An Algorithm to Determine Multicast Meshes with Maximum Lifetime for Mobile Ad Hoc Networks

Natarajan Meghanathan and Srilakshmi R. Vavilala

Jackson State University
Jackson, MS 39217, USA
{nmeghanathan,srilakshmi.r.vavilala}@jsums.edu

Abstract. We propose an algorithm called *OptMeshTrans* to determine a sequence of stable meshes that connect the source nodes and receiver nodes of a multicast session in Mobile Ad hoc Networks (MANETs). *OptMeshTrans* uses the following greedy strategy: Whenever a mesh connecting a set of source nodes to a set of receiver nodes is required, we choose the mesh, called the Stable-Static-Mesh, which exists for the longest time. In this pursuit, we determine a long-living minimum edge Steiner tree connecting the source nodes to the receiver nodes and the Stable-Static-Mesh is an extension of this Steiner tree by including in the mesh, all the edges that exist between the constituent nodes of the tree. When such a Stable-Static-Mesh gets disconnected, leading to the absence of a path from any source to any receiver node, we use the above greedy principle to construct another long-living mesh. The sequence of long-living Stable-Static-Meshes determined over the duration of the multicast session time is called a Stable-Mobile-Mesh. The lifetime of the meshes determined using algorithm *OptMeshTrans* forms the benchmark for maximum mesh lifetime in multicast routing. Simulation results indicate that the lifetime of the meshes determined using the classical mesh-based On-Demand Multicast Routing Protocol (ODMRP) is significantly lower than the optimal lifetime of the stable meshes determined using *OptMeshTrans*.

Keywords: Mesh, Multicast Routing, Optimality, Stability, Simulation.

1 Introduction

A mobile ad hoc network (MANET) is an autonomous, dynamic distributed system of mobile nodes connected through wireless links. The wireless nodes have limited battery charge and operate with limited transmission range. Hence, routes in MANETs are mostly multi-hop in nature, with a node, apart from being a source or receiver, also routing packets for other nodes in the network. Multicasting has emerged as a desirable and essential technology for several distributed applications in wireless networks such as audio/ video conferencing, distance learning, collaborative and groupware applications and etc. The mobility of nodes, with the constraints of limited battery charge and bandwidth, makes multicast routing a very challenging problem in MANETs [1]. It is advantageous to use multicast rather than multiple unicast, especially in ad hoc environments, where bandwidth comes at a premium. MANET multicast routing protocols can be classified into two types based on the multicast topology [1]: *tree-based* and *mesh-based*. In tree-based protocols, there

N. Meghanathan et al. (Eds.): NeCoM, WiMoN, and WeST 2010, CCIS 90, pp. 250–259, 2010.

exists only a single path between a source-receiver pair, whereas in mesh-based protocols there are multiple paths between a source-receiver pair. The presence of multiple paths adds to the robustness of the mesh-based protocols at the cost of multicast efficiency. We focus on mesh-based multicast protocols in this paper.

In this paper, we propose an algorithm called *OptMeshTrans* to determine a sequence of long-living stable multicast meshes connecting a set of source nodes to a set of receiver nodes for MANETs. The *OptMeshTrans* algorithm uses the following greedy strategy: Whenever a mesh connecting a set of source nodes to a set of receiver nodes is required, we choose the mesh, called the Stable-Static-Mesh, which exists for the longest time. In this pursuit, we determine a long-living minimum edge Steiner tree connecting the source nodes to the receiver nodes and the Stable-Static-Mesh is an extension of this Steiner tree by including in the mesh, all the edges that exist between the constituent nodes of the tree. We use the Kou et al.'s heuristic [2] (refer section 2) to approximate a multicast Steiner tree connecting the set of source nodes to the set of receiver nodes using the minimum number of links. When such a Stable-Static-Mesh gets disconnected, leading to the absence of a path from any source to any receiver node, we use the above greedy principle to construct another long-living mesh. The sequence of long-living Stable-Static-Meshes determined over the sequence of the multicast session time is called a Stable-Mobile-Mesh.

