An Efficient Core Migration Protocol for QoS in Mobile Ad Hoc Networks*

Manish Kochhal † manishk@cs.wayne.edu
Loren Schwiebert ‡ loren@cs.wayne.edu
Sandeep Gupta § sandeep.gupta@asu.edu
Changli Jiao † changli@wayne.edu

†Dept. of Electrical and Computer Engineering
‡Dept. of Computer Science
Wayne State University
Detroit, MI 48202

§Dept. of Computer Science and Engineering
Arizona State University
Tempe, AZ 85287

Abstract

This paper introduces a core migration protocol that provides QoS for multicast applications in Mobile Ad hoc Networks. Multicasting is usually accomplished by constructing a multicast tree and transmitting the packets over this tree, replicating packets at the branch points. In a group shared multicast tree, the choice of the root (core) plays an important role in influencing the organization of the tree and affecting the performance of packet delivery. The objective is to construct a tree whose leaves achieve the desired qualities of the multicast application. Our proposal differs from previous work on core selection and migration which typically rely on complete knowledge of network topology and also almost complete network participation for providing this information.

Keywords: mobile ad hoc networks, multicast routing protocols, core based tree, core migration.

1 Introduction

1.1 Multicasting in Mobile Ad hoc Networks

An ad hoc network is a network that is constructed on demand to enable reliable communication among portable computers in a mobile environment without the need of a central manager or additional administrative work. A mobile ad hoc network [6] in addition to having an unpredictable and dynamic topology also has several severe constraints on its resources such as bandwidth and battery power which is due to its wireless nature. These constraints and characteristics of Mobile Ad hoc Network (MANET) environments make the multicast problem more complex [7].

Multicast routing protocols generally build trees to deliver messages to a multicast group. Multicast routing protocols are of different types: sender-initiated and receiver-initiated protocols. In sender-initiated protocols, such as the DVMRP [8] and PIM-DM [4], the receiving group is assumed to be fairly dense and the sender initiates the multicast assuming all routers in the network are interested in receiving the multicast. Routers that do not need to forward the message to group members prune themselves from the multicast tree. In receiver-initiated protocols like CBT [1] and PIM-SM [2, 3], the receivers initiate their own connection to the tree. In each of these receiver-initiated protocols, a well-known router exists that accepts connection requests from other routers. This router is known as the “rendezvous point” in PIM. and the “core” in CBT. The returning acknowledgment builds a branch of the tree back to the initiator along the reverse path of the connection request.

An effort to find a minimal cost tree in terms of qualities like delay, bandwidth or hop-count, for a given subset of nodes gives rise to the well known NP-complete *Minimal Steiner Tree* problem.

1.2 Core Selection and Migration Fundamentals

The core selection and migration protocol is designed to construct core based trees using one of several different quality of QoS metrics. The core selection algorithm tries to find that network node (*i.e.*, a router) whose use as the core of the multicast group

---

*This material is based upon work supported by the National Science Foundation under Grants DGE-9870720 and ANI-0086020.

results in an optimal multicast tree with respect to a desired performance metric. Core migration occurs after selecting a new core for a multicast group.

The following factors may contribute to overhead in the core selection and migration algorithm: number of nodes participating implicitly or explicitly in core selection, number of messages exchanged, number of times the algorithm is executed, and finally the optimality of the core selected in terms of the desired performance metrics.

In this paper, we propose an efficient QoS sensitive/responsive algorithm for MANET which moves the core implicitly on a hop-by-hop basis, so as to continuously adapt to network variations and finally, in steady state, to reach an optimal position corresponding to the desired QoS. A detailed description of the protocol specifically meeting the delay requirements of a multicast application are presented, along with proofs that the protocol moves the core toward the optimal location and reaches that location when the network is stable.

2 Previous Work

Typical approaches proposed [9] to select a core can be broadly classified into administrative or static selection, or dynamic center selection.

