A Distributed Delay-Constrained Multicast Routing Algorithm Considering Identical Multiple Sources

Yung Yi, Youngseok Lee, Yongjun Im, and Yanghee Choi
Dept. of Computer Engineering
Seoul National University, Seoul, Korea.
{yiyung, yslee, yjim, yhchoi}@mmlab.snu.ac.kr

All correspondence to :
Yung Yi
Department of Computer Engineering
Kwanak-ku, Seoul, Korea 151-742
(tel) +82-2-876-7170 (fax) +82-2-876-7171
(e-mail) yiyung@mmlab.snu.ac.kr

A Distributed Delay-Constrained Multicast Routing Algorithm Considering Identical Multiple Sources *

Abstract

Multicast communication provides a mechanism by which data can be sent efficiently from a source to multiple destinations. Because the current multicast communication paradigm is based on one-to-many transmission, researchers have concentrated on finding a tree spanning from a source to multiple destinations. If the definition of source is extended to the redundant nodes of a group, however, we can also extend the multicast paradigm to include source group-based multicast communication. Application of the source group concept makes efficient multicast traffic distribution, optimizes multicast connections, and facilitates reliable and scalable service. Redundant sources can be applied to such services as anycasting ,directory services or multimedia data broadcasting. In this paper, we combine the redundant source group model and multicast communication services; then we propose a distributed source group-based multicast routing algorithm which is a modified version of the traditional multicast routing algorithm. By simulation we show that the proposed algorithm can make efficient use of the source group to reduce end-to-end delays and total connection costs satisfying constrained delay for continuous media traffic.

Keywords: *Multicast, Source Group, Virtual Top Source, Delay-Constrained, Distributed Algorithm*

1 Introduction

With the rapid progress of the network and computer technologies, it becomes possible to exchange large continuous media data over the network in realtime. Applications sitting on top of the multimedia and multipoint network, such as videoconferencing or CSCW(Computer Supported Cooperative Working), have been introduced and are penerating rapidly into the business and residential market. Those distributed realtime multimedia applications require, in general, strictly controlled quality-of-service (QoS) and multicasting capabilities at the network level.

Many new standards for packet switching networks, notably ATM(Asynchronous Trasfer Mode), Frame Relay, SMDS(Switched Multimegabit Data Service), include support for multicasting. Future applications will also rely on the ability of the network to perform multicast communications. Thus, multicasting will likely to be an essential part of the future networks[4].

Redundant and replicated servers are useful for ensuring reliability and scalability, and make several other benefits possible. Each receiver may be connected to the server following the shortest delay path or added to a multicast connection using the minimum cost path. This makes it possible for high-volume multicast traffic to be distributed efficiently. The anycast service to be used for resource discovery, which is defined in IPv4 and IPv6, also proposes the use of replicated servers. In situations where many clients are to receive data from a given server, locating identical redundant servers will reduce delays from server to client and can be used for minimizing the total number of server-client connections, allowing efficient transmission of data by use of a multicast service. Digital broadcast systems provide a good example of such an operation with their settop box and antennas for receiving, tuning, and decoding at the client location, digital broadcasting data transmitted via satellite from a broadcast station.

When upgrading softwares or transferring files to multiple sites, the client should be connected to a specific server or multicast connection. For such applications, we can improve overall system performance by locating multiple servers closer to the clients and the multicasting data. Traditionally, multicast routing has been modeled as the problem of finding a tree connecting a source node to multiple destination nodes. However we have extended it to computing a set of multicast trees connecting destination nodes to a group of identical redundant sources. Multicasting that utilizes replicated multiple sources can be viewed as a means of connecting destinations to a source group which can be aggregated to form a virtual source node. Media on demand services anchored by a multicasting server, Internet broadcast systems using real-time streaming techniques, and high volume file or software distribution services may utilize source group multicasting architecture in order to reduce server overhead and improve reliability. That is, multiple redundant servers can create their own multicast transport tree and serve client requests separately. To make such source group multicast services possible, multicast routing algorithm should be enhanced.

Here, we are interested in a minimum cost distributed route computation algorithm for connection-oriented services that satisfies the end-to-end delay constraint, which is specified by the application network management modules. The Application evironment where multicast routing devised in this paper is applied assumes that identical multiple sources are sending multimedia contents to receivers.

