

# An Integrated Routing and Admission Control Mechanism for Real-Time Multicast Connections in ATM Networks

Xiaohua Jia, *Associate Member, IEEE*, Wei Zhao, *Member, IEEE*, and Jie Li, *Member, IEEE*

**Abstract**—This letter presents: 1) a delay analysis model, which is specially for the admission control of real-time multicast connections in ATM networks; 2) a distributed multicast routing algorithm, which generates suboptimal routing trees under real-time constraints; and 3) a connection setup method that integrates multicast routing with admission control.

**Index Terms**—Admission control, ATM network, real-time communication, multicast connection.

## I. INTRODUCTION

MULTICAST is a means of group communication that simultaneously transmits messages from a source to a group of destinations. In real-time multicast, messages should be received by all destinations within a specified delay bound. There are many network applications relying on real-time multicast services, such as interactive voice or video conferencing systems, real-time control and monitoring systems, and so on.

ATM is a connection-oriented transport technology. Before any data transmission occurs, a (logical) connection from the source to destination(s) needs to be set up. There are three steps of setting up a connection. The first step is routing, which is the selection of a route from the source to destination(s). Multicast routing [4] is to find a tree rooted from the source and containing all the destinations. The second step is admission control, which is to verify real-time constraints along the selected route. The final step is confirmation of the new connection.

There are several difficulties with the setup of real-time multicast connections. First, it needs to develop a traffic-delay analysis model for the admission control of multicast connections. For unicast, many delay analysis models have been proposed in the literature, such as [2], [3], and [10]. However, little work has been done for admission control of real-time multicast. Second, it is difficult to design a distributed routing algorithm which can be integrated with the real-time admission control. Many delay-bounded multicast routing algorithms (see [12] for a recent survey) have been proposed to find multicast routing trees which minimize the network cost under the constraint that the

delay from the source to any destination is within a bound. However, most of the proposed algorithms suffer from the drawbacks, such as using centralized control [8] or heavy communication overhead [1], [9]. They are not suitable to be integrated with the admission control, which verifies real-time constraints against the dynamic traffic load of the network. Third, it is difficult to perform the on-line admission control. Existing mechanisms separate routing and admission control as two phases of operations [5]. It first generates a routing tree without considering issues of admission control. Then it performs admission tests along the generated tree. Consequently, if the generated routing tree fails the admission tests, another routing tree needs to be computed (which may fail again).

Our letter is motivated by the goal of solving these difficulties in an efficient and practical way. The main contributions of this letter are:

- 1) design of a traffic-delay analysis model for the admission control of real-time multicast connections in ATM networks;
- 2) design of a distributed and delay bounded multicast routing algorithm, which generates suboptimal routing trees under real-time constraints;
- 3) development of an integrated connection setup method, which integrates multicast routing with admission control. It significantly reduces time and messages required for a connection setup.

## II. OPTIMAL MULTICAST ROUTING

The network is modeled by a connected graph  $G(V, E)$ , where the nodes in the graph represent ATM switches and the edges represent communication links. Each edge  $l \in E$  is associated with a weight  $w_l$  that represents the distance of  $l$ .  $w_l$  can be the number of hops in WANs. Fig. 1 is an example of a network graph.

Given a source node  $s \in V$  and a set of destination nodes  $D \subset V (s \notin D)$ , a routing tree of a multicast connection is a subtree of the graph  $G(V, E)$  rooted from  $s$ , which contains all of the nodes of  $D$  and an arbitrary subset of  $(V - D)$ , and whose leaf set consists only of a subset of nodes of  $D$ . When multicasting a message to  $D$ , node  $s$  sends a copy of the message to each of its children in the tree. These children in turn transmit the message to their children until all nodes in the tree (thus, all nodes in  $D$ ) have received the message.

The network resource (e.g., bandwidth) consumption of a multicast is proportional to the number of links the message flows through. By using the tree structure for multicast, a message flows through each tree link once and only once. Therefore,

Paper approved by A. Pattavina, the Editor for Switching Architecture Performance of the IEEE Communications Society. Manuscript received May 5, 1999; revised October 16, 2000, and February 8, 2001. This work was supported in part by the City University of Hong Kong under Grants 7000927, 7001035, and 7001121 and by CERF of Hong Kong under Grant 9040442.