Administrative selection usually chooses the first node, source or member, of the multicast group (or network) as the core which remains fixed irrespective of changes in the network topology. Administrative selection may sometimes lead to a router being selected as the core/center whose optimality with respect to various performance metrics may degrade, depending upon whether or not the changes in network topology make the relative location of the core more or less efficient.

The dynamic center selection algorithm tries to adapt to the network dynamics by invoking the core selection algorithm multiple times over the lifetime of a group until the tree branches reflect the desired QoS requirements. Dynamic center selection algorithms can be further classified into two categories: explicit and implicit center selection.

An explicit core selection algorithm requires that whenever the algorithm is run, all network nodes or members of the multicast group compute their weight functions and exchange weights among themselves, so as to select the node with the minimum weight as the core. This kind of algorithm also requires a termination condition expressed in terms of, say, number of hops or links to traverse, number of members or nodes participating, or a timeout period, so as to successfully terminate the algorithm.

Thaler and Ravishankar [9] propose two minimization protocols, where the n best nodes are found in a distributed and automated fashion by minimizing a weight function using a list of group members or sources. Since the algorithm requires complete participation among either all network nodes or all group members or all sources, it is applicable only to a fixed wired network like the Internet.

Fleury *et al.* [5] propose several core selection heuristics. Each heuristic defines a method to explicitly identify a set of candidate core nodes from which the core can be selected. The set of nodes used in any of these heuristics ranges from the entire network or group, to a specific multicast tree. All of the heuristics proposed require knowledge of the network topology, which is collected at the core or any router by means of the Join_Requests that contain the member-to-core paths delivered to the core. These Join_Requests may not reflect the dynamics of the MANET.

On the other hand, an implicit core selection algorithm requires only *one* node, the core, to monitor the network over regular intervals of time for the desired QoS metric. For example, the core can monitor the delay by calculating the time difference relative to itself between the transmission of a packet (or a group of packets) and their corresponding acknowledgments. The algorithm then makes a decision according to a threshold, which may be either fixed or computed dynamically. For example, the algorithm may decide to migrate the core if one branch consistently has a cost of more than 5% over the desired QoS. Thus, intuitively, an implicit algorithm should involve less overhead than an explicit one.

3 New Center Selection / Migration Protocol

3.1 Design Philosophy

Our aim is to design an efficient core migration protocol for ad hoc networks that migrates the core until the tree branches reflect the desired QoS requirements of the multicast application. If the differences in desired qualities are not above a threshold, we do not to migrate the core, thus avoiding frequent core migrations. An ideal algorithm to locate the center of a multicast tree in an ad hoc network should have an implicit and minimum network participation to minimize a performance metric, such as delay. It should have minimal interaction between neighboring nodes during core selection, migration, and dissemination of the new core location. It should support incremental core migration as opposed to distant migration, thus ensuring minimal overhead in terms of the messages exchanged during transfer of the core state and also reducing the risk of failure, which may be aggravated by a distant core migration in a wireless mobile ad hoc network.

Finally, of the several weight functions that can be minimized to make a decision on core placement, our protocol, as an example, uses delay as the function to be monitored as opposed to parameters like minimum/maximum/average distance or maximum diameter. The reason is the other parameters are more appropriate for a static network where knowledge of network topology is relatively fixed and can be obtained more easily. Additionally, we believe that our protocol is generic as it can be easily extended to include other QoS parameters like hop-count, bandwidth, etc. or a combination of these.

3.2 Protocol Design

With respect to figure 1 the arrows indicate how routers 1, 2, 3, and 4 reach the core with acknowledgments for the data packets multicast.