Section 2 describes the problem definition, and section 3 presents the distributed heuristic algorithm. In section 4, support for the proposed algorithm is presented. In section 5, the network model and performance results obtained by simulation are presented. In section 6 the related works on this issue are presented. We conclude the paper in section 7.

2 Distributed Source Group Multicast Route Computation Problem

The network is formulated as an undirected connected graph G with a set of nodes V and links E with attributes

*This work is sponsored by the Republic of Korea Ministry of Information and Communication

such as the delay and the cost. "Undirectd" means that links are symmetrical, namely the cost and delay for the link (v, w) and (w, v) are same. We assumed the undirected network model for simplicity.

The delay between two nodes is the propagation time. The cost is a measure of edge's resource utilization. Resource means many things but in general the bandwidth is the main interest. So, The cost in one edge is the function of the amount of traffic traversing the link and expected buffer space needed for that traffic. We will show by simulation the effectiveness of our algorithm.

The nework is defined as follows.

- Graph $G = (V, E)$
- $C_e : E \rightarrow Z^+$ for the cost weighting function
- $D_e : E \rightarrow Z^+$ for the delay weighting function

When multiple redundant senders for a multicast session are distributed all over the network using the network model above, the receiver must be connected to the closest sender to minimize the total cost satisfying delay bound. This problem is defined as follows.

> **With** $G = (V, E)$, C_e, D_e, *a source node group* $S = \{s_1, s_2, ..., s_n\} \subseteq V$, *a multicast group* $D = \{d_1, d_2, ..., d_m\} \subseteq V$, *the delay constraint* Δ
>
> **Find** *a minimum cost tree forest,* $T = \{T_1, T_2, ..., T_t\}, t \leq n$
>
> **Where** S *sends identical multicast data and* T *connects the destination nodes to only one source node satisfying the condition that the delay between the destination node and source node is less than* Δ

The minimum cost tree is a Steiner Tree Problem, and is an NP-complete one. The minimum cost tree with delay bound can be reduced to the Steiner Tree Problem and the problem above can't be solved in polynomial time. So, the algorithm in the next section is a heuristic one.

3 Distributed Source Group Multicast Route Computation Algorithm

3.1 Centralized and Distributed Multicast Route Computation

Multicast tree computation can be done in two different ways: centralized and distributed. In the centralized route computation, also called source-based routing, one node which is aware of the status of the whole network computes the route. The computation is easy and fast in most centralized schemes. However, the overhead to maintain the whole network status in the route computing node can be very large. In the distributed route computation, the overhead of routing table management in a node is small because only the partial knowledge of the network status is sufficient to carry out the algorithm at each node. On the contrary, the computation time takes longer than that of the centralized algorithm since many nodes in the network participate in the route computation by exchanging messages repeatedly over the links. If the time to calculate the tree is large, then the probability of network status change during the calculation becomes high, making the solution not optimal (even infeasible) in the network. The number of exchanged messages and the number of repetitions(cycles) are two important factors deciding the total computation time, and thus are to be minimized in the algorithm.

Considering that the time is the most critical factor in the distributed algorithm, we devised an algorithm that always finds the tree in two or less than two phases while reducing the number of messages and accompanying processing time.

3.2 Network Information For Route Computation in one node

The network informations to be maintained at each node are {*destination node, least delay, cost, next node*}. Especially the source node has the network information about the location of virutal top source merging results from multiple source nodes. The *least delay* is the minimum delay to the *destination node*. And the *cost* is the sum of the link costs over the least delay path to the destination. The *next node* is the adjacent node on the least delay path to forward the packets.

3.3 $DSGDCMT_{md}$ Algorithm

We call the proposed algorithm as $DSGDCMT_{md}$ (Distributed Source Group Delay Constrained Multicast Tree - minimum delay). We assume that each node knows its least delay paths to all other nodes in the network; all least delay paths are symmetric in the sense that nodes v and w share least delay paths between them; each multicat member has knowledge of all other multicast members; each multcast member is able to determine the cost and delay to every other node from its routing table.