X. Jia is with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong (e-mail: jia@cs.cityu.edu.hk).

W. Zhao is with the Department of Computer Science, Texas A&M University, College Station, TX 77843 USA (e-mail: zhao@cs.tamu.edu).

J. Li is with the Institute of Information Sciences and Electronics, University of Tsukuba, Tsukuba Science City, Ibaraki 305-8573 Japan (e-mail: lijie@is.tsukuba.ac.jp).

Publisher Item Identifier S 0090-6778(01)08159-4.

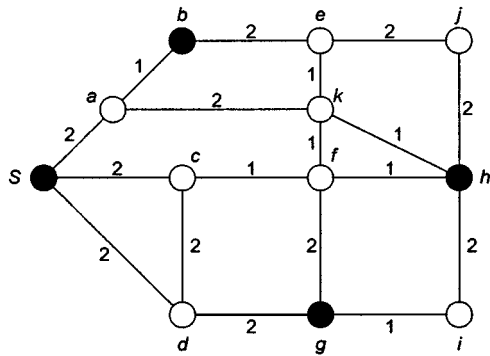


Fig. 1. A simple example of a network graph.

the network resource consumption of a multicast is proportional to the total distances of all links in the tree. We define the network cost of a routing tree  $T$  as the following:

$$\text{Network Cost}(T) = \sum_{l \in T} w_l. \quad (1)$$

By using a tree with less network cost to multicast messages, less network resource will be consumed. An important goal of multicast routing is to minimize the network cost of routing trees. Finding a tree with the minimal network cost is regarded as the Steiner tree problem, which is NP-complete [6].

In connectionless networks, such as the Internet, a routing tree is used only once for a multicast. The gain of minimizing network cost compared with the overhead of finding and maintaining an optimal routing tree is not very crucial. ATM networks are connection-oriented. A multicast connection is set up once and will be used for a while. In this case, using optimal routing trees can substantially reduce network resource consumption.

### III. DELAY ANALYSIS MODEL FOR ADMISSION CONTROL OF MULTICAST CONNECTIONS

Real-time multicast connections have a requirement that the delay for a message to travel from the source to any destination shall not exceed a prespecified bound. In ATM networks, messages are packetized into fixed-size cells. Thus, the delay bound on messages can be achieved by guaranteeing the bound on cells. The end-to-end cell delay from a source to a destination is the summation of delays over the links and switches along the path from the source to the destination. The delays of a cell at a node consist of the following.

- *Switching delay*: the time needed to switch an input cell to an output link buffer.
- *Queuing delay*: the time waiting for transmission at the transmitter of an output link.
- *Transmission delay*: the time needed to transmit the cell to the output link.
- *Propagation delay*: the time needed for a signal to reach the next switch of the link.

The switching delay, transmission delay and propagation delay are constants for an ATM switch and a link. They take much less time than the queuing delay when the traffic is busy. The queuing delay is a variable, which depends on the traffic load.

Let  $M'$  be the new connection to be established. The initial traffic rate of  $M'$  is assumed to be  $\sigma + \rho t$ , which can be achieved by a leaky bucket traffic regulator. The end-to-end deadline of  $M'$  is  $\Delta'$ . The admission conditions for  $M'$  are:

- $M'$  shall not affect the existing connections to meet their end-to-end deadlines;
- $\Delta'$  must be met.

The above two conditions are tested at each node along the routing tree of  $M'$ . Now, consider the admission tests at node  $h$ . Suppose the output link of  $M'$  at node  $h$  is  $l$ . Let  $\{M_1, M_2, \dots, M_m\}$  be the set of connections at node  $h$  on output link  $l$ , and let  $R_i^{\text{in}}(t)$  denote the upper bound of the entering traffic rate of  $M_i$  at node  $h$ ,  $1 \leq i \leq m$ .

*Definition 1*: Maximal Permissible Cell Delay, denoted by  $\Delta_{i,h}^{\text{max}}$ , is the maximal delay that a cell of  $M_i$  is allowed at node  $h$  while its end-to-end deadline can still be met.

*Definition 2*: Worst Case Cell Delay, denoted by  $\delta_{h,l}$ , is the worst case delay that a cell may experience at node  $h$  on output link  $l$ .

