New Center Location Algorithms for Shared Multicast Trees

Young-Chul Shim[1] and Shin-Kyu Kang[2]

[1] Hongik University, Department of Computer Engineering, Seoul, Korea
shim@cs.hongik.ac.kr
[2] Aston Linux, Seoul, Korea
cosmos@astonlinux.com

Abstract. Multicast routing algorithms such as PIM, CBT, BGMP use shared multicast routing trees and the location of the multicast tree has great impact on the tree cost and the packet delay. In this paper we propose new center location algorithms and a new center relocation algorithm and analyze their performance through simulation studies. The proposed center location algorithms try to find the geographic center of multicast members considering not only multicast group members but also a few non-member nodes which are carefully chosen. Simulation results show that the proposed algorithms find the better center than existing algorithms in terms of tree cost and packet delay. After many members have joined and/or left the group, the previously chosen center may not be a proper place any more and, therefore, we need to find a new center and build a new tree around this new center. We propose a new center relocation algorithm that determines the moment when the new tree should be built around the new center. The algorithm is based on measured packet delays as well as the parameter indicating how much the group has changed. It not only avoids unnecessary center relocation processes but also prevents the cost and worst packet delay of the tree from significantly deviating from the optimal values. ...

1 Introduction

Multicast is an efficient mechanism for sending packets to a group of receivers and used in many areas[1,2]. To send packets to multicast group members, a multicast routing algorithm builds multicast packet delivery trees among senders and receivers. There are two types of delivery trees: source based trees and shared trees. In the source based tree approach, a shortest path tree is built from a sender to all the receivers and one tree is built for each sender. DVMRP[3] and MOSPF are examples of routing algorithms building source based trees. The disadvantage of this approach is that there are as many trees as the senders and the management of these trees can be very complicated. To solve this problem one shared tree is built among all senders and receivers in the shared tree approach.

This work was supported by ITRC

E. Gregori et al. (Eds.): NETWORKING 2002, LNCS 2345, pp. 1045–1056, 2002.
© Springer-Verlag Berlin Heidelberg 2002

CBT(Core Based Trees)[5], PIM-SM[6], and BGMP[7] are routing algorithms in this category. In this approach the location of the center of the shared tree greatly affects the multicast tree cost and the packet transmission delay over the tree and, therefore, the determination of the proper location of the center becomes an important issue. In a dynamic environment where members can join and leave during a multicast session, the center location which may have been optimal in the beginning may not be so anymore after many membership changes. So in case of the dynamic environment, the relocation of the center also becomes an important issue.

The center location algorithms can be divided into three categories depending upon what network nodes are considered as candidates for the center. In the first category, all the network nodes can become candidates for the center and the best node is chosen as the center. With this method, the optimal center location can be found but because too many packets are exchanged among all the network nodes, it is never a practical solution. In the second category, only the multicast members are considered as candidates. This approach incurs the least overhead but because only the members are considered, the chosen center location can be far from being optimal. The last category stands between the first and the second. In this category, not only the members but also some carefully chosen non-member nodes are considered for the center and the best node among them is chosen as the center. The method of choosing non-member nodes that will be considered as candidates affects the overhead of the center location algorithm and the quality of the chosen center.

We propose a new center location algorithm called GeoCenter(Geographic Center). The idea is that we try to find the geographic center of the multicast members in the Internet map and this geographic center becomes the center of the multicast tree. This geographic center can become the member or non-member router. We introduce three algorithms GeoCenter1, GeoCenter2, and GeoCenter3 depending upon the method of finding the geographic center. The proposed algorithms try to minimize the packet delay and the tree cost. Then we consider a dynamic case and propose a new algorithm for relocating the center as the membership changes. The center relocation process is such a costly one that its execution should be limited only to unavoidable cases. The new algorithm determines the moment when the new tree should be built around the new center. This algorithm is based on measured packet delays as well as the parameter indicating how much the group has changed. It not only avoids unnecessary center relocation processes but also prevents the cost and worst packet delay of tree from deviating too much from the optimal value. We analyze the performance of the center location and relocation algorithms through simulation.

The rest of the paper is organized as follows. Section 2 surveys related work. Sections 3 and 4 describe our algorithms for center location and relocation, respectively. Section 5 presents simulation results and is followed by the conclusion in Section 6.