The lifetime of the stable meshes determined using algorithm *OptMeshTrans* forms the benchmark for the optimum (maximum) mesh lifetime that is obtainable in a given network. We conduct extensive simulations of algorithm *OptMeshTrans* and the classical mesh-based On-Demand Multicast Routing Protocol (ODMRP) [3] and compare their performance with respect to mesh lifetime, number of edges per mesh and the hop count per source-receiver path. ODMRP basically determines minimum hop paths between every source-receiver pair and the congregate of such minimum hop paths by including the edges that exist between the forwarding nodes of the paths forms the multicast mesh. Simulation results indicate a tradeoff between {mesh lifetime and number of edges per mesh} vs. {hop count per source-receiver path}. The meshes determined using algorithm *OptMeshTrans* have longer lifetime and relatively fewer edges, but have a larger hop count per source-receiver path compared to the meshes discovered using ODMRP.

The rest of the paper is organized as follows: Section 2 describes algorithm *OptMeshTrans* in detail, analyzes its complexity and also provides proof of correctness. Section 3 presents and analyzes the simulation performance results comparing *OptMeshTrans* with ODMRP. Section 4 concludes the paper. Throughout the paper, the terms 'link' and 'edge', 'node' and 'vertex' are used interchangeably.

2 Algorithm to Determine Stable-Mobile-Mesh

We model an ad hoc network as a unit disk graph [4] $G = (V, E)$, wherein V is the set of vertices representing the wireless nodes and E is the set of undirected edges representing the wireless links. An edge exists between two vertices if the corresponding nodes are within the transmission range of each other. Let S be the set of source nodes and R be the set of receiver nodes. The multicast group is represented as set $SR = S \cup R$, i.e., the union of the set of source nodes and receiver nodes. Algorithm *OptMeshTrans* uses the notion of a mobile graph [5] to represent the sequence of network topology changes.

A *mobile graph* [5] is defined as the sequence $G_M = G_1G_2 \ldots G_T$ of static graphs representing network topology changes over a time scale T. In the simplest case, the mobile graph $G_M = G_1G_2 \ldots G_T$ can be extended by a new instantaneous graph G_{T+1} to a longer sequence $G_M = G_1G_2 \ldots G_T\, G_{T+1}$, where G_{T+1} captures a link change (either a link comes up or goes down). We sample the network topology periodically for every 0.25 seconds, which could be the instants of data packet origination at the source.

We use the Kou et al's [2] well-known $O(|V||SR|^2)$ heuristic ($|V|$ is the number of nodes in the network graph and $|SR|$ is the size of the multicast group comprising of the source nodes and the receiver nodes) to approximate the minimum edge Steiner tree in graphs representing snapshots of the network topology. An *(S-R)-Steiner-tree* is defined as the multicast Steiner tree connecting the set of source nodes, S, to the set of receiver nodes, R. We give a brief outline of the heuristic in Figure 1.

Input: An undirected graph $G = (V, E)$
Multicast group $SR \subseteq V$
Output: An *(S-R)-Steiner-tree* for the set SR in G

Step 1: Construct a complete undirected weighted graph $G_C = (SR, E_C)$ from G and SR where $\forall (v_i, v_j) \in E_C$, v_i and v_j are in SR, and the weight of edge (v_i, v_j) is the length of the shortest path from v_i to v_j in G.
Step 2: Find the minimum weight spanning tree T_C in G_C (If more than one minimal spanning tree exists, pick an arbitrary one).
Step 3: Construct the sub graph G_{SR} of G, by replacing each edge in T_C with the corresponding shortest path from G (If there is more than one shortest path between two given vertices, pick an arbitrary one).
Step 4: Find the minimal spanning tree T_{SR} in G_{SR} (If more than one minimal spanning tree exists, pick an arbitrary one). Note that each edge in G_{SR} has weight 1.
Step 5: Construct the *(S-R)-Steiner-tree*, from T_{SR} by deleting the edges in T_{SR}, if necessary, such that all the leaves in the *(S-R)-Steiner-tree* are members of SR.

