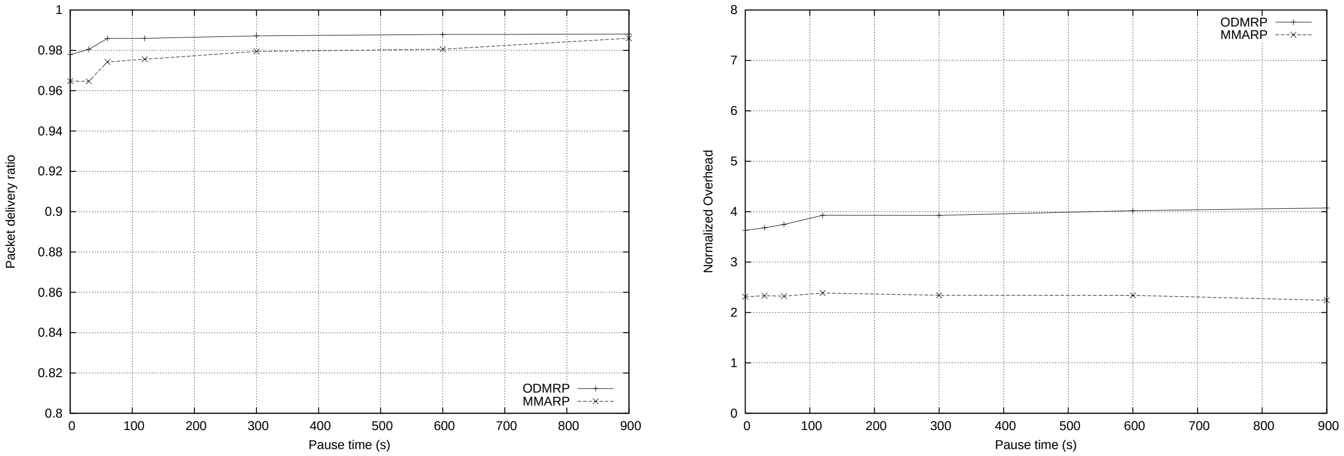
- Mean delivery latency. The mean difference between the time at which a data packet is generated and the time at which it is received at the destination. The mean latency is calculated independently for each receiver and then they are averaged across all receivers.

C. ODMRP Simulation parameters

For the simulations we used the default values for ODMRP. The REFRESH_INTERVAL was fixed at 3 seconds and the FG_FLAG timeout was fixed at 3 times the REFRESH_INTERVAL. The maximum number of JOIN_REPLY retransmissions was fixed at 3 and the time which a node waits before sending a JOIN_REPLY (in case it can aggregate several of them in a single message) was 0.025 seconds. In addition, for the modified variant of ODMRP we configured the the source aggregation timeout, being the number of seconds to wait for better routes before propagating the JOIN_QUERY, to be 0.015 seconds.

