A Multicast Routing Algorithm Using Movement Prediction for Mobile Ad Hoc Networks

Huei-Wen Ferng, Hsing-Yu Chen, †Jeng-Ji Huang, and Wen-Yan Kao

Department of Computer Science and Information Engineering
National Taiwan University of Science and Technology, Taipei 106, Taiwan
†Department of Computer Information and Network Engineering
Lunghwa University of Science and Technology, Taoyuan County 333, Taiwan
E-mail: hwferng@mail.ntust.edu.tw

Abstract— **In mobile ad hoc networks, how to achieve the multicast communication is a challenging task due to the fact that the topology may change frequently and communication links may be broken because of users' mobility. To keep network connectivity, control packets are periodically sent within the system, resulting in high system overhead. To reduce the amount of control packets, a multicast routing algorithm for mobile ad hoc networks is designed in this paper. Utilizing movement prediction to estimate the life time of connections for every node pairs, the algorithm largely reduces the amount of control packets as compared to those using periodical path maintenance. Based on the estimate, a local path search rather than a global one is performed before the breakage of a path to get a backup path. This further lowers the system load. To avoid frequent path breakage, the algorithm chooses the path with the longest life among all possible paths. Using a simulation approach, we investigate the performance of the proposed algorithm through a comparison with an algorithm previously proposed in the literature. Based on the numerical results, we demonstrate that our proposed algorithm performs well.**

Keywords: Ad Hoc, Multicast, Routing, Prediction, Wireless Networks.

I. Introduction

With the collaboration of mobile nodes, communication is established in the mobile ad hoc network, which is popular in distributed collaborative computing, communication in battle fields, etc. In such a network, how to achieve the multicast communication, i.e., group communication, is a challenging issue because of frequent topology changes and breakage of communication links caused by users' mobility.

To well address the issue in the mobile ad hoc network, one needs first to maintain states of neighboring nodes. There are two main approaches, i.e., *soft state approach* and *hard state approach*. The former one periodically updates states of neighboring nodes, while state updates are performed only after the path is broken for the latter one. In comparison, the latter one causes less overhead to the whole system but does not behave robustly. On the other hand, multicast algorithms can usually be categorized into mesh-based or tree-based types according to the path topology. More than one connections may exist between the transmitting node and the receiving node in the mesh-based type, while only one path exists between them in the tree-based one. Although better robustness is achieved by the mesh-based approach, its routing efficiency is obviously lower than the tree-based one.

In the literature, the ad-hoc on-demand distance vector (AODV) [1] is a routing algorithm designed for the unicast communication using the flood approach. It is simply an on-demand routing algorithm. That is, when the source has data packets destined for its group members and if no path is immediately available between the sender and the receiver, the source will perform a path search procedure to determine a path for data flow. Once the path is broken due to users' mobility, a new path is globally searched. As for the on demand multicast routing protocol (ODMRP) [2], it is a mesh-based routing protocol. In ODMRP, the path information and group members are updated through periodically broadcasting a *Join Request* packet by the source and replying the *Join Reply* packets by the group members. For those nodes receiving the *Join Reply* packet, a decision is made about whether to join as a member of the forwarding group or not. This approach may lead to multiple coexisting paths between the sender and each of its group members. Thus, breakage of some paths will not definitely result in loss of packets. This ensures robustness in packet delivery. However, the periodical maintenance of path information and group members requires a large amount of control packets.

Thus, a tree-based multicast routing algorithm with the hard state update is designed in this paper to minimize the overhead introduced by exchanging control packets for path maintenance. The robustness of our algorithm is strengthened by movement prediction using the information provided by global position system (GPS) employed to estimate connection life time between two neighboring nodes. Such prediction enables us to search a new/backup path with the longest life within a local area before the breakage of a path. This saves a large amount of control packets caused by a global search. As can be seen in Sec. III, the proposed algorithm not only significantly reduces the system overhead but also provides better routing efficiency as compared to ODMRP. In the following, the proposed algorithm is called tree-based multicast routing algorithm with movement prediction (TMRAMP) for simplicity.