Fig. 1. Kou et al's Heuristic [2] to find an Approx. Minimum Edge Steiner Tree

2.1 Description of Algorithm *OptMeshTrans*

We now describe the *OptMeshTrans* algorithm proposed to determine the sequence of multicast meshes connecting a set of sources (S) to a set of receivers (R), such that the meshes exist for the longest possible time and the number of mesh transitions is minimal. The pseudo code is given in Figure 2. Algorithm *OptMeshTrans* operates according to the following greedy strategy: Whenever a multicast mesh connecting all the source nodes (S) to all the receiver nodes (R) of a multicast group is required, the multicast mesh, called the Stable-Static-Mesh, represented as $(S\text{-}R)_{Stable\text{-}Static\text{-}Mesh}$, which exists for the longest time is selected.

A mobile graph $G_M=G_1G_2.....G_T$ is generated by sampling the network topology at regular time intervals $t_1, t_2 \ldots t_T$. At time instant t_i, when a multicast mesh is required, a mobile sub graph $G(i, j) = G_i \cap G_{i+1} \cap \ldots \cap G_j$ is constructed such that there exists at least one mesh connecting every source $s \in S$ to every receiver $r \in R$ in $G(i, j)$ and no mesh exists in $G(i, j+1)$. A minimum edge *(S-R)-Steiner-tree* connecting every source s ($\in S$) to every receiver r ($\in R$) is constructed based on the Kou's heuristic and the Steiner tree is extended to a mesh by including all the edges (represented by the set *Additional-Edges* in the pseudo code, Figure 2) that exist between the constituent nodes of the tree in the mobile sub graph $G(i, j)$. The above procedure is repeated until time instant $j+1 \leq T$, where T is the duration of the multicast session. The Stable-Mobile-Mesh, represented as $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh}$, is a sequence of such maximum lifetime Stable-Static-Meshes and will undergo the minimum number of mesh transitions (i.e., mesh changes).

If T is the duration of the multicast session and k is the sampling rate (k samples of static graphs collected per unit time) used to form the mobile graph, the Kou et al.'s heuristic has to be run $T*k$ times, each time on a graph of $|V|$ nodes. During each such iteration, we will also have to form the set of *Additional-Edges* to extend the minimum edge Steiner tree to a multicast mesh. At the worst case, there would be $O(|V|)$ vertices in the minimum edge Steiner tree and it would take $O(|V|^2)$ time to determine whether an edge between every pair of vertices in the Steiner tree exists in the mobile sub graph for inclusion in the set of *Additional-Edges*. Hence, the run-time complexity of *OptMeshTrans* would be $O((|V||SR|^2 + |V|^2)\ T*k)$, where SR is the union of the set of sources and receivers of the multicast group.

Input: $G_M = G_1 G_2 \ldots G_T$, Set of source nodes - S, Set of receiver nodes - R
Output: $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh}$ // Stable-Mobile-Mesh
Auxiliary Variables: i, j, *Additional-Edges*
Initialization: i=1; j=1; $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh} = \phi$, *Additional-Edges* = ϕ
Begin *OptMeshTrans*

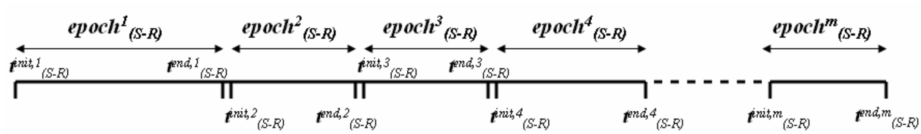
1. **while** $(i <= T)$ **do**
2. Find a mobile sub graph $G(i, j) = G_i \cap G_{i+1} \cap \ldots \cap G_j$ such that there exists at least one *(S-R)-Steiner-tree* connecting every source $s \in S$ to every receiver $r \in R$ in $G(i, j)$ and {no such *(S-R)-Steiner-tree* exists in $G(i, j+1)$ or $j = T$}
3. **if** $\exists$ a *(S-R)-Steiner-tree* in $G(i, j)$ **then**
4. **for** (every vertex u and v in *(S-R)-Steiner-tree*)
5. **if** (edge $(u, v) \in G(i, j)$ and edge $(u, v) \notin$ *(S-R)-Steiner-tree*) **then**
6. *Additional-Edges* = *Additional-Edges* U {(u, v)}
7. **end if**
8. **end for**
9. $(S\text{-}R)_{Stable\text{-}Static\text{-}Mesh}$ in $G(i, j)$={*(S-R)-Steiner-tree* in $G(i, j)$} **U** *Additional-Edges*
10. $i = j + 1$
11. *Additional-Edges* = ϕ
12. **end if**
13. **end while**
14. **return** $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh}$