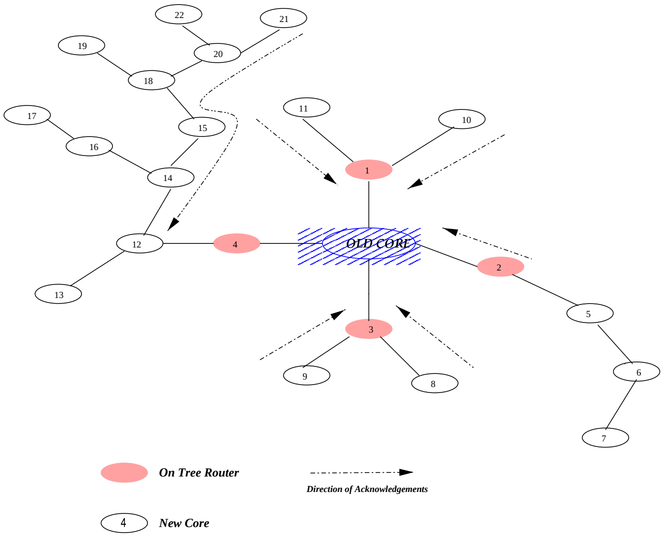

Figure 1: Example Core Based Multicast Tree

The core records the history of the delays in terms of the relative time difference between the packets sent to and the corresponding acknowledgments received from the respective subtree branches to the core.

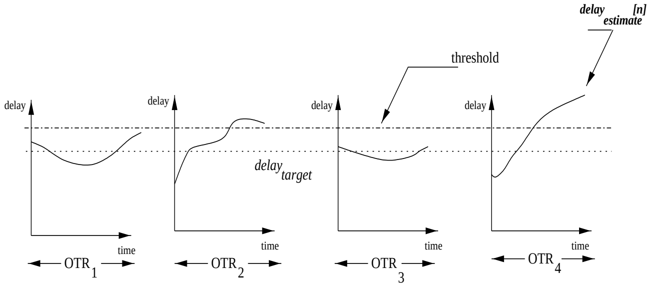

Figure 2: History of Delays used for Core Selection

Figure 2 depicts a possible history of delays accumulated by the core from the four on-tree routers (OTRs). Let us define the following array for analysis: $delay_{OTRj}[n]$: An array used by the core to store the samples of delay obtained from each of the m neighboring on-tree routers: $1 \leq j \leq m$.

In order to decide which branch is showing a consistent delay, the core has to compute a parameter, $delay_{estimate}$, for each of the m neighboring on-tree routers through which it has monitored the delay. The $delay_{estimate}$ can be computed from the recursive equation given below:

$$\begin{aligned} delay_{estimate}[i] \quad &= \quad \alpha delay_{estimate}[i-1] \\ &\quad\quad +(1-\alpha) delay_{OTRj}[i] \end{aligned}$$

That is, the $delay_{estimate}$ at step i is $\alpha \times delay_{estimate}$ at step $(i-1)$ and the $delay_{OTRj}$ at step i, where $0 \leq \alpha \leq 1, 0 \leq i \leq n,$ and$1 \leq j \leq m$.

The $delay_{estimate}$ parameter is computed by the core using both the history of delays and the weighting factor α. A small value of α, say α = 0.2, gives more weight to earlier values whereas $\alpha > 0.5$ gives more importance to later values. The $delay_{estimate}$ gives the core a comprehensive local view of the imbalances in delay over its subtree branches, which is basically caused by asymmetric changes in its network topology. The $delay_{target}$ is the desired QoS specified to the core by the multicast application and *threshold* is the maximum limit above the required QoS value which the multicast application can safely tolerate. The $delay_{estimate}$ changes with the network topology and therefore it is used by the core placement protocol along with *threshold* to move the core to the next hop in an attempt to minimize imbalances in delay. Now, referring to figure 2, it can be concluded that the OTR_4 delay graph shows a *larger and persistent deviation* from $delay_{target}$ that is, for $OTR_4, delay_{estimate} > delay_{target}$. Hence, OTR_4 is the next probable position for the core.

3.3 QoS Metrics Analyzed

In this paper, in addition to hop-count and delay, we also analyze an additional QoS metric known as hop fan-out count. The hop fan-out count, in addition to the the number of hops, also takes into account the fanout at a node. Thus, hop fan-out count can be considered as a measure of minimum contention for the shared wireless medium at a node.