0-7803-8887-9/05/$20.00 (c)2005 IEEE

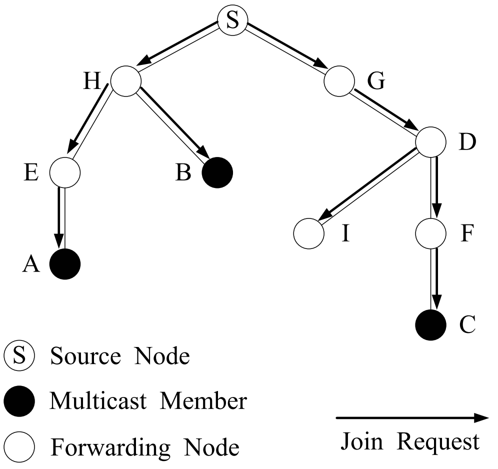

Fig. 1. The procedure of the routing tree construction.

The rest of the paper is organized as follows. Section II details the proposed algorithm and Section III investigates its performance through numerical experiments. Finally, Section IV concludes the paper.

II. Description of TMRAMP

The algorithm is composed of three parts. The first part is the movement prediction which uses GPS to estimate the connection life time between two neighboring nodes based on results in [3] and [4]. The second part is the routing protocol which describes how the multicast tree is built up. Finally, the remaining part relates to the local path search and recovery. These three parts are described as follows.

A. Movement Prediction

Assume that the moving speeds, coordinates, and directions of two mobile nodes, say node 1 and node 2, are given, then we can calculate the connection time D_t using the following equation [3]:

$$D_t = \frac{-(ab+cd)+\sqrt{(a^2+c^2)X^2-(ad-cb)^2}}{a^2+c^2} \quad (1)$$

where $a = v_1\cos\theta_1 - v_2\cos\theta_2$, $b = x_1 - x_2$, $c = v_1\sin\theta_1 - v_2\sin\theta_2$, $d = y_1 - y_2$, X represents the communication distance of each node, (x_1, y_1) and (x_2, y_2) are coordinates for nodes 1 and 2, v_1 and v_2 (θ_1 and θ_2) are moving speeds (moving directions) of nodes 1 and 2.

B. Routing Protocol

When the source node (sender) has data to send to its group members, it first broadcasts a *Join Request* packet which includes the following fields: SID (Source ID), MGID (Multicast Group ID), SN (Sequence Number), RCT (Route Connection Time) which is an estimated route connection time, HC (Hops Count), UNID (Upstream Node ID), TTL (Time To Live), etc. A node upon receiving a *Join Request* packet determines if it is a duplicate by examining the SID and SN fields. If yes, the packet is immediately discarded; otherwise, an examination is performed next whether the value of HC has exceeded a pre-specified threshold or not. If not, movement prediction is applied to estimate the link connection time (LCT) between this node and its upstream node. The minimum value of the estimated LCT and the RCT is set to the new value of RCT, and the upstream node for this *Join Request* packet is recorded. After that, the modified packet is broadcasted to its neighbors. For a group member, it further chooses the path with the largest RCT since multiple *Join Request* packets may be received from different paths. Of course, a *member routing table* is maintained at each node such that *Join Reply* packets sent by group members are able to return back to the sender along the chosen paths.

The whole process of routing tree construction is illustrated in Fig. 1, where node S is the sender, nodes A, B, and C are receiving nodes, and other nodes are forwarding nodes. All nodes in Fig. 1 are thus on-tree nodes.

C. Local Path Search and Recovery

Due to users' mobility, the connection between nodes may be broken. So, the path search/recovery is inevitable. In this paper, we implement local path search and recovery based on the information obtained from GPS and the movement prediction.

We assume that all necessary information is available and is put into packets so as to make GPS work. When receiving a data packet from a node, this information can be used to estimate the connection time which is then used to determine the time when the link will terminate. This is achieved by setting a threshold *BeginHandoff*. When the estimated connection time falls below the threshold, it indicates that link breakage is going to happen. In this case, the node will issue the *Rejoin* packet to its neighbors. A neighboring node upon receiving the packet first checks whether the packet is duplicate or not. If not, the node further examines whether itself is an on-tree node along the multicast tree or not. If yes, a *Reply Rejoin* packet is sent; otherwise, the *Rejoin* packet is broadcasted to its neighbors if the packet's TTL is not yet reached the threshold value. For an on-tree node, GPS information is collected to estimate the connection life time before returning back the *Reply Rejoin* packet to the disconnected node. For the disconnected node, a path with the longest life time is chosen as a new path. In case no any path can be found within a certain amount of time, the disconnected node tries to contact with the other on-tree nodes next to a node which has been examined previously by repeating the packet broadcast with scope one hop larger. Such a procedure is continued until an on-tree node is found so that a new path is established eventually.