End *OptMeshTrans*

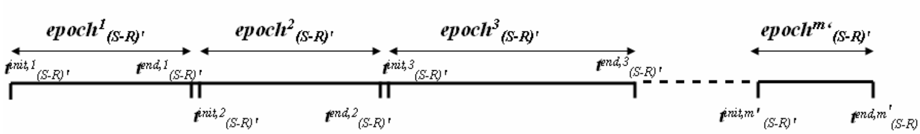
Fig. 2. Pseudo code for *OptMeshTrans* algorithm

2.2 Proof of Correctness of Algorithm *OptMeshTrans*

Given a mobile graph $G_M = G_1 G_2 \ldots G_T$, set of sources S and the set of receivers R, let the number of mesh transitions generated by *OptMeshTrans* in the Stable-Mobile-Mesh, $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh}$ be m. We use the proof by contradiction technique to prove the correctness of the *OptMeshTrans* algorithm. To show that m is optimal, we assume the contrary as the hypothesis for our proof, i.e., there exists another Stable-Mobile-Mesh $(S\text{-}R)'_{Stable\text{-}Mobile\text{-}Mesh}$ with m' number of mesh transitions, such that $m' < m$.



(a) Sampling Time Instants for $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh}$ (Algorithm *OptMeshTrans*)



(b) Sampling Time Instants for $(S\text{-}R)'_{Stable\text{-}Mobile\text{-}Mesh}$ (Hypothesis for the Proof)

Fig. 3. Sampling Time Instants to Prove the Correctness of Algorithm *OptMeshTrans*

Let $epoch^1_{(S\text{-}R)}$, $epoch^2_{(S\text{-}R)}, \ldots\ldots, epoch^m_{(S\text{-}R)}$ (refer Figure 3a) and $epoch^1_{(S\text{-}R)'}$, $epoch^2_{(S\text{-}R)'}, \ldots\ldots, epoch^{m'}_{(S\text{-}R)'}$ (refer Figure 3b) be the set of sampling time instants in $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh}$ *and* $(S\text{-}R)'_{Stable\text{-}Mobile\text{-}Mesh}$ respectively, wherein no mesh transitions exist. Let $t^{init,j}_{(S\text{-}R)}$, $t^{init,k}_{(S\text{-}R)'}$ be the initial and $t^{end,j}_{(S\text{-}R)}$, $t^{end,k}_{(S\text{-}R)'}$ be the final sampling time instants of $epoch^j_{(S\text{-}R)}$ where $1 \leq j \leq m$ and $epoch^k_{(S\text{-}R)'}$ where $1 \leq k \leq m'$ respectively. Since $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh}$ and $(S\text{-}R)'_{Stable\text{-}Mobile\text{-}Mesh}$ exist over the same time period T, the initial and final sampling time instants are same (i.e., $t^{init,1}_{(S\text{-}R)} = t^{init,1}_{(S\text{-}R)'}$ *and* $t^{end,m}_{(S\text{-}R)} = t^{end,m'}_{(S\text{-}R)'}$). As our hypothesis is that the number of transitions in $(S\text{-}R)'_{Stable\text{-}Mobile\text{-}Mesh}$ is less than that of $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh}$, there should exist a mesh in $(S\text{-}R)'_{Stable\text{-}Mobile\text{-}Mesh}$ that has longer lifetime than that in $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh}$, $m' < m \Rightarrow \exists\ j, k$ where $1 \leq j \leq m$ and $1 \leq k \leq m'$ such that $epoch^j_{(S\text{-}R)} \subset epoch^k_{(S\text{-}R)'}$, i.e., $t^{init,k}_{(S\text{-}R)'} < t^{init,j}_{(S\text{-}R)} < t^{end,j}_{(S\text{-}R)} < t^{end,k}_{(S\text{-}R)'}$). In other words, there should exist a $(S\text{-}R)'_{Stable\text{-}Static\text{-}Mesh}$ in $[t^{init,k}_{(S\text{-}R)'}, \ldots, t^{end,k}_{(S\text{-}R)'}]$. But, in *OptMeshTrans* algorithm, a transition was made at $t^{end,j}_{(S\text{-}R)}$ as the mesh that started to exist at $t^{init,j}_{(S\text{-}R)}$ does not exist beyond $t^{end,j}_{(S\text{-}R)}$. So, $t^{end,k}_{(S\text{-}R)'}$ should be less than or equal to $t^{end,j}_{(S\text{-}R)}$ and cannot be greater. There is no common $(S\text{-}R)'_{Stable\text{-}Static\text{-}Mesh}$ in $[t^{init,j}_{(S\text{-}R)'}, \ldots, t^{end,k}_{(S\text{-}R)'}]$ and hence there is no common $(S\text{-}R)'_{Stable\text{-}Static\text{-}Mesh}$ in $[t^{init,k}_{(S\text{-}R)'}, \ldots, t^{end,k}_{(S\text{-}R)'}]$. Therefore, the lifetime of all the meshes in $(S\text{-}R)'_{Stable\text{-}Mobile\text{-}Mesh}$ has to be less than or equal to that of $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh}$ i.e., $m' \geq m$. This is in contradiction to the hypothesis. Hence, m, the number of transitions in *OptMeshTrans* algorithm is optimal (minimum) and $(S\text{-}R)_{Stable\text{-}Mobile\text{-}Mesh}$ is the Stable-Mobile-Mesh connecting the set of sources *S* to the set of receivers *R*.

