

A Case for Scalable Multicast Tree Migration

Anirban Chakrabarti and G. Manimaran
Dependable Networking and Computing Laboratory
Department of Electrical and Computer Engineering
Iowa State University
Ames, IA 50011, USA

Abstract—The proliferation of QoS-aware group applications coupled with the limited availability of network resources demands for efficient mechanisms to support QoS multicasting. During a life-cycle of a multicast session, three important events can occur: membership dynamics, network dynamics, and traffic dynamics. The first two are concerned with maintaining a good quality (cost) multicast tree taking into account dynamic join/leave of members, and changes in network topology due to link/node failures/additions, respectively. The third aspect is concerned with flow, congestion, and error control. There has been many solutions proposed for dealing with each of these issues. However, the issue of *tree migration* has not been addressed as part of these solutions. In this paper, we highlight the importance of tree migration as a mechanism for handling membership and network dynamics in core-based¹ multicasting, prove that it is NP-Complete, and propose four heuristic algorithms for it. The proposed algorithms are evaluated under two performance metrics: service disruption and resource wastage. Our simulation studies show that two of the algorithms offer comparable performance to that of the other two, in addition to being highly scalable and easily implementable.

Keywords—Core-based multicasting, Group dynamics, Tree migration, Service disruption, Resource wastage.

I. INTRODUCTION

The proliferation of QoS-aware group applications associated with recent advancements in high-speed networking is driving the need for efficient multicast communication services satisfying the QoS requirements of such applications [1]. Multicasting has been a popular mechanism for supporting group communication. In a multicast session, the sender transmits only one copy of each message that is replicated within the network and delivered to multiple recipients (receivers). For this reason, multicasting typically requires less total bandwidth than separately unicasting messages to each receiver.

Multicast routing protocols can be classified into two main approaches: source-based trees (SBT) and center-based. Source-based approach uses a shortest path tree rooted at the sender/source. The branches of the tree are usually the shortest delay paths from the sender to each group member. When this approach is employed for many-to-many multicasting, a separate multicast tree must be constructed for each source, thus making the approach not scalable. Source-based routing is currently employed in Distance-Vector Multicast Routing Protocol (DVMRP) [2].

The other type of protocols, called *center-based tree*, constructs a multicast tree spanning the members whose root is the center or core node. These protocols are highly suitable for sparse groups and scalable for large networks. Core Based Tree [3] is a well known example of this type. In this protocol, when a node wishes to transmit a message to the multicast group, it sends the message towards the core. The message is distributed to the group members along the path to the core, and once the message reaches the core, it is distributed to the

¹Tree migration is applicable to source-based multicasting as well.

remaining group members. Requests to join or leave the group are processed by sending the request towards the core. When a join request reaches an on-tree node, the tree node becomes the point of attachment for the new node. When a node leaves the group, a part of the tree that supported the leaving node is pruned.

Other than these protocols, also some hybrid routing protocols like sparse-mode PIM (PIM-sparse) [4] and the Multicast Internet Protocol (MIP) [5] have been proposed. These allow a receiver to switch from the shared tree to shortest path tree.

A network architecture that aims to provide complete support for multicast communication is burdened with the task of managing the multicast sessions in a manner transparent to the users. This goal of transparent multicast service imposes specific requirements on the network implementation. To understand the different functionalities that such a network must provide shown in Figure 1, the various steps and events that can take place in the “*life-cycle*” of a typical multicast session. The sequence of phases/steps relevant to the multicast session are (i) multicast group (session) creation, (ii) multicast tree construction with resource reservation, (iii) data transmission, and (iv) multicast session tear-down. In Figure 1, Tree Rearrangement

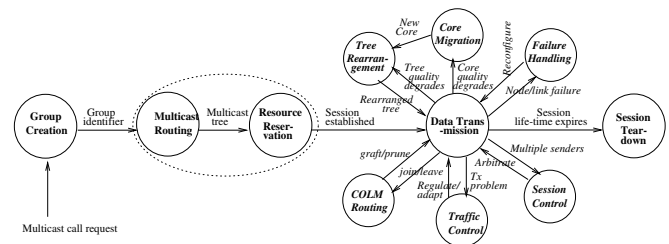


Fig. 1. Life-cycle of a multicast session - an event diagram view

and Core/Tree Migration can be invoked when the quality of the tree degrades due to membership dynamics or when node/core and/or link failures occur.