As discussed earlier, the delays of a cell at an ATM switch are of two major components: constant delays which include delays of switching, transmission and propagation of the cell, and the variable delay which is the queuing delay of the cell. Let  $\delta_{h,l}^{\text{const}}$  be the total constant delays at node  $h$  on link  $l$ , and  $\delta_{h,l}^{\text{buf}}$  the worst case queuing delay.  $\delta_{h,l}$  can be expressed as

$$\delta_{h,l} = \delta_{h,l}^{\text{const}} + \delta_{h,l}^{\text{buf}} \quad (2)$$

$\delta_{h,l}^{\text{const}}$  can be obtained if the switch hardware and the link distance are given.  $\delta_{h,l}^{\text{buf}}$  is a variable depending on the traffic load on link  $l$ .

Now, we analyze  $\delta_{h,l}^{\text{buf}}$ . Assume that cells are transmitted by switches in FCFS fashion. This assumption simplifies the analysis. However, the general methodology of the integrated routing and admission control can be easily extended to other scheduling algorithms (e.g., the static priority method [13]). Due to the space limitation, we will not discuss these extensions in this letter.

We define the following additional notations before further discussion:  $\Psi_{h,l}(t)$ : the maximum number of cells that may arrive at node  $h$  on link  $l$  during an interval of length  $t$

$$\Psi_{h,l}(t) = \sum_{i=1}^{i=m} R_i^{\text{in}}(t) + R^{\text{in}}(t) \quad (3)$$

where  $R^{\text{in}}(t)$  is the entering traffic rate of the new connection  $M'$ .

$P_{h,l}$ : the cell transmission capacity of node  $h$  on link  $l$ , which is a constant for a given link.  $Q_{h,l}(t)$ : the length of the queue at output link  $l$  of node  $h$  at time  $t$ :

$$Q_{h,l}(t) = \max\{0, \Psi_{h,l}(t) - P_{h,l}t\} \quad (4)$$

If the system is stable,  $Q_{h,l}(t)$  must become zero from time to time, as shown in Fig. 2. That is, any cell in the queue will be transmitted in finite time.

*Definition 3*: *Dormant Point* is a time point  $t$  on which the queue length becomes zero.

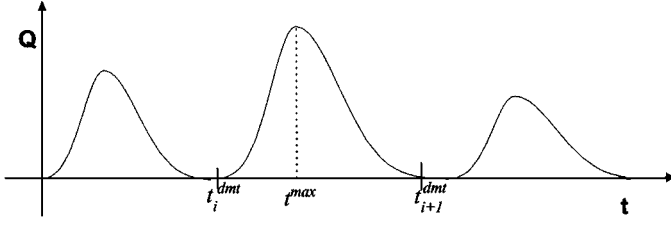


Fig. 2. Changes of the queue length at a switch.

Let  $t_i^{\text{dmt}}$  be the  $i$ th ( $i \geq 0$ ) dormant point

$$Q_{h,l}(t_i^{\text{dmt}}) = \Psi_{h,l}(t_i^{\text{dmt}}) - P_{h,l}t_i^{\text{dmt}} = 0. \quad (5)$$

We assume time is normalized in terms of a cell transmission time, which is a constant for an ATM switch. Therefore,  $t_i^{\text{dmt}}$  can be easily computed by incrementing the value of  $t$  until (5) holds.

Since switches transmit cells in FCFS fashion, the worst case queuing time of a cell  $\delta_{h,l}^{\text{buf}}$  is the largest value of the maximal queue lengths between any two consecutive dormant points (see Fig. 2). That is

$$\delta_{h,l}^{\text{buf}} = \max_{i \geq 0} \left\{ \max_{t_i^{\text{dmt}} < t < t_{i+1}^{\text{dmt}}} (Q_{h,l}(t)) \right\}. \quad (6)$$

$\delta_{h,l}^{\text{buf}}$  can be reached during such an interval that all connections have cells arrival at their peak rates at the start of the interval. Without losing generality, we assume  $\delta_{h,l}^{\text{buf}}$  is reached in  $(t_0^{\text{dmt}}, t_1^{\text{dmt}})$  and  $t_0^{\text{dmt}} = 0$ . Since  $t$  is in the unit of one cell transmission time,  $\delta_{h,l}^{\text{buf}}$  can be obtained by trying all values of  $t$  in between  $(0, t_1^{\text{dmt}})$  in (4) with all connections having their cells arriving at the peak rate from  $t = 0$  (the traffic envelope  $\sigma + \rho t$  has the largest burst rate when  $t = 0$ ). Once  $\delta_{h,l}^{\text{buf}}$  is computed,  $\delta_{h,l}$  can be obtained by (2).

