An efficient and message-optimal multicast routing protocol in mobile ad-hoc networks

Hai Trung Nguyen
University of Engineering and Technology
Vietnam National University
Hanoi, Vietnam
nguyen.hai@vnu.edu.vn

Tho Dai Nguyen
University of Engineering and Technology
Vietnam National University
Hanoi, Vietnam
nguyendaitho@vnu.edu.vn

*Abstract***— This paper presents a novel protocol for multicasting in mobile ad-hoc networks named Spanning Tree for Multicasting (STM), which establishes and maintains the spanning tree for each multicast group using optimal control packets, without requiring any underlying unicast routing protocol or any pre-configured state of group. Based on the original idea of optimal maintenance of a spanning tree (OMST), we improve this algorithm considerably and propose our multicast protocol, as well as implement it as an extension of simulator NS-2. We compare STM with MAODV and PUMA using NS-2 through various scenarios, the results show that STM attains top packet delivery ratio, while keeping less overhead than others. As far as we know, our implementation is the very first implementation inspired by OMST in practice.**

Keywords: Ad hoc netwok, routing, multicasting, multicast tree, spanning tree.

I. INTRODUCTION

Multicast communication is now widely used in distributed applications for efficiently delivering data from one source to all members of multicast group. In ad-hoc networks, communication between nodes has been established using multiple hops via relay nodes. Especially in mobile ad-hoc network (MANET), there are some special characteristics that may impact on the performance of communication [7]. In mobile environment, nodes move freely and unpredictably, so network topology changes frequently. Mobile nodes also have very limited battery power and network bandwidth. As a result, it is important to have a multicast protocol that can achieve high packet delivery ratio with overhead as low as possible.

Recent multicast protocols have been proposed for mobile ad-hoc networks [2][3][4][5][7][8] which shown the significant improvement to achieve a better packet delivery ratio, as well as less overhead. In the scope of our discussion, we classify modern approaches to design multicast protocol taken to date into tree-based and mesh-based. A tree-based multicast protocol establishes and maintains a shared tree between all group members, in which there is only one route between each pair of nodes. One state of the art protocol which is representative for tree-based solution is MAODV [2]. In contrast, a mesh-based multicast protocol maintains a mesh in which there is maybe more than one route between two remote nodes. Recent example of mesh-based protocol is PUMA, which outperformed than the former well-known protocol ODMRP [4], as shown in [3]. PUMA also has tree-based version named ROMANT which has slightly less performance than PUMA, according to [3][16].

We focus on tree-based solution that can avoid packet redundant, especially related to minimum spanning tree which has proven as the most efficient way to deliver data from one to all destinations of the network or graph. In a dynamic network like MANET, previous approaches that adapt for each topological change involve $\Omega(E)$ messages [10], where E is the number of network edges. This is likely the cost to reconstruct the tree from scratch. In most cases, E equivalents to $O(V^2)$, where V is the numbers of nodes, so that it cannot be applied to design the multicast protocol for MANET in practice. Recent result gave a better solution, optimal maintenance of a spanning tree (OMST) [1] that reduced the amortized message complexity to $O(V)$.

This paper presents a novel protocol for multicast routing in MANET with enhanced algorithm built on top of OMST. Our main contributions in this work are improving this algorithm for easier to use and designing a protocol with improvements, also making an implementation of this protocol as an extension of simulator NS-2 for evaluation. Section II describes more detail about the improved algorithm and protocol design. Section III evaluates this protocol with others to assess the correctness of algorithm (authors of OMST only prove the correctness of algorithm theoretically without any practice assessment).

II. STM DESCRIPTION

A. Overview

STM provides IP multicast service for both static and dynamic networks, featuring mobile ad-hoc networks, which allows any source to send multicast packets to a given of group through wireless broadcasting. As a tree-based protocol, STM aims to provide high packet delivery ratio while keeping overhead as low as possible in mobile ad-hoc environment. To achieve this goal, STM applies the idea of optimal maintenance of a spanning tree (OMST) algorithm which economizes the message complexity to adapt each topological change in a dynamic environment like mobile ad-hoc network. STM protocol simplifies algorithm of OMST for easier to apply in practice without changing the correctness and message complexity.

Some requirements of STM protocol that we follow to design STM protocol: (i) use wireless broadcast mechanism for control packets instead of relying on unicast protocol to deliver them, hence it may save network bandwidth, (ii) apply retransmission mechanism to cope with collision problem in wireless network, like the repair mode presented in [11], (iii) not use unbounded counter, as recommended in [12], to avoid problem of recycling sequence numbers mentioned in [3][16].