3.4 Core Migration Steps

A detailed diagram illustrating the sequence of message exchanges during core migration is given in figure 3. The core migration protocol bases its decision on the history of the delays recorded in terms of the relative time difference between the packets sent to and the corresponding acknowledgments received from the respective subtree branches to the core. Since the nodes are mobile, it is possible for nodes to have neighbors dynamically changing with time. Therefore a node would need to *synchronize* its data with its children and its data acknowledgment (it had received) with its current parent depending upon whether that node's current children had in the past forwarded such an acknowledgment upstream. A node encountering new neighbors enters this Synch Stage consisting of Synch Query and Synch Reply that tries to maintain this data and data acknowledgment relationship between a node's parent and its children. Similar is the case for a Core Information message and its associated Core acknowledgment. To avoid past acknowledgments repetitively reaching the core, the core broadcasts a Stabilized Ack message for those data acks (and core acks) that have already been recorded as history from all the other non-core multicast group members.

The core migration sequence can then be understood as consisting of building history, synchronizing data and its associated ack with neighbors, stabilizing all the data acks, sending the core information, synchronizing core information and its associated core ack with neighbors, and finally a core state transfer message unicast from the old core to the new core.

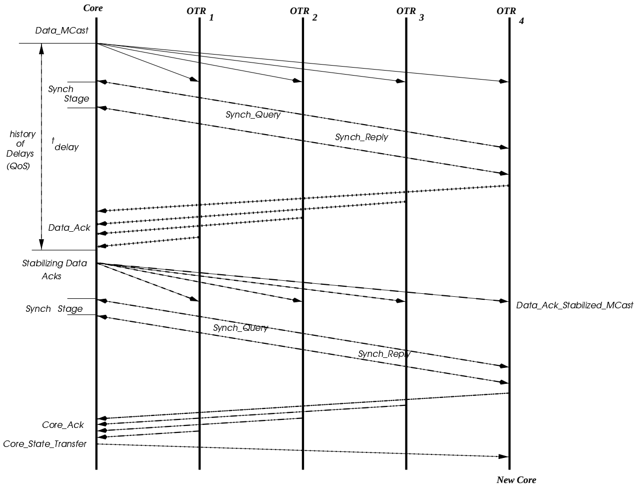

Figure 3: The Core Migration Protocol Message Sequence

4 Correctness

Theorem 1 *When the network topology is static, the core migrates to an optimal router.*

Proof. The core monitors the delay on the subtree branches by way of the time (relative to itself) at which multicast packets are sent and the corresponding acknowledgments received. For a given multicast tree topology, the delay monotonically increases as the message travels greater distances. Moving the core toward the branch with the highest delay decreases the delay on that branch. The delay increases on the other branches, but that delay remains less than the previous delay on the longest path. Since the network topology is "static", the core continuously migrates itself on a hop-by-hop basis until it reaches a position where the differences in delay are not above a *threshold* necessary to migrate. □

Theorem 2 *When the network topology is changing, the core tries to adapt to network dynamics and at steady state reaches to an optimal router.*

Proof. When the network topology changes, the $delay_{estimate}$ samples that is computed over T will also change accordingly. For smaller T the core makes a decision on core migration in less time, resulting in frequent core migrations to adapt to network dynamics. At steady state, the core would migrate to an optimal position where it equalizes the delay coming from all the subtree branches. Thus, the core is always moving toward a currently optimal position. □

5 Preliminary Simulation

We used SimJava (version 1.2) as our discrete event simulator. For ease in analysis, let us consider a scaled down example network consisting of 12 nodes moving randomly at a low rate within a 2-dimensional region of space. At any given simulation time, all the nodes are assumed to be connected to at least one other node. The wireless channel has been modeled as a bidirectional shared medium with collision avoidance. The delay is assumed to be equal in either direction – unit delay for each wireless transmission. The simulation starts with the topology in figure 4a, with node 0 selected arbitrarily as the core. The $delay_{target}$ specified during the simulation was one-fourth the number of nodes i.e. equal to 3 units with a 25% margin above it as threshold. At time 0, the core sends the first data packet and at time 15.2 (refer figure 4b), the core stabilizes the corresponding data acknowledgment for all non-core nodes. During this time, the core monitored OTR_2 having cost more than its other neighbors. The following average values were obtained for OTR_2, hop count equal to 2, hop-fanout count equal to 4 and delay equal to 8. Although the network was not static during simulation, OTR_2 consistently had more children than under any of the other core's neighbors. Hence, the core migrates to its next hop neighbor *i.e.*, OTR_2.