Since connections are not prioritized (cells are transmitted in FCFS order), all the connections at node  $h$  on link  $l$  have the same worst case cell delay, i.e.,  $\delta_{h,l}$ . The admission condition that  $M'$  does not affect the existing connections can be expressed as

$$\forall i: \delta_{h,l} \leq \Delta_{i,h}^{\text{max}}, \quad l \leq i \leq m. \quad (7)$$

Next, consider the admission condition of meeting the end-to-end deadline of  $M'$ . Let  $T$  be the routing tree for the multicast connection, and  $P(s, d)$  be the set of nodes and links on the path from the source  $s$  to destination  $d$  along  $T$ . The admission condition of meeting  $\Delta'$ , the end-to-end deadline of  $M'$ , is

$$\forall d \in D: \sum_{h,l \in P(s,d)} \delta_{h,l} \leq \Delta'. \quad (8)$$

When the worst case end-to-end delay guaranteed by the system, i.e.,  $\sum_{h,l \in P(s,d)} \delta_{h,l}$ , is less than the user specified end-to-end delay  $\Delta'$ , the delay requirement at each node can be relaxed so that more connections can be admitted to the system. The maximal permissible cell delay of  $M'$  at node  $h$ ,

denoted by  $\Delta_h^{\text{max}}$ , can be obtained by distributing the unused part of the delay evenly to the nodes along path  $P(s, d)$

$$\Delta_h^{\text{max}} = \delta_{h,l} + \frac{1}{|P(s, d)|} \left( \Delta' - \sum_{h,l \in P(s,d)} \delta_{h,l} \right) \quad (9)$$

where  $|P(s, d)|$  is the number of nodes on path  $P(s, d)$ .

At node  $h$ , the admission of  $M'$  will change the exiting traffic rates of the connections on output link  $l$ . Let  $R_i^{\text{out}}(t)$  denote the exiting traffic rate of  $M_i$  at node  $h$  on link  $l$ ,  $1 \leq i \leq m$ . According to [3, Theorem 2.1],  $R_i^{\text{out}}(t)$  is

$$R_{\text{out}}^i(t) = R_{\text{in}}^i(t + \delta_{h,l}). \quad (10)$$

Similarly,  $R_{\text{out}}^l(t)$ , the exiting rate of  $M'$  is  $R_{\text{in}}^l(t + \delta_{h,l})$ . The exiting traffic rates from node  $h$  on link  $l$  will be the arrival rates to the node at the other end of link  $l$ . The information about the changed traffic rates of the connections will be sent to the next node via a control channel (or an out-of-band channel), which will be used for computing the delay at the next node.

#### IV. INTEGRATED CONNECTION SETUP METHOD

The principal goal of real-time multicast connection setup is to minimize the network cost defined in (1) under the admission conditions defined in (7) and (8).

A Steiner tree achieves the optimal network cost, but may have a long delay to some of the destinations. On the other hand, an *SPT* (shortest path tree), which contains the shortest path from the source to each destination, has the minimal delay, but may have a high network cost. Fig. 3(a) and (b) shows the *SPT* and the Steiner tree, respectively, for  $D = (b, g, h)$  in the network graph of Fig. 1. In real-time communications, there is no need to minimize the delay as a *SPT* does. Instead, it is required to minimize the network cost under the admission conditions.

Our routing method is a combination of an *SPT* and a Steiner tree. Since finding a Steiner tree is NP-complete, we use an *MST* (minimum spanning tree) as an approximation of the Steiner tree. Routing and admission conditions are performed along two routing trees in parallel: the *SPT* and the *MST*. That is, from the source to any destination, the admission conditions are verified along both the *SPT* and the *MST*. When the admission tests fail at the *MST* on the way to a destination, the shortest path from the source to this destination will be included into the final routing tree. This avoids the problem of discarding the entire *MST* and reconstructing the routing tree when admission tests failed at a node.