As built on top of OMST algorithm, STM enhances OMST mainly in two main subroutines: UPDATE and FIND

- UPDATE: to make the tree replica identical with the real tree, invoked by root when topological change occurs. This is very important procedure, to help FIND routine reduce the number of messages to find the best outgoing edge to merge into current tree.

- FIND: Similar to GHS-83 algorithm [6] to choose the best outgoing edge to merge. In STM, distance between two remote nodes is selected as weight to find minimum outgoing edge.

Other supporting procedures are based on original OMST algorithm. We only focus on UPDATE and FIND subroutines with a simplified data structure of tree replica which we design to make algorithm easier to design and implement in practice.

B. Data structure of tree replica

In OMST, tree topology is managed in special structure called "*tree replica*" (another term called "*forest replica*" has the same structure); actually it is a list of mobile links (or edges). This structure requires a large memory to store than nodes structure, especially to store "*forest replica*" of each node and the mechanism to reconstruct the topology from broken link is complicated, hence it takes more time to complete. We present a new structure that has the same functionality of tree replica but it is easier to design and implement. Rather than managing the list of edges of current tree, we store a list of mobile nodes.

Compared with "*tree replica*" using by OMST, "*tree replica*" using by STM shows some clear advantages:

- The memory size to store the data structure is reduced about half, since a mobile link consists of two endpoints. As seen in Figure 1, node 6 has tree list consisting of 3 nodes {6,1,7} (three memory units, each unit for one node). If using data structure as mentioned in original OMST, "*tree replica*" of node 6 should contain {(6,1), (1,7)}, also "*forest replica*" consists of tree replica plus non-tree edges {(7,6), (1,2), (2, 6)} (total ten memory units).

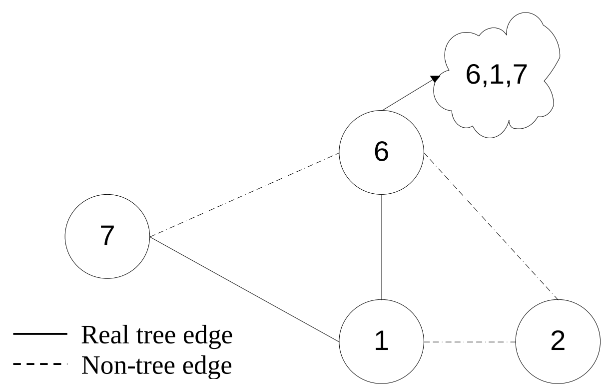
- The size of message DIFF send by subroutine UPDATE is smaller, as it doesn't send the list of mobile links (it sends list of mobile nodes instead).

- There is no need to manage forest replica of the whole tree at each node, as we describe is subroutine UPDATE and FIND below. As a result, the use of complicated "*side of tree*" is unnecessary anymore. The protocol will run faster and more effectively in highly dynamic network.

Figure 1 shows an example of data structure for topology management. Node 6 has three neighbors: 1, 2, 7, in which node 1 and 7 belongs to the same tree replica of node 6. This tree is built by two edges (7,1) and (1,6). Intuitively, edge (6,7) may create tree cycle, hence node 6 cannot choose node 7 as best outgoing node to merge trees, although both edges (7,6) and (6,2) can be candidates since they are not currently belonged to tree replica of node 6.

Each node also maintains its neighbor list by periodically asking underlying layer service (e.g. MAC layer) to provide the nodes list in its radio range. Based on neighbor list and "*tree replica*" list, multicast node only receives packets from neighbor node which has tree link with receiver. Otherwise, packets will be discarded.

The management of this node list can be simplified using binary coding: we map each node ID in node list into an index of a binary array. For example, node list {6, 1, 7} can be presented in binary array (low bit first) as 01000011. In our implementation, we use an array of 128 bits (16 bytes) to map node list, hence supporting group size up to 128 nodes.



