Distributed multicast routing algorithm with dynamic performance in multimedia networks

Zhu Baoping, Zhang Kun
(School of Computer Science and technology, Nanjing University of Science & Technology, Nanjing 210094, China)

Abstract: The delay and DVBMT problem is known to be NP-complete. In this paper, an efficient distributed dynamic multicast routing algorithm was proposed to produce routing trees with delay and delay variation constraints. The proposed algorithm is fully distributed, and supports the dynamic reorganizing of the multicast tree in response to changes for the destination. Simulations demonstrate that our algorithm is better in terms of tree delay and routing success ratio as compared with other existing algorithms, and performs excellently in delay variation performance under lower time complexity, which ensures it to support the requirements of real-time multimedia communications more effectively.

Key words: multicast routing; distributed algorithm; dynamic performance; delay and delay variation-bounded; multimedia networks

1 Introduction

Multicast is a kind of group communications for the delivery of packets from a single source to multiple destinations. The destinations are the members of a multicast group, and the group membership is said to be dynamic when a destination may join or leave the group at any instant during the session. At the routing level, a multicast routing scheme is responsible for determining the packet delivery path from the source to all destinations, typically a multicast tree. The routing scheme must dynamically reconstruct the delivery tree when membership changes.

In real-time multicast applications, messages must be transmitted from the source node to their destinations within a certain amount of time which requires the communication to be done within a pre-specified end-to-end delay bound. The stringent delay constraint imposed on multimedia traffic to ensure that audio and video data are delivered smoothly to the audience. Besides, the multicast tree must also guarantee a bound on the variation among the delays along the individual source-destination paths, which can probably avoid causing inconsistency or unfairness problem among users. Such a bound provides synchronization among the various receivers and ensures that no receiver is "left behind" and that none is "far ahead" during the lifetime of the session.

Our research subject is concerned with multicast routing with delay and delay variation constraints. The issue first defined and discussed in Ref. [1] is that of minimizing multicast delay variation under multicast end-to-end delay constraint. The authors referred to this problem as the delay and delay variation-bounded multicast tree (DVBMT) problem, and proved it to be NP-complete.

Heuristics to construct multicast tree satisfying the end-to-end delay and delay variation constraints have been developed in Ref. [1] ~ Ref. [3]. Rouskas and Baldine have pro-posed a delay variation multicast algorithm (DVMA)[1]. DVMA either returns a feasible tree, or having failed to discover such a tree, it returns a tree which satisfies the delay and has the least value of delay variation among the trees considered by the algorithm. The time complexity for DVMA is as high as $O(klmn^4)$ where m is the number of destination nodes, n is the number of network nodes, and k and l are the number of paths generated by the algorithm. The actual values of k and l are determined by the desired comprise between the quality of the final solution obtained by the algorithm and its speed. Based on the ideas from CBT and the minimum delay path algorithm, Sheu, et al. have presented another fast heuristic algorithm named DDVCA[2]. This algorithm has a much lower time complexity $O(mn^2)$ and a satisfactory performance. Yu, et al. presented a heuristic of multicast routing with delay and delay variation constraints, called SP-DVMA[3]. Their proposed algorithm is based on the shortest path set and improves DVMA by using following ideas. The authors firstly connect shortest

Received 19 *August* 2008

path to any node in a tree for each time. Then find the optimal path and add to the tree. The time complexity for this algorithm is $O(mkn^3)$. However, SP-DVMA increases the scope of optional paths and may cause the argument of the delay variation for multicast tree.

However, there are two major difficulties in deployment and application of the existing algorithms to real-time communication networks. Firstly, most of the existing algorithms are centralized in nature. A centralized algorithm requires a central node (or every node) to be responsible for computing the entire routing tree, and this central node must have the full knowledge about the global network. It suffers from some drawbacks in large networks, such as poor fault tolerance, heavy computing load at the central node, high communication cost in keeping network information up-to-date, and inaccuracy of routing information. The other difficulty is that, only a few existing algorithms are concerned with the dynamic change of multicast memberships. As we know, in many multicast applications, multicast participants are free to leave or join a multicast session dynamically. Therefore, it is important to ensure that any change of multicast memberships will not affect the traffic on the current connection, and the routing tree remains minimally disruptive to the multicast session.