In $DSGDCMT_{md}$, the virutal top source node sends `START` message to source nodes, and then the source node sends `TEST_METRIC` message to nodes already included in the tree. Initially, the message is sent to the source node itself. Upon receiving `TEST_METRIC`, each node verifies its routing table to see if there exist delay-constrained paths to the destination nodes not included in the tree (each node is aware of the delay from the source to itself, from the previous steps, and calculates the total delay to each destination by adding the *least delay* value for the destination to the source-to-itself delay). The node selects the one with minimum overall cost among the paths satisfying source-to-destination delay bound and its degree constraint and the related information is carried back to the source node in `TEST_METRIC_ACK` message.

Before sending the `TEST_METRIC_ACK` message to the source node, each tree node collects all `TEST_METRIC_ACK` messages from its children nodes and selects one with the least minimum overall cost among them including its own one. Then, the selected `TEST_METRIC_ACK` message is propagated to the next tree node on the path to the source node. This message merging is repeated at every tree node on the path to the source node. Therefore, the number of `TEST_METRIC_ACK` messages received by the source node is equal to the number of multicast branches it has.

In the source node, it collects `TEST_METRIC_ACK` messages from the children nodes, and selects one with the least cost.

And now, each source node has the information about which path is chosen to add the new node. Each source node sends that result to virutal top source node in the form of `REQ_MERGE`. Virutal top source merges the result

from all source nodes and sends the ADJUST_SRC message to the source node that has the minimum cost path result.

The node that sent the selected information is notified by ADJUST message by the source node, who is selected by the virutal top source. The news is forwarded to the newly added destination by ADD message. The nodes on the path to the added destination are also included in the multicast tree, and participate in the subsequent tree computation steps. The intermediate nodes modify the accumulated delay field in the ADD message, which stores the accumulated delay from the source node, by adding the link delay between itself and the previous node. The modified delay is stored in each node, too. The destination node replies back by ADD_ACK message when it receives the ADD message. And the selected source node sends ADD_ACK_SRC to virtual top source. After this stage, Virtual top source may be reselected.

The source node repeats the whole procedure until there are no destinations left. Thealgorithm produces same number of multicast trees as the number of source. Fig. 1

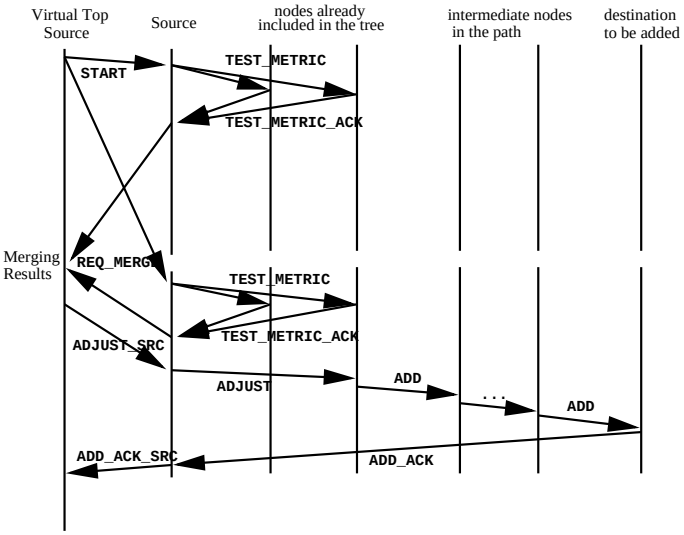

Fig. 1. Message exchanges in $DSGCMT_{md}$ algorithm

depicts the procedure of message exchanges to add one destination node to the multicast tree. It is assumed that degree-constraint, delay and cost informations are provided by the network management modules in advance.

We present the formal algorithm in Appendix, which also includes notations. Fig. 2 gives an example of the tree generation procedure for $DSGDCMT_{md}$.

$DSGDCMT_{md}$ always finds a tree if there exists a tree that satisfies the delay constraint. The exchanged message complexity in the worst cast is $O(|S||V||G|)$, where $|S|$ is the number of source nodes ,$|V|$ is the number of network nodes, and $|G|$ is the number of destination nodes.

Theorem 1 *$DSGDCMT_{md}$ always computes a tree if there exists a tree that satisfies the delay constraint.*

Proof: Assume that $DSGDCMT_{md}$ cannot find a delay-constrained path from $s \subseteq S$ to a node $v_f \in D$. Then $DELAY_{min}(s, v_f) > \Delta$, since $DELAY_T(v) + DELAY_{min}(v, v_f) > \Delta$ for $\forall v \in T$. This is in contradiction to the assumption that there exists a delay-constrained path from s to v_f. Therefore $DSGDCMT_{md}$ always finds a delay-constrained tree if there exists the solution.