To compute the cell delays at a node, each node has to record the real-time connections which go through this node and their arrival traffic rates. When verifying the end-to-end delay at a node, say  $\nu$ , defined in (8), we need to compute the so-far accumulated delay from source  $s$  to this node, denoted by  $\delta_\nu^{\text{acc}}$ . The condition defined by (8) is revised to the following:

$$\delta_\nu^{\text{acc}} + \sum_{u,l \in P(s,\nu)} \delta_{u,l} \leq \Delta' \quad (11)$$

where  $P(s, \nu)$  is a path from  $s$  to  $\nu$  along the routing tree.

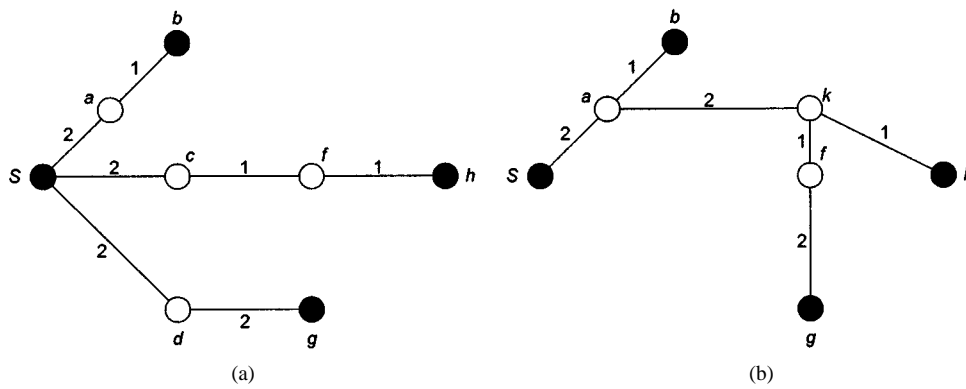


Fig. 3. *SPT* and the Steiner Tree for  $D$  in Fig. 1. (a) *SPT*. (b) Steiner Tree.

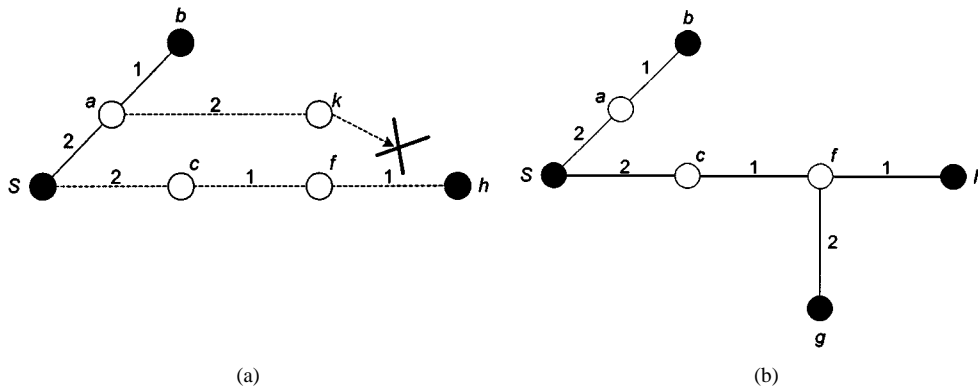


Fig. 4. (a) *MST* path versus *SPT* path. (b) The final tree  $D = \{b, g, h\}$ .

The admission control along the *SPT* is performed on the shortest path to each destination independently. At each node on the *SPT*, the admission conditions defined in (7) and (11) are tested. If the admission tests are passed at this node, the changed output traffic rates of the affected connections will be computed by using (10) and sent to the next node along the path. If the admission tests fail at any node, a *fail* message is sent to  $s$  and the connection setup stops. The connection setup fails.

The admission control along the *MST* is sequential. (The details of this *MST*-heuristic can be found in [7]). The initial routing tree contains only node  $s$ . Each time, a destination which is the closest to the tree is selected and the admission tests are performed along the shortest path from the tree to this selected destination. If the admission tests fail at a node, this shortest path is discarded and the shortest path from  $s$  to the selected destination is included into the routing tree; otherwise this shortest path from the tree to the destination is included into the tree. This operation repeats until all the destinations are included in the tree. The connection setup succeeds. Node,  $s$ , will then confirm the establishment of the connection by sending a *confirm* message through the generated routing tree. During the confirmation at each node, the maximal permissible delay of the new connection at the node is computed by using formula (9).