Fig. 2 depicts the process to locally search a new path after link breakage occurs between nodes A and E. As can be seen, the local path search is initiated by the disconnected node A and is repeatedly performed until on-tree nodes D or F

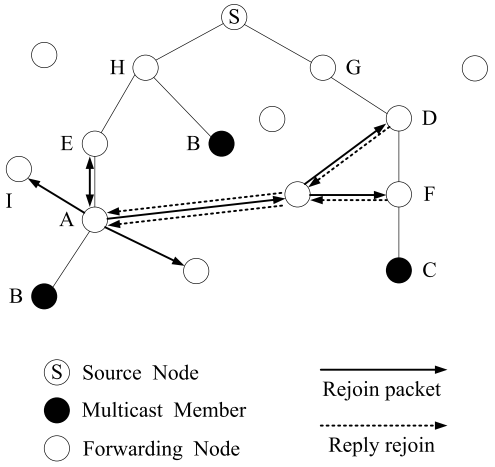

Fig. 2. Route maintenance when a route breaks.

are found; then to choose a new path between nodes A and D or between nodes A and F will depend on the estimated connection life time. It should be noted that node A has no idea about the location of the other on-tree nodes before the local path search begins. Furthermore, only local flooding is used in TMRAMP rather than global one in AODV.

III. Numerical Examples and Discussions

A. Simulation Arrangements

Using GlomoSim [5] developed by UCLA based on PARSEC [6], our simulation is run on an IBM compatible PC. Within an area of 1000 × 1000 square meters, 50 nodes are generated initially. Each node moves according to the model of random way-point; that is, the coordinate of the next destination for each node will be randomly generated; then the node moves to that destination at a speed confined to a pre-specified value; after moving to the new position, the node stays for a while before moving to its next position. Assume that the pause time is 30 seconds, the communication distance is 250 meters, transmission rate is fixed at 2 Mbps, the constant bit rate (CBR) traffic model is used with packet size of 512 bytes and generation rate of 10 packets per second. For simplicity, we assume that the group size of the multicast session is fixed and each simulation runs for 200 seconds. Based on this environment, a comparison is made between TMRAMP and ODMRP under various degrees of mobility and group sizes.

B. Performance Metrics

To gauge the performance of TMRAMP and ODMRP, we employ the following three performance metrics.

1) *(Data) packet delivery ratio* which is the number of data packets received by all group members divided by the number of total packets expected to be received by all group members.

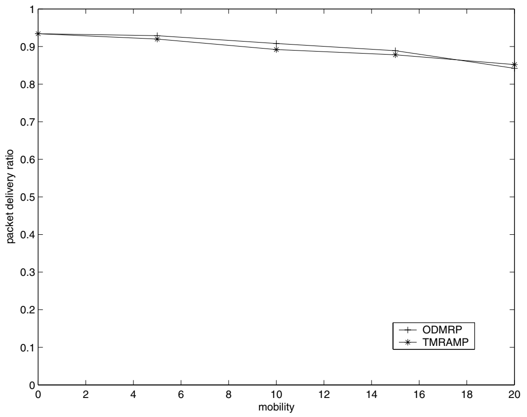

Fig. 3. Packet delivery ratio vs. mobility.

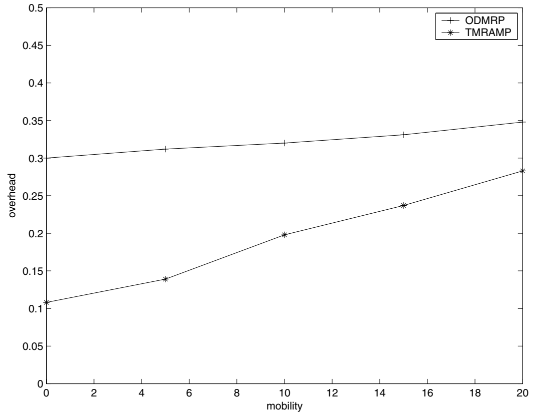

Fig. 4. Overhead vs. mobility.

2) *Number of control packets transmitted per data packet received* which is the number of control packets transmitted by all nodes divided by the number of data packets received by all group members. It can be used to reflect *overhead*.