Theorem 2 *The exchanged message complexity in the worst cast of $DSGDCMT_{md}$ is $O(|S||V||G|)$, where $|V|$ is the number of network nodes ,$|S|$ is the number of source nodes, and $|G|$ is the number of destination nodes.*

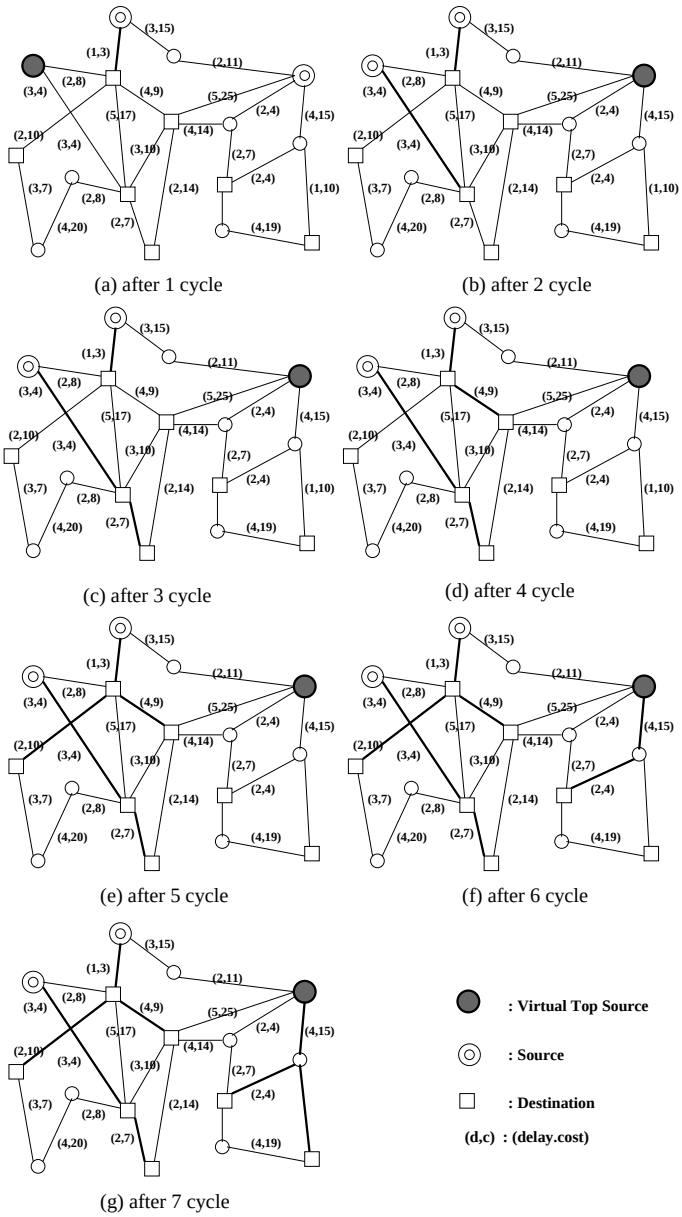

Fig. 2. Example network and tree generation procedure by $DSGDCMT_{md}$

Proof: In the first stage of our proposed heuristics, one source sends TEST_METRIC message to the nodes already included in the tree. Because the number of links from one source and one destination has the maximum value $|V|$, the complexity of links is $O(|V|)$. In addition to the links, the fact that all sources send and receive messages to all destinations already included in the tree makes the whole complexity $O(|S||V||G|)$.

According to the above two good features, $DSGDCMT_{md}$ generates the multicast tree in two phases. One is to add the new destination to a certain source and the next is to recompute the virtual top source. However, virutal top source need not be recomputed at every cycle(a cycle is one pass in the algorithm), because addition of one node does not influence the topology of multicast tree greatly. So, according to network state,

virutal top source recomputation is executed at every 5 to 10 cycles. The number of cycles in the first phase is equal to the number of destinations of multicasting, because we add the whole path to one new destination to the tree at each cycle.