2 Related Work

In this section we first present algorithms that have been proposed for the center location. Before introducing these algorithms we give the definition of the tree cost and explain weight functions that have been used in those algorithms. The tree cost is the sum of the cost of each link in the tree. The link cost can be the actual monetary value of that link, bandwidth, delay, etc. But in this paper we set the link cost to be 1 for every link. A weight function is calculated for each center candidate and the resulting values are compared to select the best one. We introduce the definitions of some weight functions in the following[8]. In the definitions, S is the set of all the senders and members, u and v represent either a sender or a member, root is the candidate for the center, d(u,v) is the distance between u and v and deg(u) is the degree of u.

$$Actual\ Cost = number\ of\ links\ in\ tree\ rooted\ at\ root\ .$$

$$Max\ Dist = \max_{u \in S} d(root, u)\ .$$

$$Avg\ Dist = \frac{1}{|S|} \sum_{u \in S} d(root, u)\ .$$

$$Max\ Diam = \max_{u \in S} d(root, u) + \max_{v \in S, v \neq u} d(root, v)\ .$$

$$Est\ Cost = \frac{Est\ Cost_{min} + Est\ Cost_{max}}{2}\ .$$

$$where \quad Est\ Cost_{min} = \max_{u \in S} d(root, u) + number\ of\ duplicate\ distance\ nodes\ in\ S$$

$$Est\ Cost_{max} = \begin{cases} \sum_{u \in S} d(root, u) & if\ |S| \leq deg(root) \\ [\sum_{u \in S} d(root, u)] - [|S| - deg(root)] & otherwise \end{cases}$$

Now we introduce several center location algorithms. The OCBT(Optimal Center-Based Tree) algorithm calculates the actual cost of the tree rooted at each node in the network and selects the one which gives the lowest maximum delay over all the roots with the lowest cost. The MCT(Maximum-Centered Tree) algorithm selects the node with the smallest $MaxDist$ value. The ACT(Average-Centered Tree) algorithm chooses the node with the smallest $Avg\ Dist$ value. The DCT(Diameter-Centered Tree) algorithm selects the node with the lowest $Max\ Diam$ value. These four algorithms belong to the first category of center location algorithms.

The RSST(Random Source-Specific Tree) algorithm chooses the center randomly among the senders and is used in CBT and PIM. In the MIN-MEM(Minimal Member Tree) algorithm, each member or sender node calculates the weight function of the multicast tree rooted at itself and exchanges the calculated value with other nodes. The node with the lowest weight function

value becomes the center. The weight function can be any of the five functions explained above. These two algorithms belong to the second category.

In the HILLCLIMB algorithm, a randomly chosen temporary center calculates the weight function of the tree rooted at itself and all the routers directly connected to the temporary center do the same calculation. If the temporary center has the lowest value, it becomes the center. Otherwise the node with the lowest value becomes the temporary center and compares the weight function value with its direct neighbors. This process is continued until the node is found such that its value is lower than those of its direct neighbors or the distance from the original temporary center to the current temporary center reaches a certain threshold. This algorithm belongs to the third category. The problem of this algorithm is that it just finds the locally optimal point among nodes within a limited distance from the original center.

In a dynamic environment, a center can be relocated by applying any of the above algorithms after some membership changes have occurred. The biggest issue here is when to apply the center location algorithm again. Thaler and Ravishankar introduce the parameter Δ defined as follows[9]:

$$\Delta = 1 - \frac{|G_0 \cap G_i|}{max(|G_0|, |G_i|)} \ .$$

where G_0 is the original group membership, G_i is the current group membership, and Δ indicates the amount by which the group has changed. They propose to recalculate the center location when Δ reaches 90%. They show that when 90% of the membership has changed, the tree cost has likewise degraded about 90% of the way toward a randomly centered tree. But we show that their algorithm does not improve the quality of the tree center at all in some cases and, therefore, incurs unnecessary center relocation processes.