Fig. 4(a) is an example of constructing a routing tree for  $D = \{b, g, h\}$  of the network graph in Fig. 1. At the time when the tree only contains path  $\langle sab \rangle$ , the *MST* path from the tree to destination  $h$  is  $\langle akh \rangle$  [see Fig. 3(b)]. Assume  $\langle akh \rangle$  fails the admission tests at node  $k$ . The *SPT* path to  $h$ , which is  $\langle scfh \rangle$ ,

is included into the tree. After that, destination  $g$  is linked to the tree by a *MST* path  $\langle fg \rangle$ . Fig. 4(b) is the final routing tree.

## V. SIMULATIONS AND PERFORMANCE DISCUSSIONS

The purpose of the simulations is to demonstrate the quality of the routing trees (in terms of the network cost) generated by our method.

The network graphs used in the simulations are constructed by using the approach given in [11]. The nodes are distributed randomly over a rectangular coordinate grid. Each node is placed at a location with integer coordinates. A link between two nodes  $u$  and  $v$  is added by using the probability function  $P((u, v)) = \lambda \exp(-d(u, v)/\rho L)$ , where  $d(u, v)$  is the distance between  $u$  and  $v$ ,  $L$  is the maximum distance between any two nodes, and  $0 < \lambda \leq 1$ ,  $0 < \rho \leq 1$ . Larger values of  $\lambda$  result in graphs with higher link densities, while small values of  $\rho$  increase the density of short links relative to longer ones. In our simulations, both  $\rho$  and  $\lambda$  are set to 0.7 to make the graphs looking sparse after many tests. Graphs are generated and tested until a connected graph is found.  $w_l$ , the distance of link  $l = (u, v)$  in the graph, is made as the distance of nodes  $u$  and  $v$  on the rectangular coordinate grid.

Three algorithms are simulated and compared: *SPT*, *MST*, and our method. *MSTs* are computed by using our algorithm with no real-time constraints. The network cost is simulated against two parameters:  $\Delta$  and  $|D|$ . The network size is fixed at 200 nodes and the topology does not change throughout the simulations.

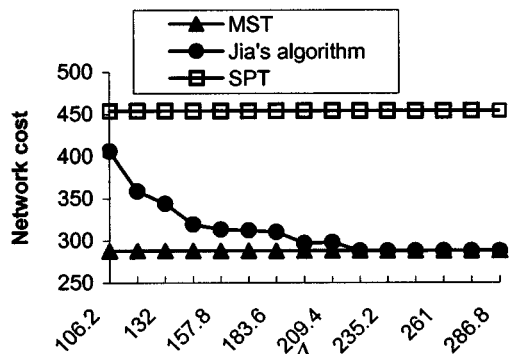
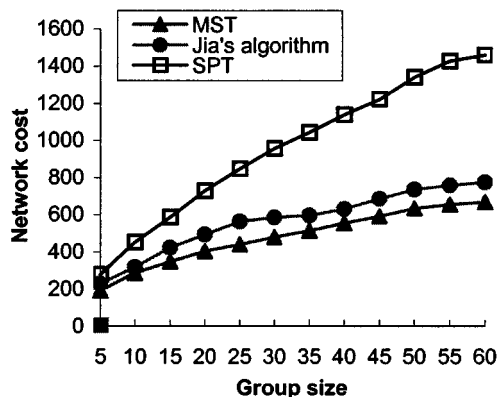
Fig. 5. Network cost versus  $\Delta$ .

Fig. 6. Network cost versus group size.

Fig. 5 shows the network cost versus real-time deadline  $\Delta$ . The group size is set to 10. During the simulations, traffic load of the network is not considered. We simply take the distance of a link as its delay. We define the minimum meaningful delay bound  $\Delta_{\min} = \left( \left\{ \sum_{l \in SP(s,d)} w_l \mid \forall d \in D \right\} \right)$ .  $SP(s,d)$  is the shortest path from  $s$  to  $d$ . The  $\Delta$  value starts from  $\Delta_{\min}$ , incremented by  $\Delta_{\min}/8$  each time. The increment of  $\Delta_{\min}/8$  is selected to capture any meaningful changes of network cost against the change of  $\Delta$ . In Fig. 5, the curves of *SPT* and *MST* remain constant, because both of them are not restricted by  $\Delta$ . The curve of our method is in between the curves of the *SPT* and of the *MST*. Its high end is close to the *SPT*'s curve and the low end merges with the *MST*'s curve. With a smaller  $\Delta$ , more *MST* paths fail and are replaced by the *SPT* paths. This makes the routing tree wider (bush-like), thus a higher network cost. As  $\Delta$  increases, more destinations are linked into the tree via *MST* paths, which results in the decrease of the network cost. When  $\Delta$  is large enough that it does not restrict routing any more, the final routing tree becomes an *MST*.