The rest of the paper is organized as follows. Section II outlines the issues in core-based multicasting. Section III highlights the importance of tree migration and proposes algorithms and protocols for tree migration. Section IV presents the performance of the tree migration algorithms obtained through simulation studies. Finally, Section V concludes the paper.

II. ISSUES IN CORE-BASED MULTICASTING

The main issues associated with core-based multicasting are shown in Figure 2. Among these, the unique issues are discussed below.

- **Core Selection:** In core-based multicasting, since the tree is rooted at the core node, core selection is an important problem

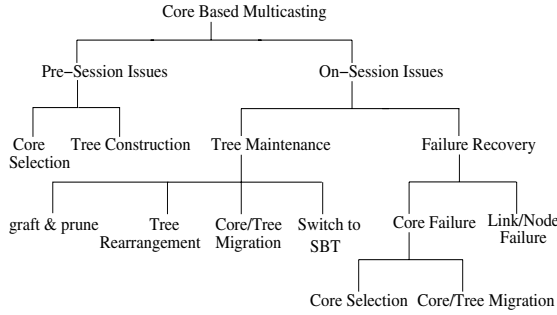


Fig. 2. Issues in Core Based Multicasting

because location of the core influences the tree structure (and cost) which in turn determines the delay experienced by individual receivers. The optimal core selection is an NP-complete problem and usually requires complete network topology and exact group membership details.

- *Core Failure Recovery*: In core-based multicasting, core is a single point of failure. If the core fails, there is a large amount of *Service disruption* as many of the receivers cannot receive the data sent by the senders. Therefore, recovery from both core failure and node/link failure [6] are important issues.
- *Core Degeneration & Core Migration*: If the group is highly dynamic, then the core ‘degenerates’ with time [7]. That is, for dynamic groups, the quality² of the core and hence the multicast tree will deteriorate with time. This means that the core migration has to be invoked when the quality of the core degenerates.

III. TREE MIGRATION

Core migration involves the following: (i) selecting a new core that can offer better performance than the current core, (ii) constructing a multicast tree based on the new core, and (iii) migrating the group members from the current multicast tree to the new multicast tree (tree migration). Tree migration can also be invoked as part of the core failure recovery when the current core fails. Moreover, partial/full tree migration is employed as part of the tree rearrangement algorithms [11] which usually monitors a certain damage/quality index to the multicast tree as members join/leave, and trigger tree rearrangement when the index exceeds/falls a certain threshold. Thus, tree migration plays a vital role in tree maintenance and core failure recovery (see Figure 2).

A number of heuristics have been proposed in [8] for core selection and are evaluated in [9]. Several algorithms have been proposed for multicast tree construction [10] (and the references in [1]) and tree maintenance [11]. Despite its importance, to the best of our knowledge, the tree migration problem is not known in the literature. In [12], Carlberg described a very simple core migration strategy where new core and the old core reside in the same tree. This strategy does not result in tree migration. But, it is to be noted that such a simple strategy is not always viable because new core need not always be the part of the existing tree if the group is highly dynamic. Also, core migration may be initiated because of fault in the existing tree. In that case, new core has to be shifted outside the existing tree

²Quality of the multicast tree is measured in terms of its cost and the delay experienced by individual receivers.

and then tree migration becomes inevitable. Therefore, tree migration is a general strategy to optimize the cost of the tree than the simple strategy described in [12].

In this paper, we first state the tree migration problem and prove that it is NP-Complete and then propose four heuristic algorithms for the problem. The heuristic algorithms are scalable and aim at minimizing either or both of the following performance metrics.

A. Tree Migration Problem and Performance Metrics

In order to precisely state the problem of tree migration we define two performance metrics - service disruption and resource wastage.