3 New Algorithms for Center Location

In this section we introduce 3 new center location algorithms: GeoCenter1, GeoCenter2, and GeoCenter3. These algorithms pick the center based upon the information on routes between members. The route information from a node A to a node B is the list of all the routers visited on the path from A to B and this information can be obtained by using the program called traceroute or the IP route record option. In the GeoCenter1 algorithm, each member finds the route information to all the other member nodes, compiles all the routers appearing in the routes, and sends this information to a temporary center. Upon receiving the route information from all the members, the temporary center finds the routers that appear most frequently in the route information. If there are several such routers, one router is selected randomly. This selected router and the member nodes become the candidates for the center. The center is chosen as the node which has the lowest $Max\ Dist$ value among these candidates.

GeoCenter2 and GeoCenter3 take different approaches in selecting non-member candidates. In the GeoCenter2 algorithm, each member first collects

route information from all the other router as in GeoCenter1 but, when compiling this information, records not only the addresses of each router in the route information but also the number of times a router appears in the route information. This information is sent to the temporary center. In the GeoCenter3 algorithm, each member finds the midpoints on the path to other members and sends the list of these midpoints to the temporary center. The way the temporary center selects the center is the same as in GeoCenter1.

Based upon the explanation given in the above, we now present each algorithm in detail.

GeoCenter1

① When a multicast group is created, a temporary center is chosen arbitrarily.
② The temporary center sends a probe message to each member. The probe message also contains the list of the multicast group members.
③ Upon receiving the probe message, each member finds the route information to all the other members using either the traceroute program or the record route IP option. At the same time the member measures the packet delay to other members and records the largest packet delay as its weight function value.
④ Each member compiles the list of nodes appearing on the path to other members from itself. It sends this list and its weight function value to the center.
⑤ Upon receiving the list from all the members, the temporary center selects a node(s) that appears most frequently in the lists. If there are many such nodes, one or more nodes are randomly picked. If the picked node(s) are a member, go to step ⑧.
⑥ The temporary center sends a probe message to the selected non-member candidate(s).
⑦ The non-member candidate measures the packet delay to all the members, records the largest delay as its weight function value, and sends this value to the temporary center.
⑧ The temporary center selects the node which has the lowest weight function value as the center.

GeoCenter2

GeoCenter2 is the same as GeoCenter1 except the step ④. The member nodes send not only the address of nodes on the path to the other member nodes but also the visit counts of such nodes.

GeoCenter3

①②③ The same as in GeoCenter1.
④ Each member finds the midpoints on the path to other members and sends the list of midpoints and its weight function value to the temporary center.
⑤ The temporary center adds up all the visit counts for each node appearing in the lists received from the member nodes. It picks the node(s) with the largest accumulated visit counts. If the picked node(s) is a member, go to step ⑧.
⑥⑦⑧ The same as in GeoCenter1.

4 The Center Relocation Algorithm in a Dynamic Multicast Environment

In this section we explain the algorithm for relocating the center after membership changes have occurred many times. After many members have joined and/or left the group, the center that was carefully chosen in the previous time may not be a good place any more. The quality of the tree may have deteriorated during the membership changes and, therefore, the tree cost and the maximum delay may have become too high compared with the optimal tree. As already explained in the previous section, the most important issue in the center relocation is the determination of the moment when the center location algorithm is applied. Because the center relocation process requires not only the determination of a new center location but also building a new multicast delivery tree around this new center, it is a very costly process. In reality, building a new tree will consume more time than calculating a new center location. So it is imperative to minimize the numbers that the multicast delivery tree is rebuilt.

As we described in Section 2, Thaler and Ravishankar introduce a parameter Δ indicating the amount by which the group has changed and used this parameter to determine when to calculate the new center location. They propose to calculate the new center location when Δ reaches 90%. They also show that the time interval which it takes for Δ to reach from 0 to 90% roughly corresponds to two to three times of the average connection duration of a member in a multicast group. But as we will show with simulation, if the area in which members are located does not change very much and members are uniformly distributed in this fixed area, the center relocation using only Δ does not improve the tree cost and the packet delay and, is unnecessary in many cases.