3 Simulations

We implemented ODMRP and *OptMeshTrans* in a discrete event simulator developed by us in Java. The network dimensions are 1000m x 1000m. The transmission range of each node is 250m. We vary the density of the network by conducting simulations with 50 nodes (low density) and 100 nodes (high density). The simulation time is 1000 seconds. The IEEE 802.11 Medium Access Control (MAC) protocol [6] has been used as the link-layer protocol for ODMRP. The mobility model used is the Random Waypoint model [7], wherein the velocity of a node is uniform-randomly selected from $[0,\ldots,v_{max}]$ every time the node incurs a direction change to travel to a randomly selected location within the network. The v_{max} values used are 5 m/s and 50 m/s, characteristic of low and high node mobility respectively. For each v_{max} value, we generated five mobility profiles of the nodes for the simulation time of 1000 seconds.

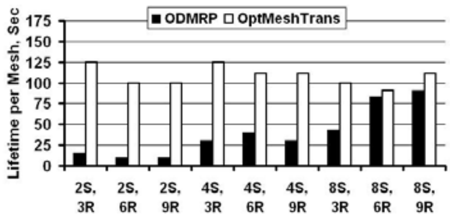
The values for the number of sources used are: 2, 4 and 8; and the values for the number of receivers used are: 3, 6 and 9. The data packet size is 512 bytes and the packet sending rate from each source to the set of receivers is 4 packets per second. For each value of the number of sources and receivers, we created one list of source nodes and five lists of receiver nodes. All the node lists are generated randomly; but, we made sure a node acts at most only as a source or a receiver; not both. Simulations for a given number of sources were run for each of these five lists using the five mobility profiles generated for each v_{max} value. Each data point obtained for ODMRP [3] and *OptMeshTrans* in the performance figures 4, 5 and 6 is the average value obtained from these 25 experiments for a given number of sources and v_{max} value.

The following performance metrics are measured for the ODMRP protocol and the *OptMeshTrans* algorithm under the different simulation conditions described above.