- *Service Disruption (SD)* is defined as the amount of packet loss experienced by the receivers during the process of tree migration.
- *Resource Wastage (RW)* is defined as the amount of excess resource (bandwidth) that remain idle multiplied by the duration for which it is idle during the process of tree migration.

$$RW = \sum_{i=0}^{R-1} (t'_{join_i} - t_{leave_i}) \times BW(path(C_o, r_i)),$$

where R is the number of receivers, t'_{join_i} is the time at which receiver r_i joins the new tree, t_{leave_i} is the time at which r_i leaves the old tree, and $BW(path(C_o, r_i))$ refers to the sum of bandwidth used for the multicast session along the path from old core to r_i in the old tree.

Tree Migration Problem: The problem is to migrate the receivers from one multicast tree to another with the goal of minimizing Service Disruption and Resource Wastage.

Lemma: Tree Migration problem is NP-Complete.

Proof: It is well known that the optimization problems that have two *path constraints* are NP-Complete [1]. For example, delay-constrained least-cost routing problem is NP-Complete as delay and cost are path (additive) metrics. The Tree Migration Problem has two path constraints: (i) Service Disruption, which is cumulative of packet loss experienced by each receiver; (ii) Resource Wastage, which is cumulative extra bandwidth temporarily reserved along the path from the source to receivers. Therefore, Tree Migration Problem is an optimization problem with two path constraints, and hence is NP-Complete.

B. Heuristic Algorithms for Tree Migration

We propose the following four heuristic algorithms for tree migration.

- *Add First Delete Last (AFDL)*: In this algorithm, the new tree is created first (i.e., all the receivers join the new tree) and then the old tree is deleted (i.e., all the receivers leave the old tree). This algorithm offers less service disruption, but incurs a high resource wastage.
- *Delete First Add Last (DFAL)*: This algorithm is the complement of the AFDL algorithm wherein the old tree is deleted first and then the new tree is created. This algorithm incurs no resource wastage, but incurs a high service disruption.
- *Interleaved Add Delete (IAD)*: This algorithm adopts AFDL on a receiver basis. Each receiver first joins the new tree and

then immediately leaves the old tree without waiting for the whole new tree getting created.

- *Interleaved Delete Add (IDA)*: This algorithm adopts DFAL on a receiver basis. Each receiver first leaves the old tree and then immediately joins the new tree without waiting for the whole old tree getting deleted.

The advantage of AFDL is that it incurs a low service disruption (packet loss) as the old tree is deleted after constructing the new tree. The disadvantage of AFDL is that it incurs a high resource wastage as both the trees consume resources simultaneously for some duration of time. It is to be noted that the service disruption incurred by AFDL does not necessarily be zero because a packet sent along the old tree may find that the resources on a link released and hence the packet may be dropped. The performance of DFAL is exactly the opposite of AFDL.

The difference between IAD and IDA bears resemblance to the difference between the AFDL and the DFAL algorithms, on a per-receiver basis. Thus, the service disruption of the IAD algorithm is less than that of the IDA algorithm, and the resource wastage incurred by IAD algorithm is more than that of the IDA algorithm. The performance of both the interleaved algorithms are bounded by the AFDL and the DFAL algorithms.

C. Tree Migration Protocols

The protocols that implement the proposed tree migration algorithms are shown in Figures 3(a)-(d).

IAD Protocol: Figure 3(a) shows this protocol. Here, the old core first sends the “migrate” message to all the receivers. Upon receiving this message, each receiver joins the new tree by sending a “join” message to the new core. The new core responds to the “join” message by sending “reserve” message to the receiver indicating reserving of resources along the path from new core to the receiver. Once the “reserve” message reaches the receiver, the receiver sends “leave” message to the old core. The old core responds to this message by sending a “release” message to the receiver indicating the release of resources along the path from old core to the receiver in the old tree.

IDA Protocol: Figure 3(b) shows this protocol. Here, the old core first sends the “migrate” message to all the receivers. Upon receiving this message, each receiver leaves the old tree by sending a “leave” message to the old core. The old core responds to the “leave” message by sending “release” message to the receiver indicating releasing of resources along the path from old core to the receiver in the old tree. Once the “release” message reaches the receiver, the receiver sends “join” message to the new core. The new core responds to this message by sending a “reserve” message to the receiver indicating reserving of resources along the path from new core to the receiver.