3.4 Virtual Top Source

Virtual top source is responsible for merging results from multiple source nodes to decide which source will attatch the added node to itself. Virtual top source is defined per each multicast session and can be elected by any mechanism. It may be one of multiple sources or may be other third-party node. The selection of virtual top source is irrelevant to the algorithm proposed in this paper.

In this paper a simple and efficient virtual top source election algorithm is considered. We assume that one of multiple sources becomes the virtual top source. The key factor to choose the virutal top source is set to load-balancing. The load at each source node is mainly the copy overhead of multicast data and messages sent to destinations. The copy-overhead is the function the degree of edges in source's multicast tree. The complexity of messages sent to destinations is the function of number of destination nodes in source's multicast tree. This paper proposes a simple virtual top source election policy as follows

> *One of multiple sources becomes virutal top source if its $VW(s)$ in multicast tree is the minimum value of all source nodes ,where $VW(s) = a \times Deg(s) + b \times NodeNum(s), s \subseteq S$, $Deg(s)$ = degree of s's multicast tree, $Nodenum(s)$ = number of nodes in s's multicast tree and a and b are constants.*

The procedure to choose the virtual top source node is executed after a new destination node is added to multicast tree, which is accompanied by the extra message exchange.

4 Support for Source Group Multicast Routing

The current multicast architecture can be applied to source group multicast services with slight or no modification. The source-based multicast tree is designated by the (source, group) pair. As with the DVMRP or MOSPF, we can use a separate tree for each (source, group) combination to enable source group multicasting. Thus, the current routing scheme can support seamless source group multicasts, and source group multicast services can be extended to the current multicast application in a transparent manner. A host can be extended by use of the "source filtering" capability in the IGMPv3[11], which allows a user to specify the source lists. This can make the source group node selection more explicit.

5 Performance Evaluation By Simulation

5.1 Simulation Environment

The simulation environment used in [12], is modified to evaluate the performance of the heuristics in this paper. In this simultaion, networks with link capacity of OC3(155Mbps) are used. The positions of the nodes are fixed in a rectangle of size $3000 \times 2400 km^2$. The size of this area is similar to that of the United States.

Each node in the network functions as an ATM(Asynchronous Transfer Node) switch or IP(Internet Protocol) router in real networks. It is assumed that each node has a small output buffer. The delay in each link is mainly composed of the propagation delay, the queueing delay, and the transmission delay. However, we consider only the propagation delay, which is a dominant components of them. The delay is generated by the Waxman's random graph generator[3]. This graph model is known to generate network graphs similar to real networks.

First, the generator locates a fixed amount of nodes at random places and calculates the probability of a link between two nodes v and w by

$$P_e(v,w) = \beta exp \frac{-d(v,w)}{\alpha L} \quad (1)$$

, where
$d(v,w)$:distance between two nodes v, w
L :maximum possible distance between two nodes
α, β :parameters, $0 < \alpha, \beta \leq 1$

The connectivity of a link is determined by P_e. The values α and β determines the pattern of degree and connectivity of generated network. The delay of a link is set to be linearly proportional to this probability. For the cost of each link, the randomly generated background traffic is assigned to each link. And, Destination and source nodes are also randomly chosen.

5.2 Simulation Results

For each run of the experiments, we generate a random set of links by which the fixed nodes are interconnected with each other and random background traffic for each link. The experiment is repeated with different multicast destination size and source size and different delay bound value. We measure the probability of multicast tree construction success of algorithms and the cost of the constructed multicast tree. The probability of multicast tree construction success is defined as the probability that an algorithm succeeds in constructing a delay-constrained multicast tree spanning all destination and source members for all experimented example network. Each experiment is conducted to guarantee less than 5 % confidence intervals.

In this section, we show three simulation results. Each results shows the performance of $DCMT_{md}$ and $DSGDCMT_{md}$. $DCMT_{md}$ is known to show better performance than many other distributed algorithms whereas it does not support multiple sources explicitly.

Fig. 3 shows the probability of multicast tree construction success of $DSGDCMT_{md}$ and $DCMT_{md}$ for a fixed multicast group of 20 members and 5 sources, when applied to 200 node network. When delay bound is 10msec, both algorithms can not contruct multicast tree because the delay bound is too small. $DCMT_{md}$ achieves probability of multicast tree construction success about 50% for a practical stringent delay bound(25msec). Only when $DCMT_{md}$ reaches delay bound of 40msec, it can always construct multicast tree. In a global view of Fig. 3, $DCMT_{md}$ need more delay bound of 5msec than $DSGDCMT_{md}$. Small delay bound is very important to end host because the larger delay bound, the more buffer space it needs.