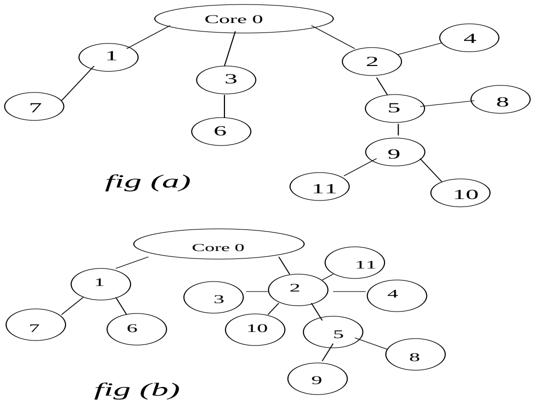

Figure 4: (a) Initial Network Topology for simulation and (b) Snapshot of Network Topology when core stabilizes the first Data Ack

6 Conclusions

In this paper we have proposed a simple, efficient, and reliable core migration protocol for providing QoS in MANET. Our proposal differs from previous proposals as it does not rely on complete knowledge of network topology and also almost complete network participation for providing this information. Our protocol achieves the same result as [9] in a stable network, although with some additional delay. Both the sampling time period T and the method used to calculate the *threshold* and $delay_{estimate}$ determine the effectiveness of the core selection protocol in adapting to the network dynamics and associated protocol overhead. For smaller T the core makes a decision on core migration in less time and this results in frequent core migrations and more overhead. Our future research involves extending our QoS sensitive protocol to also meet the other QoS requirements like bandwidth. We are presently working on simulations to study the tradeoff between the responsiveness of the protocol to the ad hoc network dynamics and the value of the *threshold.*

References

[1] A. Ballardie, P. Francis, and J. Crowcroft. Core Based Trees (CBT). In *ACM SIGCOMM*, pages 85–95, September 1993.

[2] D. Estrin *et al.* Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification. RFC 2362, Internet Engineering Task Force, June 1998.

[3] Stephen Deering *et al.* Architecture for Wide Area Multicast Routing. In *ACM SIGCOMM*, September 1993.

[4] D. Estrin, D. Farinacci, A. Helmy, V. Jacobson, C. Liu, and L. Wei. Protocol independent Multicast (PIM), Dense Mode Protocol Specification. Internet Draft draft-ietf-idmr-PIM-DM-spec-01.txt, Internet Engineering Task Force, September 1996.

[5] Eric Fleury, Yih Huang, and Philip K. McKinley. On the Performance and Feasibility of Multicast Core Selection Heuristics. In *Seventh International Conference on Computer Communications and Networks, Lafayette, Louisiana*, October 1998.

[6] Mobile Ad hoc Networks (MANET) Charter. MANET Homepage. http://www.ietf.org/html.charters/manet-charter.html.

[7] Young-Bae Ko and Nitin H Vaidya. Geocasting in Mobile Ad hoc Networks: Location-Based Multicast Algorithms. In *Second IEEE Workshop on Mobile Computing Systems and Applications*, pages 101–100, 1999.

[8] T. Pusateri. Distance Vector Multicast Routing Protocol. Internet Draft draft-ietf-idmr-dvmrp-v3-09.txt, Internet Engineering Task Force, September 1999.

[9] David G. Thaler and Chinya V. Ravishankar. Distributed Center-Location Algorithms. *IEEE Journal on Selected Areas in Communications*, 15(3):291–303, April 1999.