AFDL Protocol: Figure 3(c) shows this protocol. This is similar to IAD protocol except that the receivers expect a “tree_created” message from the new core indicating that the new tree has been created spanning all the receivers. This message is sent by the new core only when all the receivers have joined (attached to) the new tree. This means that the new core should know the membership of the multicast session.

DFAL Protocol: Figure 3(d) shows this protocol. This is similar to IDA protocol except that the receivers expect a “tree_deleted” message from the old core indicating that the old tree has been deleted. This message is sent by the old core only

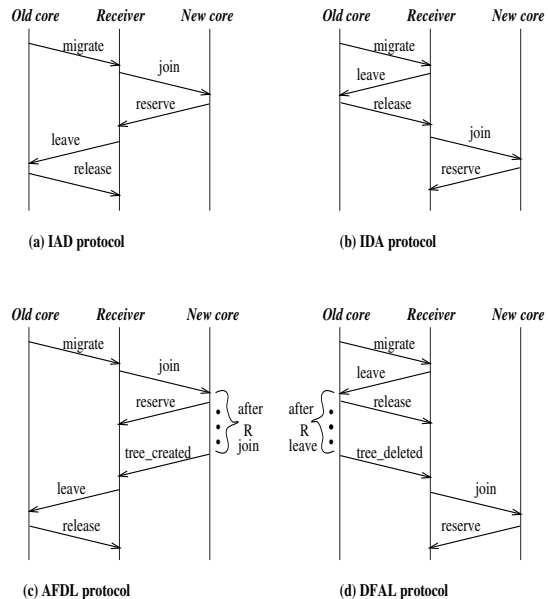


Fig. 3. Proposed tree migration protocols

when all the receivers have left (detached from) the old tree. This means that the old core should know the membership of the multicast session.

Among the four protocols, the interleaved protocols (IAD and IDA) are highly scalable (with group size and network size) and easily implementable as they do not require the core nodes (old core and new core) have the knowledge of group membership and network topology. Whereas, the other two protocols (AFDL and DFAL) suffer due to these requirements.

IV. SIMULATION STUDIES

We have conducted extensive simulation studies, using NS [13], to evaluate the effectiveness of the proposed tree migration algorithms. For our experiments, the various inputs were generated as follows.

- Random network topologies were generated based on a given input parameter “graph density”. This parameter determines the degree of the nodes and hence the connectivity of the network. Higher its value, the denser the topology is.
- The selections of source and receivers for a given multicast session are uniformly distributed from the node set.
- For each simulation point, approximately 500 core migrations were triggered.
- The initial and subsequent selection of core node for a given multicast session is uniformly distributed from the group members (i.e., source and receivers).
- The multicast tree based on the new core was constructed using the following two different approaches.

1. *Shortest Path Tree (SPT) Approach*: Combining the shortest (unicast) paths between the core and each receiver. The multicast tree thus created may not be cost³ optimal, but is amenable for distributed implementation. Also, it is highly scalable.

2. *Centralized Steiner Heuristic (CSH) Approach*: Using a centralized algorithm that exploits the membership and topology information. We have used KMB heuristic [10] for this

³Cost of the multicast tree is sum of cost of the tree edges.

purpose, whose cost ratio bound is 2. Though this approach has the potential to offer a multicast tree that has better (smaller) cost than the previous approach, its distributed implementation is difficult and hence not scalable.

The experiments were conducted in two parts: (i) evaluation of tree migration algorithms measuring resource wastage and packet loss incurred by them - the tree migration algorithms are independent of the multicast tree construction approach used - and (ii) evaluation of multicast tree construction approaches measuring the cost of the multicast tree produced by them. For both the parts, parameters such as number of receivers, graph density, and number of nodes in the network were varied. In our studies, the values of these parameters were taken to be 10, 3, and 31, respectively, when these parameters were not varied.