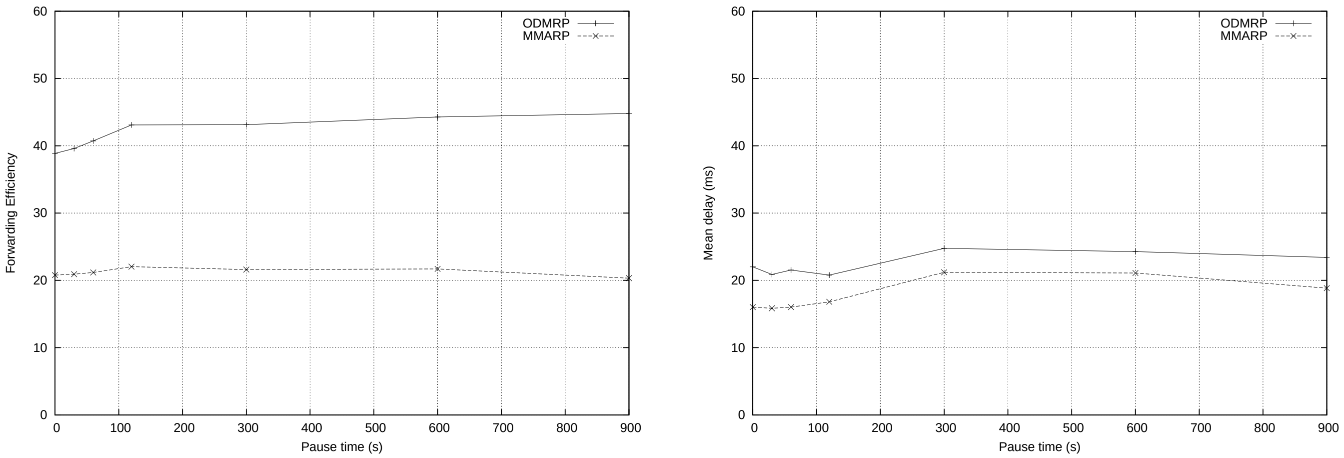
D. Performance evaluation

The packet delivery ratio as a function of the pause time is depicted in 3(a) for the two variants in the 1 source and 15 receivers scenario. Both variants deliver over 98% of the traffic even in the highly mobile scenarios. The original ODMRP version delivers around a 1% more of data packets than the proposed alternative. However, to achieve this packet delivery ratio, the original version requires around a 37% more of forwarding nodes (see figure 3(c)). In addition, the proposed alternative has a lower overhead. This is because by reducing the number of forwarding nodes, the data overhead is also reduced. An additional benefit of reducing the data overhead is the reduction of the mean latency between the source and the receivers. The expected behavior is that ODMRP would have



(a) Packet delivery ratio

(b) Normalized packet overhead



(c) Forwarding efficiency

(d) Latency

Fig. 4. Performance results for 2 sources and 15 receivers

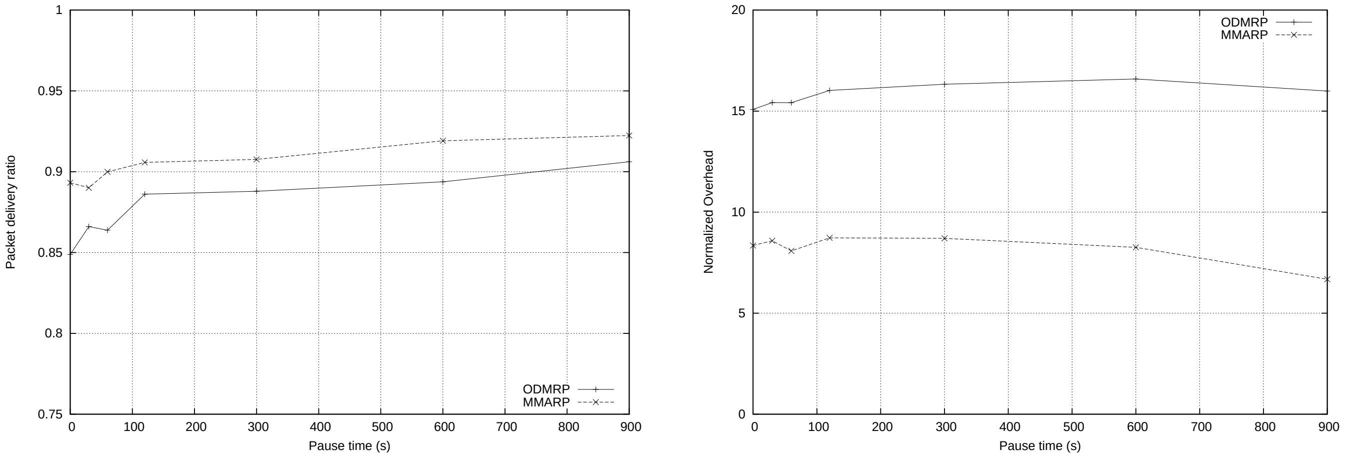
offered a lower latency because it uses the shortest path tree. However, the proposed approach by reducing the number of forwarding nodes also reduces the MAC-layer contention and collisions among forwarding nodes. Thus, the overall average latency can be reduced.