Another parameter we can use to determine when to calculate the new center location is the maximum delay from the center to the member nodes. When the center location is calculated, the maximum delay at that moment is recorded as Prev_Max_Delay. When a new member joins the group, the delay from the current center to this node is calculated and compared with Prev_Max_Delay. If the delay to this new node exceeds a certain constant times of Prev_Max_Delay, a new center location is calculated. Using this method, the unnecessary recalculation of the center in the case of just using the parameter Δ can be avoided because the center will be recalculated only if the proof that the quality of the tree has deteriorated enough is obtained. This method of measuring the delay to a joining member and comparing against the Prev_Max_Delay works if the area where members are located remains the same, moves, or gets larger. But this method does not work if the member distribution area gets smaller. If the size of the area gets smaller, the delay to a new member will rarely exceed the previously measured Prev_Max_Delay value. But the quality of the tree may have been deteriorated compared with the optimal tree.

We propose a new algorithm for determining when to recalculate the center location and when to actually rebuild the tree around the new center. The proposed algorithm uses both the parameter Δ and the delay to a new member. We assume that the multicast routing algorithm enables the center to be noti-

fied of all the join events of new members and calculate the delay to these new members. Now we explain our algorithm in detail as follow.

① The location of a center is calculated.
② The multicast delivery tree is built around the calculated center. The Prev_Max_Delay and Curr_Max_Delay values are initialized to be the value of the maximum delay of this new tree. Set G_0 and G_i to be the current set of members.
③ Wait until a membership changes. If the membership change is a join event, calculate the delay from the current center to this new member. Update the Curr_Max_Delay to be the maximum of the delay to this new node and the current value for Curr_Max_Delay. If Curr_Max_Delay is greater than C1 * Prev_Max_Delay, go to step ①.
④ Update G_i and the value of Δ. If this value does not exceed C2, go to step ③. Otherwise calculate the location of a new center. Assuming this new center, calculate the maximum delay of the new tree and set this value to be Opt_Max_Delay. If Curr_Max_Delay is greater than C1 * Opt_Max_Delay, go to step ②. Otherwise, set G_0 and G_i to be the set of current members and go to step ③.

In the above algorithm C1 determines the extent to which the worst case packet delay of the current tree is permitted to exceed the worst case packet delay which was calculated when the center was determined in the most recent time. If C1 is 1.4, the excess up to 40% is permitted. If Δ reaches C2, it means that C2 * 100% of members have changed. In the algorithm given in [9], C2 was set to be 0.9. The algorithm calculates a new center location and builds a new tree around this new center if the delay to a new joining member is bad enough compared with the maximum delay which was calculated when the tree was built in the last time. But the area where the members are distributed gets smaller, this condition will be rarely satisfied. So after we have seen enough changes in the membership, we calculate the new center location and also calculate the new tree around this new center. If we see that the quality of the current tree is bad enough compared with this new tree, we actually rebuild the tree around the calculated new center .

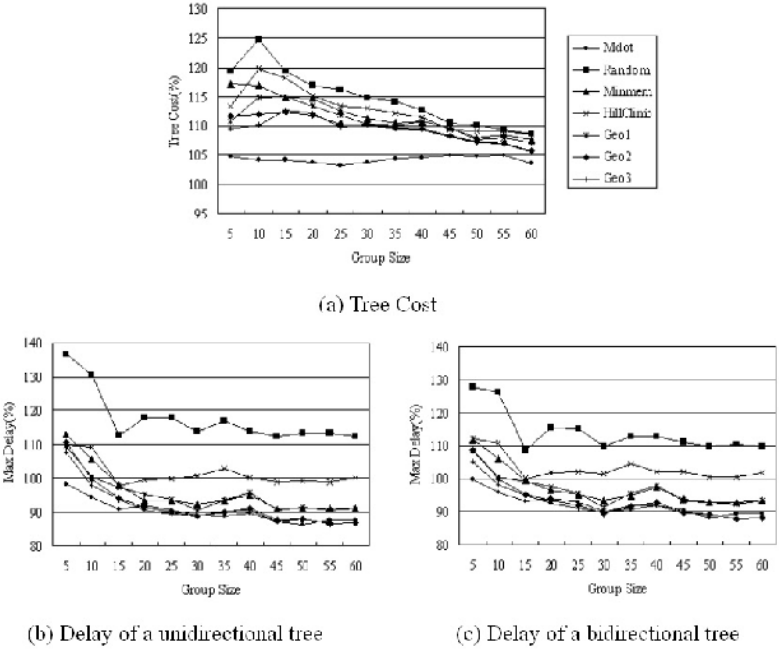
5 Experimentation Results

In this section we present and analyze simulation results for our center location and relocation algorithms. We first present simulation results for our center location algorithms assuming that the set of members and senders are fixed and compare their performance with other center location algorithms. Then we show simulation results for our center relocation algorithm in the environment where members join and leave.