To the best of our knowledge, little work has been done on finding delay and delay variation-bounded multicast routing tree in a distributed manner so far. In this paper, we propose a distributed multicast routing algorithm for obtaining multicast trees with delay and delay variation constraints, aimed at overcoming the above two difficulties. The proposed algorithm has the following advantages.

1) Fully distributed. Each node operates based on its local routing information and coordination with other nodes is done via network message passing;

2) Dynamic changes of multicast memberships. We also give a method to dynamically reorganize the multicast tree in response to changes for the destinations, and guarantee the minimal disruption to the multicast session;

3) High performance with low complexity. Our algorithm performs excellently in delay, delay variation, and routing success ratio with a lower time complexity, which ensures it to support the requirements of real-time multimedia communications more effectively.

2 The problem definition

We represent a communication network by an undirected graph $G(V,E)$, where V denotes the set of nodes, and E, the set of edges, corresponds to the set of communication links connecting the nodes. Any link has a delay $d(e): E \rightarrow R^+$ associated with it, where $d(\cdot)$ represents the delay that the packet experiences on link including queuing, transmission, and propagation delay. Let $s \in V$ be a source node and $M \subseteq V - \{s\}$ be the set of destination nodes, called the multicast group. A multicast tree $T(T \subseteq G)$ is a tree rooted at s and spanning the nodes in M. Let $p(u,v)$ denote the path from u to v. Then, multicast packets from u to v experience a total delay of $\sum_{e \in p(u,v)} d(e)$.

For the sake of convenience, we use Δ_T and δ_T to represent the multicast end-to-end delay and the multicast delay variation in a multicast tree T. Based on these definitions, we can formally present the DVBMT problem as follows:

Definition 1 Delay and DVBMT problem. Given a network $G(V,E)$, a source node s, destination node set M, a link delay function $d(\cdot)$, a positive delay bound Δ and a positive delay variation bound δ, the objective of the DVBMT problem is to construct a multicast tree $T(V_T, E_T)$ which spans s and M such that the delay and delay variation constraints are satisfied, i. e.,

$$\Delta_T = \max_{m \in M}\Big(\sum_{e \in p(s,m)} d(e)\Big) \leqslant \Delta \quad (1)$$

$$\delta_T = \max_{u,v \in M}\Big\{\Big|\sum_{e \in p(s,u)} d(e) - \sum_{e \in p(s,v)} d(e)\Big|\Big\} \leqslant \delta \quad (2)$$

3 Our proposed algorithm

In this section, we firstly give the assumptions and basic ideas of our algorithm. Then describe the proposed algorithm in detail, and consider an example for ease of understanding. Finally, the analysis of correctness and complexity for our algorithm is discussed.

3.1 Assumptions and basic ideas

The basic idea of our proposed algorithm derives from following theorem.

Definition 2 Adding a path $p(u,v)$ into a tree T refers to all nodes and links on the path are included into the tree, denoted by $T + p(u,v)$.

Theorem 1 Given a network $G(V,E)$, a source node s, destination set M. Δ and δ are delay bound and delay variation bound of multicast session respectively. Suppose T' is a subtree, and $\Delta_{T'} \leqslant \Delta$, $\delta_{T'} \leqslant \delta$. Sub$(M)$ is destinations covered in T', and Sub$(M) \subset M$. We use max$d$ and min$d$ to represent the maximal delay and minimal delay of the path among the paths from s to each destination of Sub(M) in T', respectively. $\forall m \in M$, $m \notin$ Sub(M), if $p(s,m)$ satisfies,

$$\max\{0, \max d - \delta\} \leqslant d(p(s,m)) \leqslant \min\{\min d + \delta, \Delta\} \quad (3)$$

$$T = T' + p(s,m) \quad (4)$$

then $\Delta_T \leqslant \Delta$, $\delta_T \leqslant \delta$.

Proof Obviously, we just need to prove,

$\forall m' \in \mathrm{Sub}(M)$ and $d(p(s,m)) \leqslant \Delta$,

$$|d(p(s,m)) - d(p(s,m'))| \leqslant \delta \quad (5)$$

We can see from Eq. (3) ,

$\max d - \delta \leqslant d(p(s,m)) \leqslant \min d + \delta$.