Fig. 4 shows the total tree cost excess of $DCMT_{md}$ relative to $DSGDCMT_{md}$ for a fixed multicast group of 20

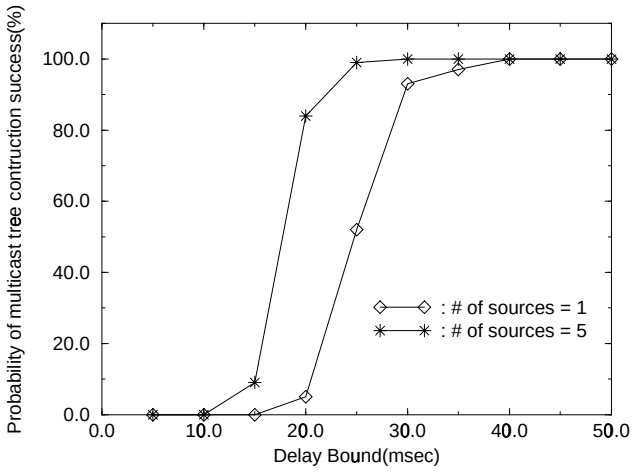

Fig. 3. Probability of multicast tree construction success(number of networks nodes = 200, number of destination nodes = 20)

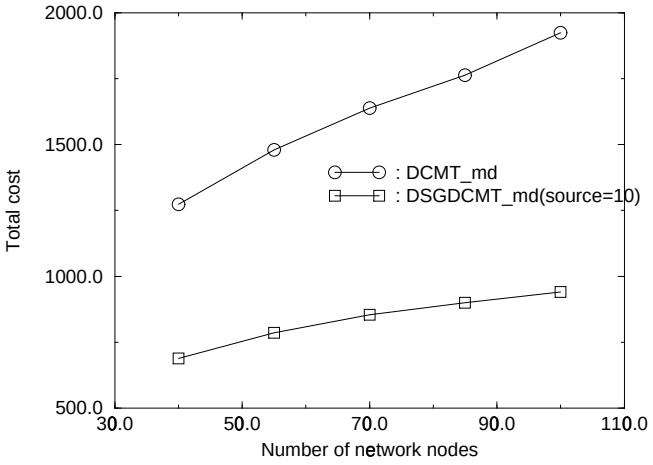

Fig. 4. Total tree cost(number of destination nodes = 20, delay bound = 30msec)

members and 10 sources altering the number of network nodes. When small network($|N| = 40$) is applied to the proposed algorithm, $DSGDCMT_{md}$ has a superior performance comapred to $DCMT_{md}$ by 46%. This performance increase relative to $DCMT_{md}$ is almost same(51%) in large network($|N| = 100$).

The performance increase of $DSGDCMT_{md}$ relative to $DCMT_{md}$ is given in Fig. 5 for a fixed source group of 10 sources assigning different values to number of network nodes. The results of Fig. 5 show the scalability of $DSGDCMT_{md}$. In real networks, there can be various sizes of multicast session. Good routing algorithm must be run effectively regardless of the size of multicast session. For small multicast session($|G| = 15$), total cost of $DSGDCMT_{md}$ is approximately 550, whereas that of $DCMT_{md}$ is 1573. Under this small multicast session, $DSGDCMT_{md}$ has more than 50% performance increase than $DCMT_{md}$. Keeping the performance increase in large multicast session over $DCMT_{md}$ is similar to Fig. 4.