The cause for the important difference in the number of forwarding nodes between the two variants is that in the original ODMRP the randomness in the access to the MAC layer can make the shortest path routes to change very quickly. Thus, after a new route has been selected, the forwarding nodes in the old path will still remain active for two additional refresh intervals. Therefore a higher reliability is obtained at the cost of augmenting very much the number of forwarding nodes. In the proposed approach, although the shortest path tree may change quickly, the least cost path does not change so frequently. Thus, the number of forwarding nodes is not increased that much.

The performance results are very similar in the scenarios for 1 source and 5 and 30 receivers. In general, the higher the number of receivers the lower the differences in packet delivery ratio and also the lower the differences in forwarding efficiency. This is because as the number of receivers increase, so does it the number of forwarding nodes which are really needed. Thus, the additional number of forwarding nodes used by ODMRP in a change of the shortest path tree is reduced because most of the new nodes are already forwarding nodes. In addition, even in the case of 5 receivers the difference in the packet delivery ratio between both approaches never went beyond 2%.

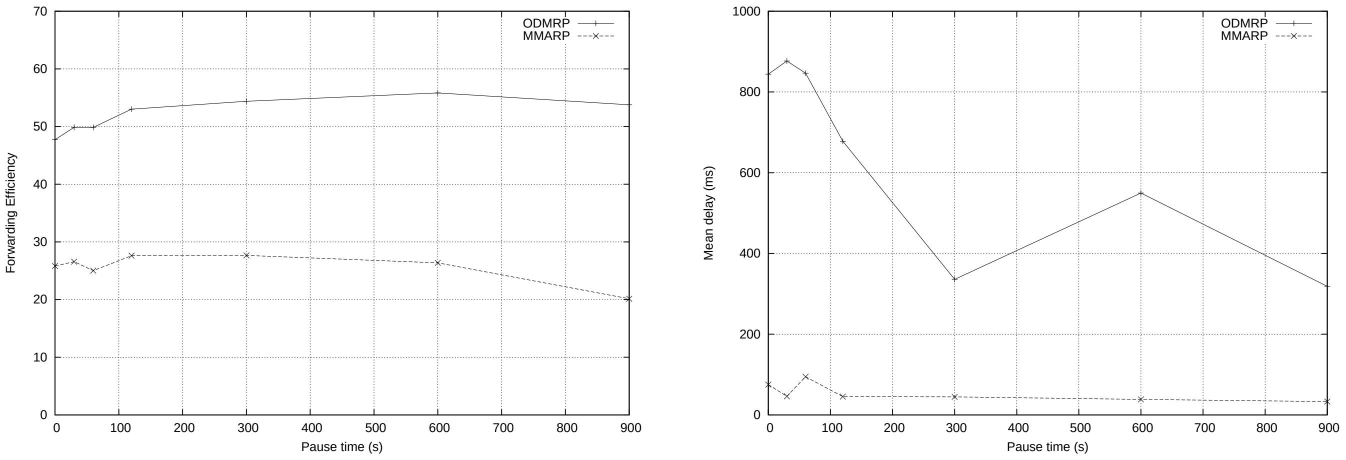
The evaluation of the scenarios with 2 sources and 15 receivers in Fig. 4 shows a similar trend.

The packet delivery ratio of the proposed approach is still



(a) Packet delivery ratio

(b) Normalized packet overhead



(c) Forwarding efficiency

(d) Latency

Fig. 5. Performance results for 5 sources and 5 receivers

around 1.4% lower for the highly mobile scenarios and around 0.8% lower for the medium mobility scenarios. However, the proposed approach becomes even more efficient compared to the original ODMRP. In Fig. 4(c) it is showed that the proposed approach manages to improve the forwarding efficiency by 50%. Given that in these scenarios the traffic load is doubled, both approaches experiment a slightly increase in the average latency. This is due to the higher contention at the MAC-layer. However, as Fig. 4(d) depicts, the proposed approach improves even more its average latency compared to the one of the original ODMRP. This is explained by the fact that in this scenarios the proposed approach has around a 50% less forwarding nodes. Similar results were obtained for the case of 5 and 30 receivers.