Fig. 6 shows the network cost versus group size. The value of  $\Delta$  is set to  $\Delta_{\min} + 3\Delta_{\min}/8$ . Group size is always made less than 30% of the total nodes, because multicast applications running in a wide area network usually involve only a small number of nodes in the network. As group size increases, a routing tree contains more destinations, resulting in an increase of the network cost. The *SPT*'s curve is above the other two and rises

much faster than the others. This is because *SPT* does not consider any path sharing. The performance of our method is close to that of the *MST*'s. The curves of our method and the *MST*'s rise slower as the increase of group size (the rise is not linear). This is because destinations in a bigger group have a higher probability of path sharing.

The simulation results show that the routing trees generated by our method have much less network cost than *SPT*'s. They also suggest that *SPT*'s is not appropriate to be used as routing trees where the multicast traffic is heavy and the network bandwidth is a major concern.

## VI. CONCLUSION

We have presented a traffic-delay analysis model for ATM networks and an integrated admission control method for multicast connections. There are three advantages of our admission control method. First, it is fully distributed. Second, it integrates routing together with admission control as a single phase of operations, which significantly reduces the time and message cost for a connection setup. Third, it sets up connections on both *SPT* routing and *MST* in parallel, which avoids the unnecessary restart of routing if the *MST* routing fails admission tests.

## ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their valuable comments and constructive suggestions that helped improve the quality of the letter.

## REFERENCES

- [1] F. Bauer and A. Varma, "Distributed algorithms for multicast path set up in data networks," *IEEE/ACM Trans. Networking*, vol. 4, pp. 181–191, Apr. 1996.
- [2] J.-Y. Le Boudec, "Application of network calculus to guaranteed service networks," *IEEE Trans. Inform. Theory*, vol. 44, pp. 1087–1096, May 1998.
- [3] R. L. Cruz, "A calculus for network delay," *IEEE Trans. Inform. Theory*, vol. 37, pp. 114–141, Jan. 1991.
- [4] S. E. Deering and D. R. Cheriton, "Multicast routing in datagram internetworks and extended LAN's," *ACM Trans. Comput. Syst.*, vol. 8, no. 2, pp. 85–110, May 1990.
- [5] V. Firoiu, J. Kurose, and D. Towsley, "Efficient admission control for EDF schedulers," *IEEE INFOCOM*, 1997.
- [6] Gilbert and H. O. Pollak, "Steiner minimal tree," *SIAM J. Appl. Math.*, vol. 16, 1968.
- [7] X. Jia, "A distributed algorithm of delay bounded multicast routing for multimedia applications in wide area networks," *IEEE/ACM Trans. Networking*, vol. 6, pp. 828–837, Dec. 1998.
- [8] X. Jia, N. Pissinou, and K. Makki, "A delay bounded multicast routing algorithm in wide area networks," *Comput. Commun. J.*, vol. 20, no. 12, pp. 1098–1106, Nov. 1997.
- [9] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, "Multicast routing for multimedia communication," *IEEE/ACM Trans. Networking*, vol. 1, pp. 286–292, June 1993.
- [10] J. Liebeherr, E. Wrege, and D. Ferrari, "Exact admission control for networks with a bounded delay service," *IEEE/ACM Trans. Networking*, vol. 4, pp. 885–901, Dec. 1996.
- [11] B. M. Waxman, "Routing of multipoint connections," *IEEE J. Select. Areas Commun.*, vol. 6, pp. 1617–1622, Dec. 1988.
- [12] B. Wang and J. C. Hou, "Multicast routing and its QoS extension: Problems, algorithms, and protocols," *IEEE Network*, pp. 22–36, Jan.–Feb. 2000.
- [13] C. Li, R. Bettati, and W. Zhao, "Static priority scheduling for ATM networks," in *Proc. IEEE Real-Time Systems Symp.*, Dec. 1997.