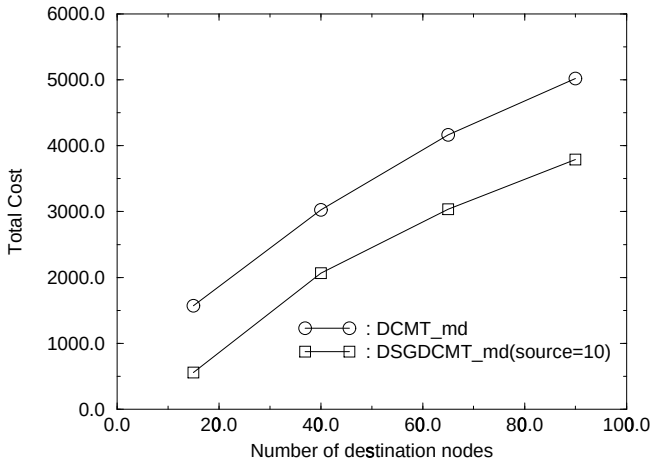

Fig. 5. Total tree cost(number of networks nodes = 100, delay bound = 30msec)

6 Related Researches

Originally, multicast routing problem was conceptualized as the search for a tree connecting the source node and the multicast group receivers on a graph. This multicast tree can be classified as the source-based shortest path tree or the minimum cost tree. The shortest path-based multicast routing algorithm tries to reduce the delay from the source to the destination, and is relatively simple to construct. For example the multicast routing protocol DVMRP[13] used in MBone is based on a slightly modified reverse path multicast routing algorithm.

There has also been much research on finding a minimum cost multicast tree known as the *Steiner* tree[5]. The Steiner tree problem is proved to be $NP - complete$ one. Many heuristics have been proposed to reduce its complexity. There is an algorithm that guarantees that the cost of the multicast tree is not greater than two times that of the optimal solution[3].

Recently, several algorithms have been proposed that satisfy the QoS(Quality of Service) requirements of the multimedia applications; a delay constrained multicast routing algorithm[1], a delay and delay variation constrained algorithm[14], a degree-constrained algorithm[4] and one that considers both delay and bandwidth[2]. The Qos routing algorithms are primarily the centralized algorithm, however the distributed algorithm is also conceived.[7][8] In [12], the authors evaluated and compared many QoS-constrained algorithms.

The feasibility of QoS-constrained algorithm is dependent on resource reservation mechanism. Before reserving network resources, a route satisfying the QoS requested by the application is determined. And then resources along the route are actually assinged hop-by-hop by the resource reservation or channel establishment protocol(e.g., RSVP[6]).

Some researches considering multiple sources or redudant servers have been reported recently ,but those mechanisms deal with multiple sources at the application layer[9] in that this paper deals with multiple sources at the network layer. Previous work on Source group multicasting[10] at the network layer, was on centralized multicast route computation algorithm, not a distributed one.

7 Conclusion

Multicast routing algorithms satisfying stringent delay bound are becoming increasingly important. We proposed a new distributed multicast routing algorithm that considers identical multiple sources. The proposed algorithm finds a minimum cost tree under an end-to-end delay bound and has the scalability in that it works well regardless of number of network nodes and size of multicast session. We proved that it always contructs a delay-bounded multicast tree, if a tree exists, and it has a message complexity of $O(|V||S||G|)$. Especially for an applicatoin requring a stringent delay-bound, the algorithm shows remarkable performace than $DCMT_{md}$.

Appendix

- V: set of network nodes
- S: source node Group
- D: set of destination nodes, where $D \subseteq V - \{S\}$
- v_T: virtual top source node
- T: set of multicast trees
- T_i: set of nodes in the multicast tree of source node i, where $T_i \subseteq T$
- $PATH_{min}(v, w)$: the least delay path between node v and w, where $v, w \in V$
- $COST_{min}(v, w)$: cost of $PATH_{min}(v, w)$
- $DELAY_{min}(v, w)$: delay of $PATH_{min}(v, w)$
- $DELAY_T(v)$: delay from node s to node v in the multicast tree T
- Δ: end-to-end delay constraint specified by the application

```
/*
Procedure to add a new destination to
  already constructed multicast tree
*/
```

Step 1: v_T sends `START` to $s \in S$,

Step 2: Node s, upon receiving `START`, s sends `TEST_METRIC` to $v_s \in T_s$,

Step 3: [1] Node v_s, upon receiving `TEST_METRIC`, constructs `TEST_METRIC_ACK` after finding a destination node w with the least $COST_{min}(v_s, w)$ where $DELAY_{T_s}(v) + DELAY_{min}(v_s, w) < \Delta$ for $w \in S - T$.

Step 4: Before `TEST_METRIC_ACK` is sent to s, v_s collects all `TEST_METRIC_ACK`'s from its children nodes, and selects one with the least $COST_{min}(x, y)$ among them including v_s's own one, where $x \in T_s$ and y is a destination node which is not yet included in the multicast tree. The selected message is sent to s and other messages are discarded by v_s.