Bit	0	1	2	3	4	5	6	7
	0	1	0	0	0	0	1	1

Figure 1: Node list management

C. Root election

At network starts up, each member of multicast group consider itself as root of a tree fragment containing single node. Based on "*tree replica*" list, root of each fragment updates the whole fragment topology by sending UPDATE messages through tree links to all fragment members. Each node chooses best outgoing edge (minimum weight) and sends this weight back to its parent until root. At that time, each node has a pointer to the next node (one of its children) on the route to the best outgoing edge. The root then transfers root privilege to the next node on that route using CHANGE ROOT message containing the node adjacent to the best outgoing edge. This message continues forwarding until it reaches the node which has the same identity with the one in CHANGE ROOT message. This new root will try to connect to the other endpoint of outgoing edge using the same mechanism as described in [1]. When two endpoints agree to connect, the one with higher identity will become the new root of combined fragment. This new root will continue this process until there is only one tree fragment left.

D. Tree formation and maintenance

Like MAODV and ROMANT, STM uses receiver-initiated approach to form the tree without pre-assignment of root (or core) to multicast group. Initially each multicast group member considers itself as a basic tree fragment in which this member elects itself as root. All trees in the whole network will try to connect with others using MERGE subroutine. This distributed implementation of MERGE subroutine is essentially similar with GHS-83 algorithm [1][6]. In STM, the root of tree sending MERGE message will be a new root of new tree combined by two tree fragments. Within a finite time proportional to the size of multicast group, all members of multicast group will join in single tree of the whole network. At the first time of tree formation, this tree is minimum spanning tree in which each node has the unique path to the root.

When a link in spanning tree is lost (due to node is moving out of range or turned off), this tree is separated into two sub tree fragments. The parent node of broken link will send ALERT message to its tree root through the spanning tree path. The child node of this link will become a new root of new tree fragment formed after link broken. Each tree fragment will try to find the outer fragment to connect, hence reconstruct the spanning tree. Before finding outgoing edge, each tree fragment updates the current topology to avoid tree cycle. After found the best outgoing edge to connect, this tree fragment will start the MERGE subroutine, finally the whole network will be reconstructed. This maintenance process will execute for each topological change. To reduce message complexity for adapting, tree fragment will execute UPDATE subroutine before finding best outgoing node to connect.

E. UPDATE subroutine

UPDATE subroutine is activated from root when topological change occurs, broadcasting UPDATE message. Each node receives UPDATE message through tree link will mark sender as parent and forward message to all neighbor members using wireless broadcast till leaf nodes. Leaf nodes start Echo process by sending UPDATED message to parent, this message contains the node list of all members presented in the current tree. This node list is initialized from leaf nodes, beginning with ID of leaf node itself, and then feeding back to its parent. Each node receives all UPDATED messages from its children will make an union of all node lists into one, append ID of itself into this list, and continue feeding back to its parent. This process will end at root, similarly to Broadcast and Echo mechanism [9] implemented in original OMST algorithm. Root node has the complete node list of all members which are presented in its current tree. This structure will be very useful later for FIND routine.

F. FIND subroutine

Root invokes FIND routine to broadcast complete node list of current tree to all members. Based on this structure, each node will choose the best outgoing node from its neighbor list which is not present in its "*tree replica*" list (to avoid tree cycle). The FIND routine is similar with the original OMST algorithm with above modification to make the correct choice of best outgoing node.

G. Data packet forwarding

When a node wants to send data packet to a multicast group, it firstly sets the address of this group in destination field of packet, and then broadcasts this message. A neighbor received this data packet will check if it has already received before. If yes, packet will be dropped. If no, receiver node will cache the packet number and forward this packet by broadcasting. This process continues until packet reaches multicast group member. From there, packet will be flooded through tree link similar to how UPDATE and FIND message are broadcast. Like ROMANT and PUMA, there is no need to encapsulate a data packet inside a unicast packet as the data packet forwards from sender to all multicast group members.

H. Message retransmission

In mobile environment, there is a lot of collision happen when packet is transmitted broadcasting over the air. Our deep analysis shows that it may causes packet dropped at receiver before it reaches to IP layer, hence the Echo process of Broadcast and Echo mechanism cannot be terminated at root. To overcome this obstacle, we define the retransmission mechanism to allow broadcast message like UPDATE, FIND or data packet can be re transmitted. When the parent node broadcasts a message, it expects all children neighbors to forward this message. Because all communication is broadcast, parent node will receive the original message from each neighbor. Parent node will consider this ping back message as implicit acknowledgement. When parent received enough implicit acknowledgements from all children, the original message is broadcast completely. Otherwise, after a timeout, parent node will retransmit current message. Each child node which already finish the Echo process of this message simply discard this message, otherwise it simply processes message like the first time receiving it.