A. Evaluation of Tree Migration Algorithms

A.1 Effect of Number of Receivers

The total packet loss is plotted in Figure 4(a) for varying number of receivers. It is obvious to see that the total packet loss incurred by the algorithms increases with increasing number of receivers. Among the four algorithms, it can be seen that the packet loss experienced by AFDL is the lowest and DFAL is the highest confirming the very nature of their designs. The interleaved versions of these algorithms lie in between the performances of these extremes with IAD being closer to (in fact overlaps with) the AFDL, and IDA being closer to the DFAL algorithm.

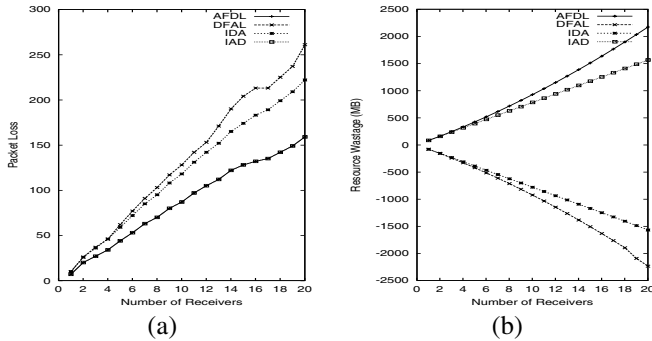


Fig. 4. Effect of number of receivers on (a) packet loss and (b) bandwidth wastage

In Figure 4(b), the total resource wastage is plotted for varying number of receivers. Also, it is obvious to see that the total resource wastage incurred by the algorithms increases with increasing number of receivers. The resource wastage incurred by AFDL is the highest because it constructs the new tree before deleting the old tree, thus resulting in full/portion of both old and new trees consuming resources during the tree migration process. Algorithm IAD incurs the next highest resource wastage as it is the interleaved version of AFDL. Both DFAL and IDA do not incur any resource wastage because the receivers are first deleted and then added. The graph shows negative resource wastage for DFAL and IDA which means that the receivers have left (i.e., not part of) the old group earlier than they joined the new group.

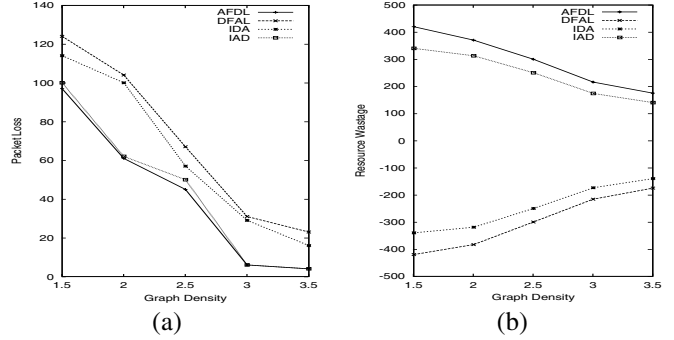


Fig. 5. Effect of network connectivity on (a) packet loss and (b) bandwidth wastage

A.2 Effect of Network Connectivity

The effect of graph density on packet loss for different tree migration algorithms is shown in Figure 5(a). As the graph density (average node degree) increases, the packet loss decreases almost linearly for all the four algorithms. This is because as the topology becomes denser, the chances of finding shortest paths to the receivers become higher. As a result, the packet loss decreases with increasing graph density. The resource wastage incurred by the algorithms decreases with increasing graph density for the same reason (shown in Figure 5(b)). The relative performances of the algorithms for both the metrics are similar to that in the previous study.

A.3 Effect of Number of Nodes

Figures 6(a) and 6(b) show the effect of varying number of nodes in the network on packet loss and resource wastage, respectively. Both Packet Loss and Resource Wastage increase with increasing number of nodes. The reason for this behavior is attributed to the sparse nature of the groups as the number of nodes increases. That is, for a given group size and graph density, the group becomes sparser when the number of nodes in the network increases because the number of hops separating the members increases. When the groups become sparser, the resultant trees have more links and hence more resource wastage and packet loss. The relative performances of the algorithms are the same as in sections A1 and A2.