Step 5: s selects the node whose $COST_{min}(v_s, w)$ in the `TEST_METRIC_ACK` is minimum, then sends `REQ_MERGE` to v_T. s and v_s may be s_{min} and v_{min} if $COST_{min}(v_s, w)$ is the least of all $COST_{min}(v_i, w)$, $i \in S$ in Step 6.

Step 6: Node v_T, upon receiving `REQ_MERGE`s, chooses the s_{min} that has the minimum added cost and sends `ADJUST_SRC` to s_{min}.

Step 7: Node s_{min}, upon receiving `ADJUST_SRC`, sends `ADJUST` to v_{min}

Step 8: Node v_{min}, upon receiving `ADJUST`, sends `ADD` message to the next node x on the $PATH_{min}(v_{min}, w)$.

Step 9: Node x, upon receiving `ADD`, forwards `ADD` to the next node y on $PATH_{min}(x, w)$,x is included in the multicast tree.

Step 10: Node w, upon receiving `ADD`, sends `ADD_ACK` to s_{min} indicating that w is included in the tree.

Step 11: Node s_{min}, upon receiving `ADD_ACK` sends `ADD_ACK_SRC` to v_s indicating that w is included in the tree.

Step 12: S recomputes the v_s.

Step 13: The algorithm is finished when $D \subseteq T$. Otherwise go to Step 1.

[1] It should be noted, however, that the `TEST_METRIC` message need not be sent to all tree nodes in every cycle. For tree nodes which didn't have the newly added destination node in their previous `TEST_METRIC_ACK` messages, `TEST_METRIC` need not be repeated since we would get the same answers. The above policy and message merging scheme, which was described in Step 3 of the formal algorithm, contribute largely to make the algorithm a scalable one in large network with large multicast group.

References

[1] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, *"Multicasting for Multimedia Communication"*, IEEE Transactions On Networking, vol. 1, no. 3, pp. 286-292, June. 1993.

[2] Z. Wang, and J. Crowcroft, *"QoS Routing for Supporting Resource Reservation"*, IEEE JSAC Special Issues on Multimedia Systems, 1994.

[3] B. M. Waxman, *"Routing of Multipoint Connections"*, IEEE JSAC, vol.6, no.9, pp. 1617-1622, Dec. 1988.

[4] F. Bauer and A. Varma, *"Degree-constrained multicasting in point-to-point networks"*, INFOCOM '95, 1995.

[5] P. Winter, "Steiner problem in networks : a survey," *Networks*, vol. 17, pp. 129-167, 1987.

[6] L. Zhang, S. Deering, D. Estrin, S. Shenker and D. Zappala, *"RSVP: A new resource reservation protocol"*, IEEE Network, vol. 7, no. 5, pp. 8-18, Sep. 1993.

[7] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, *"Two distributed algorithms for multicasting multimedia information"*, ICC '93, 1993.

[8] Yongjun Im, Youngseok Lee, Sunjoo Wi and Yanghee Choi, *"Delay constrained distributed multicast routing algorithm"*, Computer Communications, vol. 20, no. 1, pp. 60-66, Jan. 1997.

[9] Rovert L. Carter, Mark E. Crovella, *"Sever Selection Using Dynamic Path Characterization in Wide-Area Networks"*, INFOCOM '97, 1997.

[10] Youngseok Lee, Yung Yi, and Yanghee Choi, *"Multicast Routing Algorithm Considering Multiple Sources"*, submitted to LCN'98, 1998.

[11] B. Cain, S. Deering, and A. Thyagarajan, *Internet Group Management Protocol, Version 3*, Internet Draft, Nov. 1997.

[12] H. F. Salama, D. S. Reeves and Y. Viniotis, *"Evaluation of multicast routing algorithms for real-time communication on high-speed networks"*, IEEE JSAC, vol. 15, no. 3, Apr. 1997.

[13] D. Waitzman, C. Partridge, and S. Deering, *"Distance Vector Multicast Routing Protocol"* RFC 1075, Nov. 1988.

[14] G. N. Roukas and I. Baldine, *"Multicast Routing with End-to-End Delay and Delay Variation constraints"*, INFOCOM'96, 1996.