I. Correctness

Theorem II.1: The tree built from nodes list maintains loop freedom invariant.

Proof: We proof that the each local tree in each execution loop doesn't contain any cycle. Actually, assume that mobile link *u*-*v* make the tree cycle, hence there is a preceded execution that merge two local trees through *u*-*v* link. This is impossible because the in this execution, *u* and *v* must belong to one local tree (otherwise it cannot make the tree cycle), it means *u* and *v* are presented concurrently in the tree list. Therefore, FIND subroutine cannot choose *u* (or *v*) to be the best outgoing node to merge.

Theorem II.2: As the loop freedom invariant holds, subroutine UDPATE and FIND always terminate, and the termination is detected by the real tree root.

This is directly resulted from idea of Broadcast and Echo mechanism, as described in [1].

J. Message complexity

Subroutines UPDATE and FIND only execute through spanning tree's edges using Broadcast and Echo mechanism, similarly with the original OMST, hence the amortized (on the number of topological changes) message complexity is O(V).

III. PERFORMANCE EVALUATION

To prove the correctness and message complexity of STM algorithm, we have implemented STM routing protocol as an extension of NS-2[13]. To our best knowledge, this is the first implementation of MANET routing protocol inspired by the original idea of OMST. We compared performance of STM against performance of PUMA and MAODV, which were representatives of modern approaches to multicast routing protocols for MANET. Like STM, PUMA uses broadcast mechanism to transmit control packets without using any underlying unicast routing protocol, but it is mesh-based protocol which provides multiple routes between senders and receivers, hence may provide better availability. MAODV and STM share the same tree-based protocol, but MAODV relies on AODV to send unicast packets while STM uses wireless broadcast only. On the other hand, STM, which is based on top of OMST, creates and maintains tree topology which is more likely minimum spanning tree than one created by MAODV.

We compared STM, PUMA and MAODV under NS-2. PUMA source code for NS-2 was provided by [14] and MAODV source code was obtained from [15]. We did not compare with ROMANT, although ROMANT was also tree based protocol, because it did not achieve as good performance as PUMA [3] [16].

Various experiment scenarios were carried out to figure out the effect of group mobility and group member on performance of each protocol. At first experiment, mobility was set across {2, 5, 10, and 15} m/s, group size is 20; traffic load was 10 packets/s. At second one, group size was set across {10, 20, 30 and 40}, mobility was 2 m/s, traffic load was 10 packets/s. Other environment configurations using for simulation are listed in Table 1.

Simulator	NS-2
Simulation area	1000m x 1000m
Radio range	250m
Simulation time	700s
Sender source	CBR
Node placement	Random
Mobility model	Random Waypoint Model
Pause time	0
Packet size	512 Bytes

Table 1: Simulation environment

The metrics used for our simulation were packet delivery ratio and control overhead as defined in Table 2. We focused on control overhead efficient factor (we now call "control overhead" for short) as it concerned to the ratio of useless packets sent out (control packets) over useful packets sent out (data packet). Because the data packets were sent from a CBR source, the number of data packets was constant in the same duration of simulation, hence the control overhead ratio reflected the number of control packets sent through various protocols. The lower control overhead ratio makes more efficient network bandwidth.

Packet Delivery Ratio	$\frac{\text{Data packets delivered}}{\text{Data packets sent} * \text{number of receivers}}$
Control overhead	$\frac{\text{Control packets sent}}{\text{Data packets sent}}$

Table 2: Metrics definition

A. STM vs PUMA

Figure 2 illustrates the packet delivery ratio (PDR) archived by three protocols, varied by speed and group members. In both cases, STM and PUMA still maintain high ratio, around over 80%, although PUMA has slightly higher score than STM. Since PUMA manages topology as mesh, it can keep the network availability as high as possible. A careful analysis shows that because STM uses two rounds of network iteration to reconstruct the topology (for subroutines UPDATE and FIND), it may lead to decrease PDR when speed increases. In highly dynamic network using STM routing protocol, topology change occurs so frequent that it continues occurring when the procedure of two-round reconstruction adapted to previous change has not been completed yet. This is one drawback of STM. However, as shown in figure 2, STM, a tree-based protocol, has the PDR which almost keeps up with PUMA, one on top of mesh-based protocols.