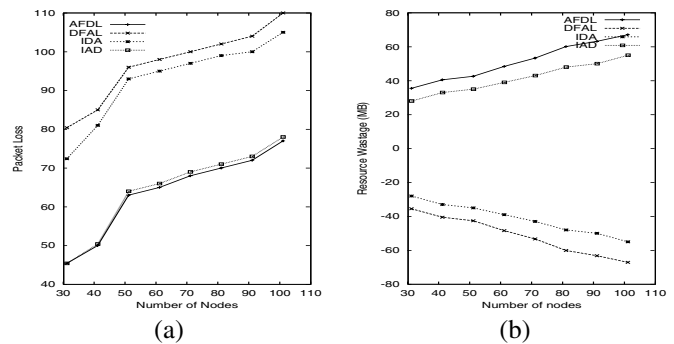


Fig. 6. Effect of number of nodes on (a) packet loss and (b) bandwidth wastage

B. Evaluation of Multicast Tree Construction Approaches

Here, we present the studies that were carried out to evaluate the cost of the multicast tree produced by different multicast routing approaches. Let N be the number of receivers in the

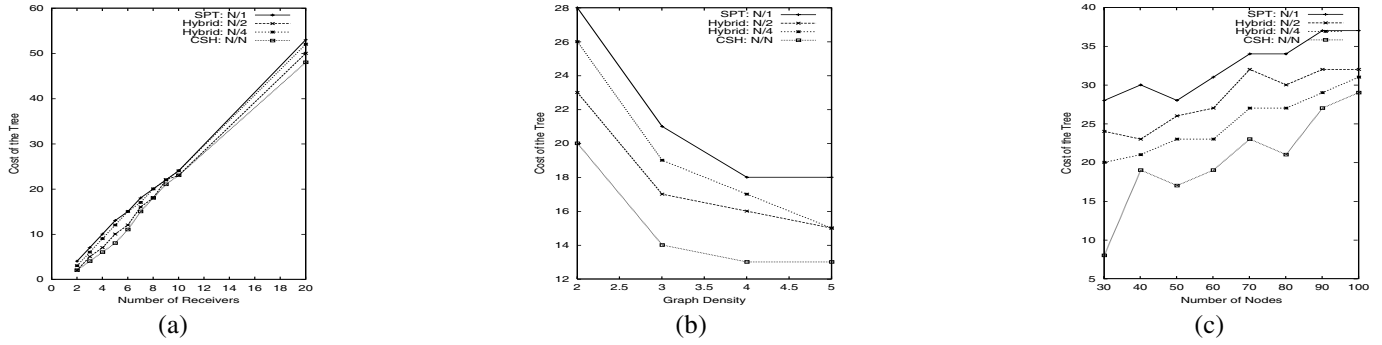


Fig. 7. Effect of (a) number of receivers, (b) network connectivity and (c) number of nodes on tree cost

group. The receivers are divided into two partitions, one with K receivers and the other with $N - K$ receivers. In our study, the multicast tree was constructed using CSH approach spanning K receivers and the remaining $N - K$ receivers were grafted (join) to this tree, using SPT approach. When $K = N$, the algorithm reduces to CSH approach on N receivers; when $K = 1$, it reduces to SPT approach. The value of K captures the tradeoff between the practicality of the approach and the cost of the tree offered by the approach. As mentioned earlier, $K = N$ (i.e., CSH) is the best in terms of tree cost and worst in terms of practicality, whereas $K = 1$ (i.e., the SPT) is the other way. In our simulation studies, we considered the following cases: $K = 1$, $K = N/4$, $K = N/2$, and $K = N$. The simulation results presented here are aimed at quantifying this tradeoff between the approaches.

In Figure 7(a), the cost of the multicast tree is plotted for four different values of K by varying the number of receivers. As the number of receivers increases, it is obvious to see that the cost of the tree also increases almost linearly. The figure also shows that for the CSH approach (referred to as N/1 in the figure), the cost of the multicast tree is the minimum. The cost of the tree offered by SPT approach (referred to as N/N in the figure) is the maximum. The cost of the tree in case of the other two algorithms $K = N/2$ and $K = N/4$ lie somewhere in between these two extreme cases. The interesting point to note here is that the cost of the tree offered by the SPT approach is very close to that of the CSH approach.