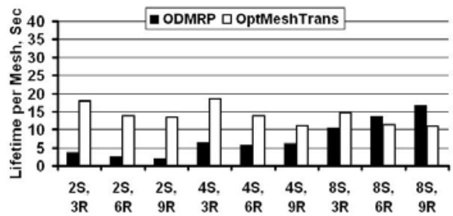
(i) Lifetime per Mesh – average of the lifetimes of the sequence of multicast meshes discovered over the duration of the entire multicast session.
(ii) Edges per Mesh – time-averaged value of the number of edges per mesh connecting the set of sources to the set of receivers, for the multicast session.
(iii) Hop Count per Source-Receiver Path – time-averaged hop count of the paths from the source to each receiver, considering all the source-receiver pairs and computed over the entire multicast session.

3.1 Average Mesh Lifetime

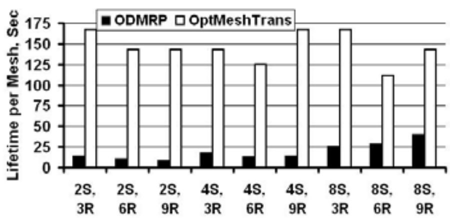
It is imperative to form multicast meshes with larger lifetime because each time a new mesh is to be formed, a global network-wide broadcast of the control messages from each source node is initiated. The larger the value for the mesh lifetime, the lower will be the number of times such resource-consuming global broadcast of control messages will be needed in the network. The Stable-Mobile-Meshes are relatively more stable (have larger lifetime) than compared to ODMRP meshes. The meshes formed using algorithm *OptMeshTrans* have 400%-450% (on average) and 830%-1450% (at the worst case) longer lifetime than the meshes formed using ODMRP.



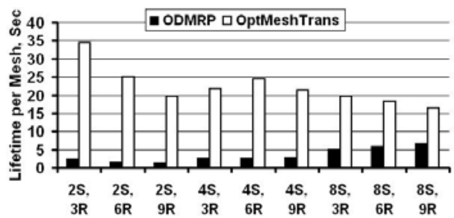
(a) 50 Nodes, v_{max} = 5 m/s



(b) 50 Nodes, v_{max} = 50 m/s



(c) 100 Nodes, v_{max} = 5 m/s



(d) 100 Nodes, v_{max} = 50 m/s

Fig. 4. Average Lifetime per Mesh

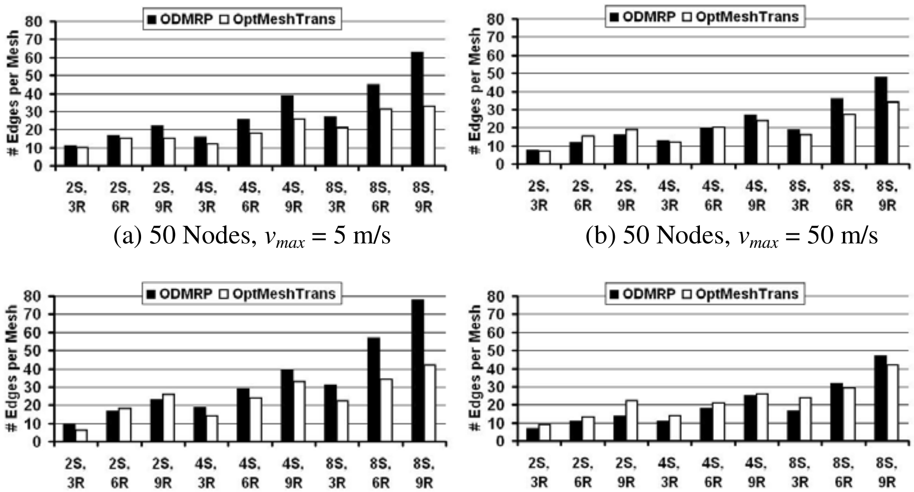
For a given node mobility and network density, the lifetime of meshes formed using ODMRP increases with increase in the number of sources. This is due to the increase in the number of edges to ensure connectivity in the mesh with increase in the number of sources. With an increased number of edges, there is an increase in the number of alternate paths between a source-receiver pair in a mesh, resulting in larger lifetime between two successive mesh transitions. The lifetime of meshes formed using *OptMeshTrans* does not relatively change much with increase in the number of sources. This is because the algorithm looks into the future topology changes and considers a mobile sub graph that consists of the minimum number of edges that will exist for a longer time as well as constitute a mesh.