3) *Number of data packets received per data packet transmitted* which is the number of data packets received by all group members divided by the number of data packets transmitted by all nodes. It is used to represent *routing efficiency*.

C. Simulation Results

1) Effect of Mobility

First of all, the effect of mobility is investigated under the setting that group size is 5 and moving speed is varied from 0 to 20 km/hr. In Fig. 3, we have observed that ODMRP performs a bit better in terms of packet delivery ratio than TMRAMP as the moving speed increases because mesh-based routing is used in ODMRP so that multiple paths may exist to improve robustness. However, ODMRP incurs higher overhead as compared to TMRAMP shown in Fig. 4. TMRAMP outperforms ODMRP by reducing 20% to 60% of overhead.

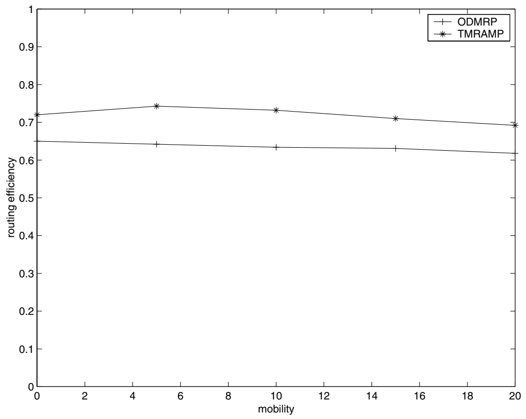

Fig. 5. Routing efficiency vs. mobility.

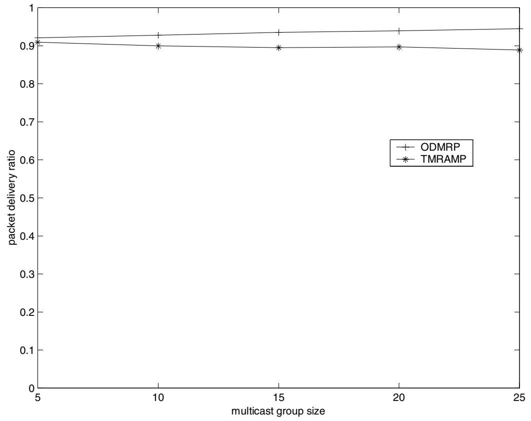

Fig. 6. Packet delivery ratio vs. group size.

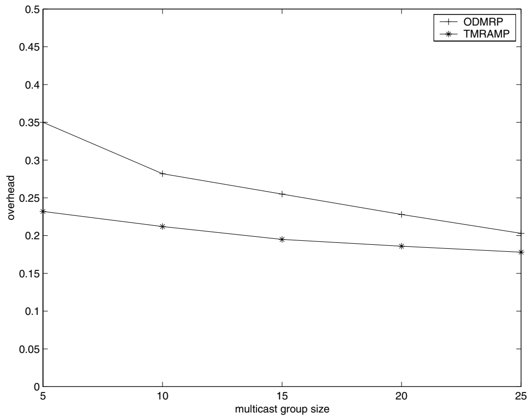

Fig. 7. Overhead vs. group size.

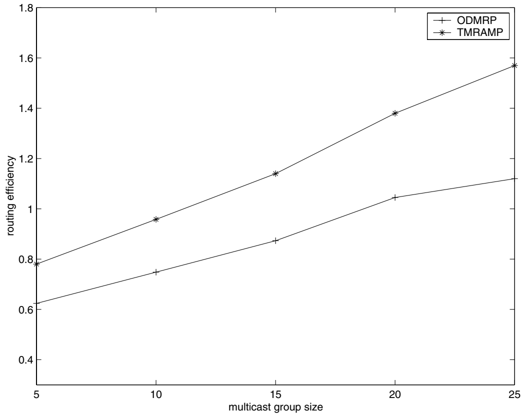

Fig. 8. Routing efficiency vs. group size.

In addition, the comparison on routing efficiency is also shown in Fig. 5 under various moving speeds. It is obvious that TMRAMP performs better by gaining 10% to 15% more routing efficiency than ODMRP. The performance gain in routing efficiency comes from the fact that only one path exists in our proposed algorithm, while multiple paths may exist simultaneously in ODMRP. This makes more nodes in ODMRP be involved in forwarding packets, thus decreasing the overall routing efficiency.