For the experimentation we used NS(Network Simulator) developed in UC Berkeley and ran the simulation on Linux 5.1 platforms. Algorithms were implemented with TCL and network topologies were generated with the GT-ITM(Georgia Tech Internetwork Topology Models) provided in the NS.

5.1 Experimentation Results for Center Location Algorithms

In this subsection we compare the proposed center location algorithms with other algorithms through simulation changing the multicast group size, the network size, and the average number of links for a node in the network. We measure the tree cost and packet transmission delay of the multicast trees built using various center location algorithms. We compare the proposed algorithms with OCBT, MDOT(Minimum Delay Optimal Tree), Random, MIN-MEM, and HILLCLIMB algorithms. The OCBT algorithm is optimal in terms of the tree cost. The MDOT considers all the nodes as the candidates for the center and picks the node such that the tree built around this node has the lowest $MaxDist$ weight function value. If several nodes have the same lowest $Max\ Dist$ value, the node with the lowest tree cost is selected. This algorithm is optimal in terms of tree costs when the shared trees are unidirectional such as in PIM-SM but may not be optimal in case of bi-directional shared trees such as in CBT. The Random algorithm chooses the center randomly. These three algorithms, OCBT, MDOT, and Random, are not practical but considered here just for the comparison. MIN-MEM and HILLCLIMB algorithms are practical solutions and shown to find a good center[8,9]. For the experimentation we assume that all the senders are also members and for each simulation we perform 100 experiments and take the average as the result.



(a) Tree Cost

(b) Delay of a unidirectional tree

(c) Delay of a bidirectional tree

Fig. 1. Effects of group size on algorithms

A group size is the number of member nodes in a multicast group. In the first experimentation we assumed that there were 100 nodes in the network and the average number of links for nodes was 4. We changed the group size from 5 to 60 and measured the tree costs and the packet delays. Figure 1 (a) shows the tree costs of various algorithms and the tree cost is represented as the ratio to OCBT. The methods for measuring the packet delay become different depending on the type of shared trees: unidirectional trees or bidirectional trees. In unidirectional shared trees, packets are sent to the center and then distributed to all the members. But in bidirectional shared trees, packets are sent along the shortest path on the shared tree from the sender to members and in many cases may not pass the center. Figures 1 (b) and (c) show the packet delays of various algorithms and the packet delays are represented as the ratio to OCBT. In the figures some algorithms sometimes show better packet delay better than MDOT and this can be possible because MDOT is optimal when the packet delays are measured between the center and the receivers not between senders and receivers. The figures show that the tree cost of GeoCenter2 and GeoCenter3 stays within 112% of OCBT and is better than MIN-MEM and HILLCLIMB and the packet delay of GeoCenter2 and GeoCenter3 is comparable to MDOT and always lower than other algorithms. We see that GeoCenter2 and GeoCenter3 algorithms show better result than GeoCenter1 because former algorithms find better geographic centers than GeoCenter1.

Next we summarize the results for the second and third sets of simulations without showing the figures because they were similar to Figure 1. The second set of experiments was performed varying the network size that is the number of nodes in the simulated network. The average number of links for nodes was set to 4 and the group size was assumed to be 20% of the network size. The tree cost of GeoCenter2 and GeoCenter3 was 12% higher than OCBT in the worst case but always better than MIN-MEMB and HILLCLIMB. The packet delay of GeoCenter2 and GeoCenter3 was comparable with MDOT, even better than MDOT at some points, and constantly better than other algorithms.

The third set of experiments was performed varying the average number of links of a node in the network. The network size and the group size were assumed to 100 and 20, respectively. The simulation showed that the tree cost GeoCenter2 and GeoCenter3 stayed within 114% of OCBT and always better than MIN-MEMB and HILLCLIMB. The packet delay of GeoCenter2 and GeoCenter3 was very similar to MDOT and always better than other algorithms.