In the scenarios with 5 multicast sources, we can see that the traffic load is so high, the ad hoc network starts getting congested. This can be shown by the reduction in the packet delivery ratio shown in Fig. 5(a). The congestion for ODMRP is so high for the case of 15 and 30 receivers that we focus here in the case of 5 receivers.

As it is shown, the packet delivery ratio obtained by the proposed approach is around a 3.5% better than the original ODMRP. The difference in packet delivery ratio is because the original ODMRP, by using around half of the network as forwarding nodes (see Fig. 5(c)) creates a big amount of contention and collisions while propagating data messages. The proposed approach by using around a 50% less of forwarding nodes, manages to become not so much affected by the congestion. In addition, this is also supported by the huge differences in the latency which Fig. 5(d) shows. In the case of 15 and 30 receivers the differences are even higher in favor of the proposed approach due to a big congestion in the case of

ODMRP. So, we conclude that by reducing data overhead the proposed approach manages to offer similar performance than ODMRP as a lower cost in terms of forwarding efficiency. This lower cost allows the proposed approach to support a higher overall traffic load.

VI. CONCLUSIONS

We have introduced an heuristic algorithm to reduce the data overhead of mesh-based multicast ad hoc routing protocols. The algorithm adapts to the number of sources in the network to further reduce the number of forwarding nodes when there is enough reliability in the forwarding mesh. In addition, we justify the use of this heuristic algorithm by showing the NP-completeness of the problem and the effectivity of reducing the number of forwarding nodes to reduce data overhead.

We have implemented a modified version of ODMRP based on the proposed scheme. The performance evaluation shows that the proposed scheme can achieve similar packet delivery ratios than the original ODMRP with a reduction of around 40-50% in the number of forwarding nodes and an enhancement in the average latency. The results in scenarios with a high traffic load show that the proposed scheme is able to achieve a higher overall network capacity.

ACKNOWLEDGMENT

Part of this work has been funded by Spanish MCYT by means of the "Ramon y Cajal" workprogramme, the ICSI Call for Spanish Technologists and the SAM (MCYT, TIC2002-04531-C04-03) project. This work has been performed during a Visiting Fellowship of Pedro M. Ruiz at the International Computer Science Institute (www.icsi.berkeley.edu).

REFERENCES

[1] S. Deering, "Multicast Routing in a Datagram Internetwork," *Ph.D. Thesis, Electrical Engineering Dept., Stanford University*, Dec. 1991.

[2] S.-E. Deering and D.-R. Cheriton, "Multicast Routing in datagram internetworks and extended LANs," *Transactions on Computer Systems*, vol.8, no.2, May 1990, pp. 85–110.

[3] J. Moy, "Multicast routing extensions for OSPF," *Computer communications of the ACM*, vol.37, no.8, August 1994, pp.61–66.

[4] T. Ballardie, P. Francis and J. Crowcroft, "Core Based Trees (CBT) – An architecture for scalable inter-domain multicast routing, " *Proc. of ACM SIGCOMM'93*, San Francisco, CA, October 1993, pp.85–95.

[5] S. Deering, D.-L. Estrin, D. Farinacci, V. Jacobson, C.-G. Liu and L. Wei, "The PIM architecture for wide-area multicast routing," *IEEE/ACM Transactions on Networking*, vol.4, no.2, April 1996, pp. 153–162.

[6] C. Cordeiro, H. Gossain and D. Agrawal, "Multicast over Wireless Mobile Ad Hoc Networks: Present and Future Directions" *IEEE Network*, no. 1, Jan 2003, pp. 52–59.

[7] C.-W. Wu, Y.-C. Tay and C.-K. Toh, "Ad Hoc Multicast Routing Protocol Utilizing Increasing id-numberS (AMRIS) Functional Specification," Internet-draft, work in progress, draft-ietf-manet-amris-spec-00.txt, November 1998.