2) Effect of Group Size

Next, the effect of group size is investigated under the setting that the moving speed is fixed at 10 km/hr, while the group size is varied from 5 to 25. In Fig. 6, we witness that packet delivery ratio for TMRAMP remains almost constant under various group sizes, while it increases for ODMRP as the multicast group grows up and is a bit higher than TMRAMP. Since mesh-based routing is used in ODMRP, more backup paths are established between the sender and all its group members when the group size increases. Thus, better packet delivery ratio is ensured.

On the other hand, as shown in Fig. 7, the overhead under both schemes decreases as the group size increases. It is mainly due to the fact that more data packets are received at the receiving ends when the group size becomes larger. Although control packets may also increase as the group size grows, the increasing rate of control packets is not larger than that of data packets. Therefore, the overhead decreases as the group size increases and this phenomenon is especially prominent for ODMRP. Again, it is noted that about 10% to 30% less in overhead are observed for TMRAMP due to the fact that TMRAMP gets rid of periodically sending of control packets to maintain routing paths. This also explains why the decrease of overhead in TMRAMP is less sensitive to the increase of group size.

Regarding the routing efficiency, it increases as the group size grows under both schemes as shown in Fig. 8. As mentioned previously, more data packets are received at all receiving ends when the group size becomes larger. Although the total packets forwarded also increase when the group size rises, the increasing rate of total received data packets is larger than that of total packets forwarded, thus making the routing efficiency get higher when the group size becomes larger. In general, about 40% improvement can be achieved by TMRAMP as compared to ODMRP. This indicates that a scheme using tree-based routing is more suitable than that using mesh-base routing when applied to an environment having a large group.

IV. Conclusions

In this paper, a multicast routing algorithm called TMRAMP is proposed to reduce the overhead introduced by exchanging control packets. TMRAMP is a tree-based algorithm with the hard state updates with the aid of movement prediction to estimate the life time of links for every communicating node pairs. Based on the estimate, a local path search rather than a global one is then performed before the breakage of a path to obtain a backup path. Moreover, the algorithm picks up the path with the longest life among all possible paths to avoid frequent path breakage. Through extensive simulations, comparisons in terms of three performance metrics have been conducted between TMRAMP and ODMRP under various degrees of mobility or group sizes. Based on numerical results, we conclude the followings. First, although ODMRP performs a bit better in terms of packet delivery ratio, a higher overhead than TMRAMP is introduced. About 20% to 60% higher under various moving speeds and 10% to 30% higher under various group sizes in overhead are observed. Second, TMRAMP outperforms ODMRP in routing efficiency by 10% to 15% under various moving speeds and up to 40% under various group sizes. Third, a scheme using tree-based routing is more suitable than that using mesh-base routing when applied to an environment with a large group. Hence, we suggest TMRAMP to be used in mobile ad hoc networks based on the above observations.

Acknowledgment

The first author thanks the partial support in finance by the National Science Council, Taiwan under Contract NSC 93-2219-E-011-007.

References

[1] C. E. Perkins and E. M. Royer, “Ad-hoc on-demand distance vector routing,” in *Proc. IEEE WMCSA ’99*, pp. 90–100, Feb. 1999.

[2] S. J. Lee, M. Gerla, and L. Zhang, “On demand multicast routing protocol,” in *Proc. IEEE WCNC ’99*, pp. 1298–1302, Sept. 1999.

[3] W. Su, S. J. Lee, and M. Gerla, “Mobility prediction in wireless networks,” in *Proc. IEEE MILCOM ’00*, pp. 491–495, Oct. 2000.

[4] W. I. Kim, D. H. Kwon, and Y. J. Suh, “A reliable route selection algorithm using global positioning systems in mobile ad-hoc networks,” in *Proc. IEEE ICC ’01*, pp. 3191–3195, June 2001.

[5] UCLA Parallel Computing Laboratory and Wireless Adaptive Mobility Laboratory, “GloMoSim: A scalable simulation environment for wireless and wired network systems,” http://pcl.cs.ucla.edu/projects/domains/glomosim.html.

[6] R. Bagrodia, R. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, and H. Y. Song, “PARSEC: A parallel simulation environment for complex systems,” *IEEE Computer*, vol. 31, pp.77–85, Oct. 1998.