From the above three sets of experiments, we conclude that two of the proposed algorithms, GeoCenter2 and GeoCenter3, achieve near optimal packet delay compared with the MDOT algorithm while not incurring too much increase on the tree cost compared with the OCBT algorithm.

5.2 Experimentation Results for the Center Relocation Algorithm

In this subsection we consider a dynamic environment where members can join and leave during the lifetime of a multicast session.

Thaler and Ravishankar introduced the parameter Δ and proposed to calculate the new center and rebuild the tree as Δ reaches at some fixed value[9]. We first show that if the area in which the members are located is fixed and the members are uniformly distributed in this area, their simple method of just using the Δ value does not improve the quality of trees at all. We ran experiments with a network of 200 nodes. The members were uniformly distributed in this network and their average number was 40. We compared the quality of trees of two cases. In the first case the center location is never recalculated and in the second case the center location is recalculated when Δ reaches 90%. The simulation results showed that recentering and rebuilding a tree with just using Δ did not improve the quality of trees at all and in some cases gave worse performance. So we conclude that if members are uniformly distributed in a fixed area, we need not recalculate the center location. This is because the center that was calculated in the beginning remains to be near optimal if the distribution of the members remains uniform even though members join or leave the multicast group. From the experiment we can see that the packet delay remains to be within 140% of the optimal value, so if we set C1 to be 1.4 in our center relocation algorithm, the center need not be moved and, therefore, unnecessary overhead can be avoided. But the algorithm by Thaler and Ravishankar using only Δ regularly changes the center location but does not improve the quality of the multicast tree.

Now we consider three cases where the area in which members are distributed changes and show how our center relocation algorithm explained in the previous section performs. In the first case the area expands, in the second case the size of the area remains the same but the area moves gradually, and in the last case the size of the area becomes reduced.

Figure 2 shows the simulation results when the area expands. Each figure has two graphs. The first graph shows the result when the center is calculated once in the beginning and is never recalculated. The second graph describes the result when the center is recalculated by the proposed algorithm using both Δ and the worst case packet delay measurement. In the experiments GeoCenter3 algorithms was used to determine the center location. The points on graphs are represented

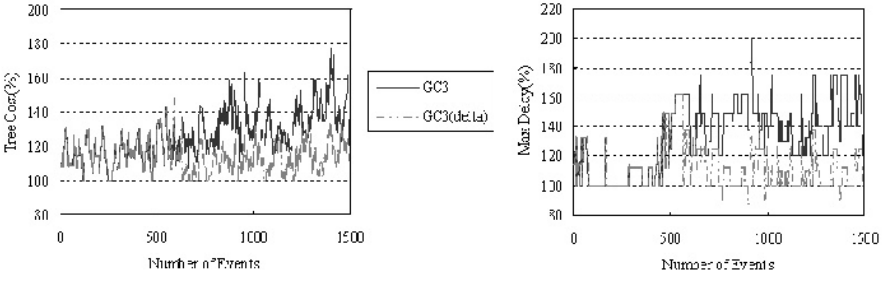


Fig. 2. Center relocation in an expanding area

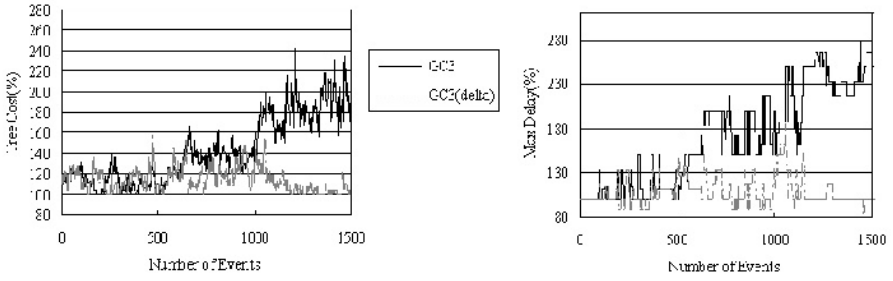


Fig. 3. Center relocation in a moving area

as the ratio to the value of optimal tree generated using the OCBT algorithm at each measurement point. The figure shows that the tree cost gradually increases without the center relocation algorithm as the area expands with the center recalculation. But if we use the proposed center relocation algorithm, the tree cost never becomes 30% higher than the optimal value calculated with the OCBT algorithm at each point. The figure also shows that the packet delay becomes almost 1.8-2 times of the OCBT tree without the center recalculation but rarely becomes 20% higher than the OCBT tree if we use the proposed center relocation algorithm.