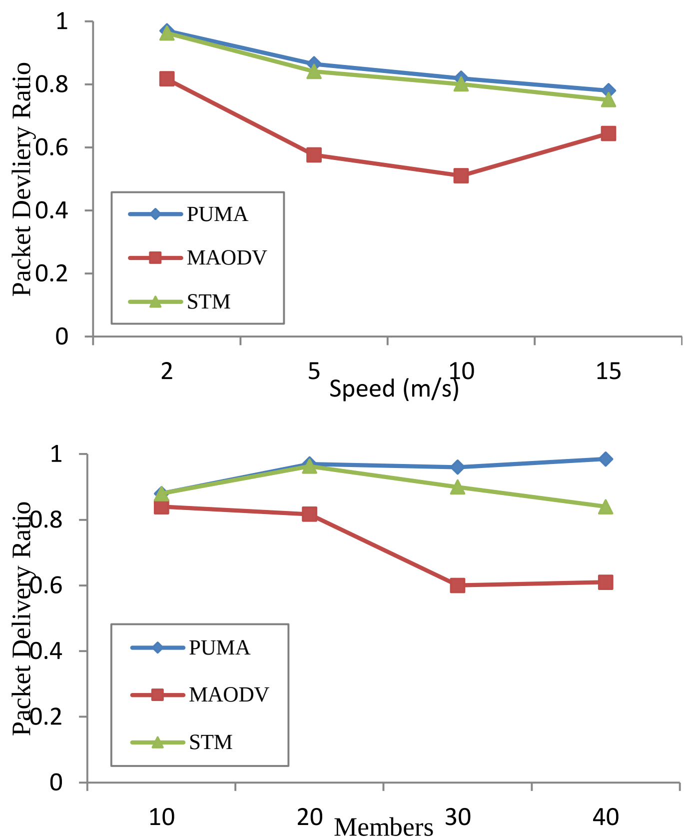
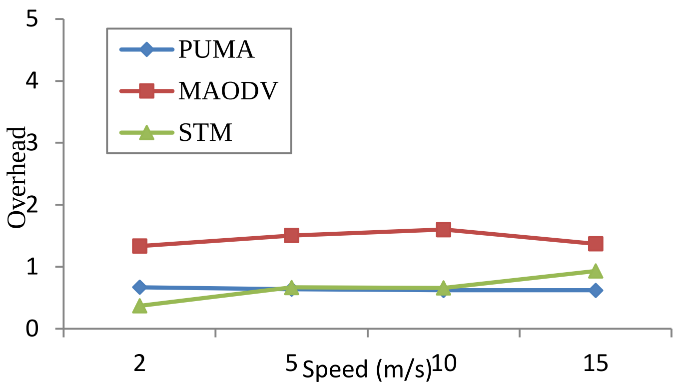


Figure 2: Packet Delivery Ratio in various scenarios



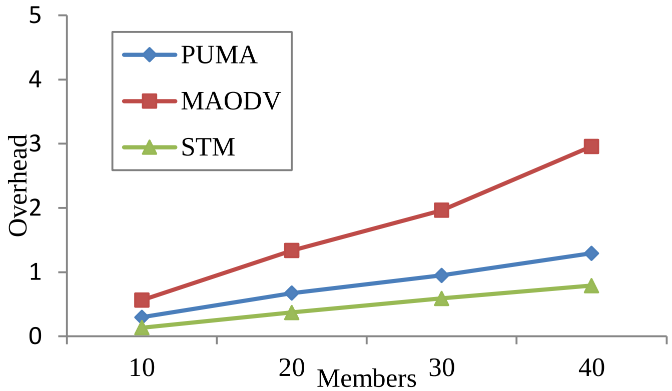


Figure 3: Overhead ratio in various scenarios

The overhead result which has shown in Figure 3 illustrates that as speed increases; the overhead ratio of STM is increased, while that one of PUMA is likely constant over various speeds. This is because topology in PUMA protocol is periodically updated through Core announcement message regardless the current topology, while STM protocol only reacts with each topological change. Therefore, when nodes move faster, STM protocol spend more control packet to reconstruct the spanning tree, which makes the overhead of STM protocol is increased.

However, in quite "stable" networks, as in our experiment with various group sizes (speed of nodes is 2 m/s), the overhead ratio of STM is nearly half of PUMA's overhead ratio. This result is expected as an evidence of OMST's theorem, the number of control messages in STM is optimal, that leads to lower overhead ratio of STM protocol than that of PUMA, although PUMA has attained a very low overhead [3].