For fixed node mobility and number of sources, with increase in node density, the average lifetime per mesh discovered using ODMRP decreases. This can be attributed to the decrease in the hop count per source-receiver path with increase in node density, leading to an increase in the probability of a path break in the near future. As node density increases, the Stable-Mobile-Mesh comprises of relatively better stable paths in which the physical Euclidean distance between the end nodes of the constituent links is close to only 50-60% of the transmission range of the nodes.

For fixed node density and number of sources, with increase in node mobility, the neighbors of each node move very fast, leading to a larger probability of link break in the near future. Hence, the lifetime per mesh for both ODMRP and *OptMeshTrans* would naturally be lower with increase in node mobility. For a fixed node density, mobility and number of sources, the lifetime per mesh is more likely to decrease with increase in the number of receivers as it becomes difficult to maintain the connectivity of a mesh involving more receiver nodes, but a fixed number of source nodes.

3.2 Average Number of Edges Per Mesh

The meshes formed using the *OptMeshTrans* algorithm have relatively few edges compared to those discovered using ODMRP. This is due to the decrease in the number of edges in the mobile sub graph which is an intersection of the static graphs of the network in the future. The ODMRP protocol focuses on discovering minimum hop paths between a source-receiver pair available at the current instant and the congregation of such locally optimal paths forms the mesh. There is no inclination to reduce the number of edges in the mesh when the individual paths are discovered in the case of ODMRP. On the other hand, *OptMeshTrans* looks at the future and is based on the minimum edge Steiner tree heuristic. Hence, it focuses on discovering a mesh that will exist for a longer time with a reduced number of constituent links.



(a) 50 Nodes, v_{max} = 5 m/s (b) 50 Nodes, v_{max} = 50 m/s

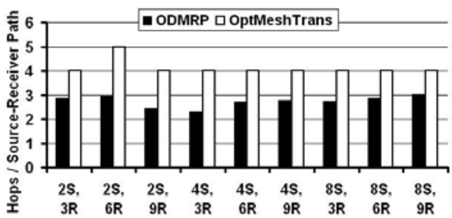
(c) 100 Nodes, v_{max} = 5 m/s (d) 100 Nodes, v_{max} = 50 m/s

Fig. 5. Average Number of Edges per Mesh

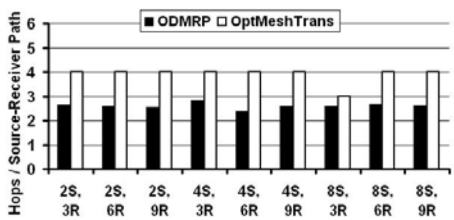
As node mobility increases, the number of edges per mesh decreases for both ODMRP and *OptMeshTrans*. For fixed node mobility, the number of edges also increases for both of them, with increase in node density. For different node mobility and density scenarios, the number of edges per mesh increases with increase in the number of sources. We can also observe an increase in the number of edges with increase in the number of receivers with different node density and node mobility values. As the number of sources and receivers increases, the number of edges per mesh increases to maintain connectivity in the mesh. However, for all the above scenarios, the rate of increase in the number of edges per mesh discovered using the *OptMeshTrans* algorithm is lower than that observed with ODMRP. The meshes formed using algorithm *OptMeshTrans* have 13%-20% (on average) and 45%-48% (at the worst case) fewer edges than that of the meshes formed using ODMRP.

3.3 Average Hop Count Per Source-Receiver Path

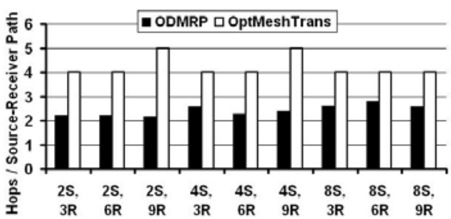
The average hop count per source-receiver path is a measure of the end-to-end delay per data packet. The source-receiver paths that are part of the meshes discovered using algorithm *OptMeshTrans* have a larger hop count compared to those discovered using ODMRP. This can be attributed to the relatively fewer number of edges in the meshes discovered using algorithm *OptMeshTrans*. With fewer edges, some of the paths between a particular source node and receiver node in the mesh could be relatively longer (i.e. more hops). ODMRP looks at the current network topology and determines minimum hop paths between individual source and receiver nodes.