Figure 3 shows the simulation results when the area is moving and Figure 4 shows the simulation results when the area gets reduced. They show the same result as in the case when the area gets expanded.

From these simulation results we see that the algorithm using just the Δ parameter regularly recalculates the center location and actually rebuilds the multicast tree without making much improvement on the tree quality in the case that the area in which members are distributed is fixed. We note that rebuilding a tree is a very costly process. But the proposed algorithm uses both the Δ parameter and the measurement data of the worst case packet delay and can avoid unnecessary rebuilding of the multicast tree in this case. In the cases where

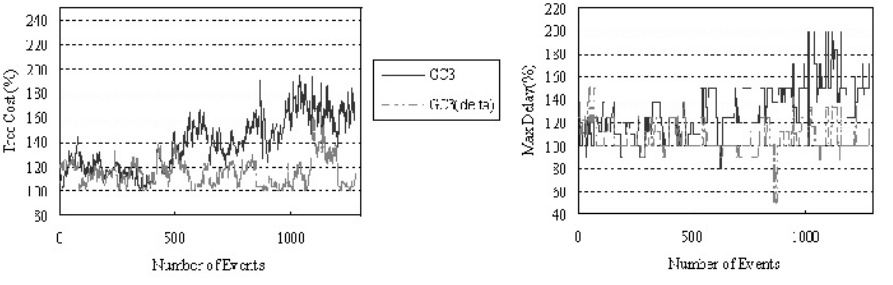


Fig. 4. Center relocation in an reducing area

the area gets expanded, moves, or becomes reduced, the proposed algorithm generates multicast trees of reasonable quality. And by properly adjusting the value of C1, which is the multiplier on the previously measured maximum packet delay and determines when the center should be recalculated, we can bound the maximum packet delay within a certain limit of the maximum delay of the OCBT.

6 Conclusion

In this paper we proposed new center location algorithms and a new center relocation algorithm for multicast routing alogrithms building shared trees and analyzed their performance through simulation.

The proposed center location algorithms try to find the geographic center of multicast members considering not only multicast group members but also a few non-member nodes that are carefully chosen. We built multicast trees around the centers found by our algorithms and we found that these trees had slightly higher tree cost than the cost-optimal tree, the similar packet delay as the delay-optimal tree, and constantly better cost and packet delay than the trees built around the centers found by algorithms proposed by other researchers.

Then we considered a dynamic environment where members could join and leave a multicast session and proposed a center relocation algorithm which determined the moment when the new tree should be built around the new center. The algorithm is based on measure packet delays as well as the parameter indicating how much the group membership has changed. Our algorithm not only avoids unnecessary center relocation processes but also prevents the cost and worst packet delay of the tree from deviating too much from the optimal values.

References

1. T.A. Maufer: Deploying IP Multicast in the Enterprise. Prentice Hall. (1997)
2. B. Quinn: IP Multicast Applications: Challenges and Solutions. Internet Draft drft-ietf-mboned-mcast-apps-01.txt. (June 1999)
3. D. Waitzman, C. Partridge, and S. Deering: Distance Vector Multicast Routing Protocol. RFC 1075. (1988)
4. J. Moy: MOSPF: Analysis and Experience. RFC 1585. (1994)
5. B. Cain, Z. Zhang, and A. Ballardi: Core Based Trees Multicast Routing: Protocol Specifcation. (1998)
6. D. Estrin et al: Protocol Independent Multicast-Sparse Mode: Protocol Specifiction. RFC 2362. (1998)
7. S. Kumar et al: The MASC/BGMP Architecture for Inter-Domain Multicast Routing. ACM SIGCOMM Conference. (August 1998)
8. D. Thaler and C. Ravishankar: Distributed Center-Location Algorithms: Proposals and Comparisons. IEEE Infocom. (1996)
9. D. Thaler and C. Ravishankar: Distributed Center-Location Algorithms. IEEE Journal on Selected Area in Communications, vol. 15, no. 3. (1997)