In Figure 7(b), the cost of the tree for varying graph density is plotted. When the graph density is increased, the cost of the tree decreases for most cases. The reason being the possibility of finding better shortest paths in denser networks and hence better cost trees as discussed before. Here too, the cost of the tree offered by the SPT approach is close to that of the CSH approach for the simulated conditions.

In Figure 7(c), the cost of the tree for varying number of nodes is plotted. The cost the tree increases with increasing number of nodes because of the sparse nature of the groups as discussed in section A3. The relative tree costs among different cases of the algorithms exhibit similar behavior as figures 7(a) and 7(b).

V. CONCLUSIONS

In this paper, we highlighted the importance of tree migration as a mechanism for handling membership and network dynamics in multicasting and proposed heuristic algorithms and protocols for it. The proposed algorithms were evaluated un-

der two performance metrics: service disruption and resource wastage. Our simulation studies show that the algorithms IAD and IDA offer performance comparable to that of AFDL and DFAL, in addition to being highly scalable and easily implementable. Our tradeoff studies on multicast tree construction approach have shown that the distributed SPT approach offers tree costs that are comparable (in most cases) to that of the centralized CSH approach, in addition to being highly scalable and easily implementable. The choice of the right combination of multicast tree construction approach and tree migration algorithm depends on various performance goals and varies with applications. In particular, the SPT based multicast tree construction with IAD/IDA tree migration algorithm is very attractive in terms of scalability and implementation complexity. Currently, we are developing an integrated tree maintenance technique which encompasses tree rearrangement and tree migration.

REFERENCES

- [1] J. Hou and B. Wang, "Multicast routing and its QoS extension: Problems, algorithms and Protocols," *IEEE Networks*, Jan./Feb. 2000.
- [2] S. Deering, C. Partridge, and D. Waitzman, "Distance vector multicast routing protocol," *RFC 1075*, Nov. 1988.
- [3] T. Ballardie, P. Francis, and J. Crowcroft, "Core-based trees (CBT): An architecture for scalable inter-domain multicast routing," in *Proc. ACM SIGCOMM*, pp.85-95, 1993.
- [4] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. Liu, and L. Wei, "The PIM architecture for wide-area multicast routing," *IEEE/ACM Trans. Networking*, vol.4, no.2, pp.153-162, Apr. 1996.
- [5] M. Parsa and J.J. Garcia-Luna-Aceves, "A protocol for scalable loop-free multicasting," *IEEE JSAC*, vol.15, no.3, pp.316-331, April 1997.
- [6] L. Schwiebert and R. Chintalapati, "Improved fault recovery for core based trees," *Computer Communications*, vol.23, no.9, Apr. 2000.
- [7] C. Donahoo and Zegura, "Core migration for dynamic multicast routing," in *Proc. ICCCN*, 1995.
- [8] D.G. Thaler and C.V. Ravishanker, "Distributed center location algorithms," *IEEE JSAC*, vol.15, no.3, pp.291-303, Apr. 1997.
- [9] E. Fleury, Y. Huang, P.K. McKinley, "On the performance and feasibility of multicast core selection heuristics," in *Proc. ICCCN*, pp.296-303, 1998.
- [10] L. Kou, G. Markowsky, and L. Berman, "A fast algorithm for Steiner trees," *Acta Informatica*, vol.15, no.2, pp.141-145, 1981.
- [11] R. Sriram, G. Manimaran, and C. Siva Ram Murthy, "A rearrangeable algorithm for the construction of delay-constrained dynamic multicast trees," *IEEE/ACM Trans. Networking*, vol.7, no.4, pp.514-529, Aug. 1999.
- [12] K. Carlberg, "QoS Multicast using single metric Unicast Routing", PhD thesis, University College London, Oct. 1999.
- [13] UCB/LBNL/VINT Network Simulator - ns (version 2), Available at www.mash.cs.berkeley.edu/ns/.