B. STM vs MAODV

Based on simulation results shown in Figure 2,3; we can see that STM has a significantly better performance over MAODV in term of PDR as well as overhead ratio in both scenarios. We analyze deeply about control packets sent in each protocol and figure out that MAODV sent a large number of RREQ, RREP packets during simulation time. This is because multicast tree built by MAODV is unstable, especially when the mobile network is highly dynamic or has a large group size, the number of links breaking increases, hence the number of control packets increases, too. Since MAODV relies on unicast routing protocol to establish communication, it doesn't utilize efficiently the network bandwidth in mobile environment like STM and PUMA. In the other hand, tree created by MAODV, not like STM, is not optimized in term of message complexity. Therefore, the number of control packets sent by MAODV was growing larger and larger during simulation time, leading to packet lost due to collision, as said in [3]. MAODV doesn't have any mechanism to fix such a collision like that, leads to very low packet delivery ratio and high overhead.

IV. CONCLUSION AND FUTURE WORKS

Spanning Tree for Multicast protocol (STM) which is based on algorithm of optimal maintenance of a spanning tree (OMST), is low overhead multicast protocol by economizing the amortized message complexity,. With new data structure proposal to manage network topology, STM can simplify the operation of OMST's algorithm without changing the correctness and message complexity, make this protocol can run fast and efficiently in dynamic networks. Reliable communication in STM is accomplished by wireless broadcast only with retransmission enhancement, without requiring any underlying unicast routing protocol. Results from various scenarios of evaluation show that STM can achieve high or comparable delivery ratio while keeping lower control overhead than two well-known multicast protocols MAODV and PUMA. We intend to enhance the protocol to adapt better with topological changes in highly dynamic network.

REFERENCES

[1] B. Awerbach, I. Cidon, S. Kutten, "Optimal maintenance of a spanning tree", J. ACM 55(4), 2008.

[2] E. Royer and C. Perkins, "Multicast operation of the ad hoc on-demand distance vector routing protocol," Proceedings of Mobicom, 1999.

[3] R.Vaishampayan and J.J.Garcia-Luna-Aceves, "Efficient and robust multicast routing in mobile ad hoc networks", IEEE International Conference on Mobile Ad-hoc and Sensor Systems, pp. 304-313, 2004.

[4] S. Lee, W. Su, and M. Gerla, "On-demand multicast routing protocol (ODMRP) for ad hoc networks", draft-ietfmanet-odmrp-02.txt, 2000.

[5] J.Xie snd R.R.Talpade, A.McAuley, and M.Liu,"Amroute: Ad hoc multicast routing protocol", Mobile Networks and Applications, 2002.

[6] R. G. Gallager, P. A. Humblet, and P. M. Spira, "A distributed algorithm for minimum-weight spanning trees," ACM Transactions on Programming Languages and Systems, vol. 5, no. 1, pp. 66–77, 1983.

[7] O. S. Badarneh and M. Kadoch, "Multicast Routing Protocols in Mobile Ad Hoc Networks: A Comparative Survey and Taxonomy", EURASIP Journal on Wireless Communications and Networking, Vol. 2009, p. 42.

[8] C. S. R. Murthy and B. S. Manoj, "Ad Hoc Wireless Networks: Architectures and Protocols", Prentice-Hall, Upper Saddle River, 2004.

[9] Edsger W. Dijkstra and C. S. Scholten, Termination detection for diffusing computations, 1980, Inf. Proc. Letter 11, pages 1-4.

[10] B. Awerbuch,O. Goldreich,D. Peleg, and R. Vainish "A tradeoff between information and communication in broadcast protocols", J.ACM37(2), pages 238-256(1990).

[11] S.S. Doria, M.A. Spohn, "A Multicast Approach for Peer-to-Peer Content Distribution in Mobile Ad Hoc Networks," Wireless Communications and Networking Conference, pp.1-6, 2009.

[12] Y. Afek, E. Gafni, A. Rosen, "The Slide Mechanism with Applications in Dynamic Network", Proceedings of ACM PODC 1992, pp.35-46.

[13] The Network Simulator - ns-2: http://www.isi.edu/nsnam/ns/

[14] PUMA source code for NS2, http://sourceforge.net/projects/puma-adhoc

[15] MAODV source code for NS2, http://kunz-pc.sce.carleton.ca

[16] R.Vaishampayan and J.J.Garcia-Luna-Aceves, "Efficient and robust multicast routing in mobile ad hoc networks" (ROMANT version)