[8] E.-M. Royer and C.-E. Perkins, "Multicast operation of the ad-hoc on-demand distance vector routing protocol," *Proceedings of ACM/IEEE MOBICOM'99*, Seattle, WA, August 1999, pp. 207–218.

[9] L. Ji and S. Corson, "A Lightweight Adaptive Multicast Algorithm," *Proceedings of IEEE GLOBECOM'98*, Sydney, Australia, November 1998, pp. 1036–1042.

[10] J.-G. Jetcheva, D.-B. Johnson, "Adaptive Demand-Driven Multicast Routing in Multi-Hop Wireless Ad Hoc Networks," *Proceedings of ACM MobiHoc'01*, Long Beach, CA, October, 2001, pp. 33–44.

[11] S.-J. Lee, W. Su, and M. Gerla, On-demand multicast routing protocol in multihop wireless mobile networks, *ACM/Kluwer Mobile Networks and Applications*, vol. 7, no. 6, pp. 441–452, December 2002.

[12] J.J. Garcia-Luna-Aceves and E.-L. Madruga, "The Core-Assisted Mesh Protocol," *IEEE Journal on Selected Areas in Communications*, vol.17, no.8, August 1999, pp. 1380–1394.

[13] C.-K. Toh, G. Guichala, S. Bunchua, "ABAM: On-Demand Associativity-Based Multicast Routing for Mobile Ad hoc Networks," *Proceedings of IEEE VTC-2000*, Boston, MA,September 2000, pp.987–993.

[14] E. Bommaiah, M. Liu, A. MacAuley and R. Talpade, "AMRoute: Ad hoc Multicast Routing Protocol," Internet-draft, work in progress, draft-talpade-manet-amroute-00.txt, August 1998.

[15] P. Sinha, R. Sivakumar and V. Bharghavan, "MCEDAR: Multicat Core-Extraction Distributed Ad hoc Routing," *Proc. of IEEE Wireless Commun. and Networking Conf.*, New Orleans, LA, September 1999, pp.1313–1319.

[16] L. Ji, and M.-S. Corson, "Diferentical Destination Multicast: A MANET Multicast Routing Protocol of Small Groups," *Proc. of IEEE INFOCOM'01*, Anchorage, Alaska, April 2001, pp. 1192-1202.

[17] J.-G. Jetcheva, Yih-Chun Hu, David A. Maltz and David B. Johnson, "A Simple Protocol for Multicast and Broadcast in Mobile Ad Hoc Networks," Internet-draft, work in progress, draft-ietf-manet-simple-mbcast-01.txt, July 2001.

[18] K..Bharath-Kumar and J.-M. Jaffe, "Routing to Multiple Destinations in Computer Networks," *IEEE Transactions on Communications*, no.31 Vol.3, 1983, pp. 343–351.

[19] R.-M. Karp, "Reducibility among combinatorial problems," *In Complexity of computer computations*, Plenum Press, New York, 1975, pp.85–103.

[20] L. Kou, G. Markowsky, and L. Berman, "A fast algorithm for Steiner trees," *Acta Informatica*, no. 15, vol. 2, 1981, pp. 141–145.

[21] J. Plesnik, "The complexity of designing a network with minimum diameter," *Networks*, no. 11, 1981, pp. 77–85.

[22] A. Zelikovsky, "An 11/6-approximation algorithm for the network Steiner problem," *Algorithmica*, no. 9, 1993, pp.463–470.

[23] K. Fall, K. Varadhan, "ns, Notes and Documentation", The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, November 2003.

[24] The Monarch Project, mobile networking architectures, Rice University. [On-line]. Available: http://www.monarch.cs.rice.edu/

[25] H. Lim and C. Kim, "Multicast Tree Construction and Flooding in Wireless Ad Hoc Networks," *Proceedings of the 3rd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, Boston, MA, USA. August, 2000, pp. 61–68.

[26] S. Lee, C. Kim, "Neighbor Supporting Ad hoc Multicast Routing Protocol," *Proceedings of the 1st ACM MobiHoc'*, Boston, MA, USA, 2000, pp. 37–44.