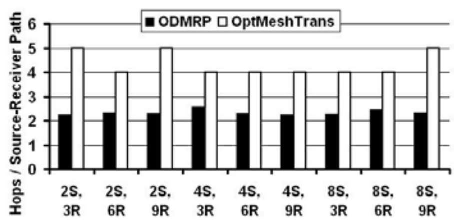
(a) 50 Nodes, v_{max} = 5 m/s



(b) 50 Nodes, v_{max} = 50 m/s



(c) 100 Nodes, v_{max} = 5 m/s



(d) 100 Nodes, v_{max} = 50 m/s

Fig. 6. Average Hop Count per Source-Receiver Path

For fixed node mobility, the average hop count per source-receiver path in the meshes discovered using ODMRP increases with increase in the number of sources and receivers, but the increase is below 25%. The average hop count per source-receiver path in meshes discovered using ODMRP decreases with increase in node density, but the decrease is below 20%. For different node mobility and node density, the average hop count per source-receiver path in the meshes discovered using *OptMeshTrans* is not affected much by the number of sources and receivers. The difference in the hop count per source-receiver path in the meshes formed using *OptMeshTrans* and those formed using ODMRP increases with increase in density.

4 Conclusions and Future Work

The high-level contribution of this paper is the development of a theoretically optimal algorithm *OptMeshTrans* that forms a sequence of stable meshes. The complexity of *OptMeshTrans* would be $O((|V||SR|^2 + |V|^2)\, T{*}k)$, where *SR* is the union of the set of sources and receivers of the multicast group, *T* is the duration of the multicast session and *k* is the sampling rate, number of static graphs collected per unit time. The lifetime of the meshes determined using algorithm *OptMeshTrans* forms the benchmark for maximum possible mesh lifetime in multicast routing.

Simulation results indicate that the meshes formed using algorithm *OptMeshTrans* have significantly longer lifetime than those formed using ODMRP. Hence, there is still lot of scope to improve the stability of the meshes discovered by the multicast routing protocols. With respect to the number of edges per mesh (a measure of bandwidth efficiency and also energy-efficiency), the meshes formed using algorithm *OptMeshTrans* have fewer edges than those formed using ODMRP. However, the hop count per source-receiver path in the Stable-Mobile-Meshes is significantly larger compared to those discovered using ODMRP. This indicates the tradeoff between ODMRP and *OptMeshTrans* and our future research will be on developing a distributed version of *OptMeshTrans* that can minimize this {mesh lifetime and number of edges per mesh} vs. {hop count} tradeoff.

References

1. Murthy, C.S.R., Manoj, B.S.: Ad Hoc Wireless Networks: Architectures and Protocols. Prentice Hall, Upper Saddle River (2004)
2. Kou, L., Markowsky, G., Berman, L.: A Fast Algorithm for Steiner Trees. Acta Informatica 15, 141–145 (1981)
3. Lee, S.-J., Gerla, M., Chiang, C.-C.: On-Demand Multicast Routing Protocol. In: Wireless Communications and Networking Conference, pp. 1298–1302. IEEE, New Orleans (1999)
4. Kuhn, F., Moscibroda, T., Wattenhofer, R.: Unit Disk Graph Approximation. In: Joint Workshop on Foundations of Mobile Computing, pp. 17–23. ACM, Philadelphia (2004)
5. Farago, A., Syrotiuk, V.R.: MERIT: A Scalable Approach for Protocol Assessment. Mobile Networks and Applications 8(5), 567–577 (2003)
6. Bianchi, G.: Performance Analysis of the IEEE 802.11 Distributed Coordination Function. IEEE Journal of Selected Areas in Communication 18(3), 535–547 (2000)
7. Bettstetter, C., Hartenstein, H., Perez-Costa, X.: Stochastic Properties of the Random-Way Point Mobility Model. Wireless Networks 10(5), 555–567 (2004)