$\because \min d \leqslant \max d$,

$$\min d - \delta \leqslant \max d - \delta \leqslant d(p(s,m)) \leqslant \min d + \delta \leqslant \max d + \delta \quad (6)$$

$\forall m' \in \mathrm{Sub}(M)$, $\because \min d \leqslant d(p(s,m')) \leqslant \max d$. Hence,

$$\min d - \delta \leqslant d(p(s,m')) - \delta \leqslant \max d - \delta \leqslant d(p(s,m)) \leqslant \min d + \delta \leqslant d(p(s,m')) + \delta \leqslant \max d + \delta \quad (7)$$

i. e. $|d(p(s,m)) - d(p(s,m'))| \leqslant \delta$, $\forall m' \in \mathrm{Sub}(M)$.

$\because d(p(s,m)) \leqslant \min\{\min d + \delta, \Delta\} \leqslant \Delta$.

Thus, $\Delta_T \leqslant \Delta$, $\delta_T \leqslant \delta$ after adding $p(s,m)$ into the subtree T' .

Theorem 1 shows that during of constructing a multicast tree which meets delay and delay variation constraints, if the delay of a path from s to next uncovered destination satisfies Eq. (3), and then the tree after adding this path is still a feasible tree.

The basic idea of our proposed algorithm works as follows. We firstly compute the k least delay paths from s to each destination node $m \in M$ by using the distributed k-Bellman-Ford (kBF) algorithm as a candidate-paths-set[4]. Then, a destination node is randomly selected, and the least delay path from s to this destination in candidate-paths-set is added into an initial empty tree T. At each step, we select the first path in candidate-paths-set, which starts from s to a nontree destination and satisfies Eq. (3), and add it to T. This operation repeats until all nodes in M are included in the tree.

We assume that each node has the information about the k shortest paths (in terms of delay) and the delay of each path to every destination node. The information is stored in the local routing table denoted by Route at each node. This can be achieved by running the distributed kBF algorithm on delay metric. In Route, each node $v \in V$ consists of $k \times |M|$ entries, one entry for the kth shortest path of every destination node. An entry Route[i][m] has two fields Route[i][m].n and Route[i][m].d, representing the next neighbor node on the i^{th} least delay path from v to m and the delay of this path, respectively.

A simplified data structure for a control message is a 3-tuple <type, k^{th}, dest>, where type is the type of the message, k^{th} denotes the k^{th} least delay paths in candidate-paths-set, and dest is the current selected destination. Five main types of messages are used in our algorithm, which are

open—opening a multicast connection, and getting candidate-paths-set;

start—starting the construction of the multicast tree;

add—adding a candidate path from the source to a destination node into the tree;

notify—notifying the source that a destination has been added to the tree;

finish—finishing the construction of the multicast tree.

3.2 Algorithm details

Every node in the system executes the same routing algorithm. It is initially in an idle state waiting for connection setup requests.

When a node receives a request (open message) for opening a multicast connection, with parameters such as destination set M, a delay bound Δ and a delay variation bound δ , it computes the k least delay paths from s to each destination node $m \in M$ by using the distributed kBF algorithm. Let P_m be the set of k least delay paths for the destination m, i. e. ,

$$P_m = \{p_1(s,m), p_2(s,m), \ldots, p_k(s,m)\}$$
$$\text{with } d(p_1(s,m)) \leqslant d(p_2(s,m)) \leqslant \ldots \leqslant d(p_k(s,m)) \quad (8)$$

When source s receives an start request, an empty tree T is first initialized. Then a destination node m is randomly selected. Let max$d$ and min$d$ be the maximal delay and minimal delay of the path among the paths from s to each destination covered in T so far, respectively. max$d$ and min$d$ are initialized as $d(p_1(s,m))$ (i. e. Route[1][m].d). A connection add message <add, 1, m> is sent to the neighbor v via which the selected destination m can be reached by $p_1(s,m)$. The edge (s, v) is added into T.

When the add message arrives at an intermediate node, say u, on the way to the designated destination m, it passes this add message <add, k^{th}, m> to its next neighbor (u'), leading to m (u' = Route[k^{th}][m].n). Then the edge (u, u') is added to T.

When the add message reaches the designated destination, an notify message is sent to s to show a destination has been added to the tree. Upon the receipt of this notify message, a nontree destination, say m' , is selected. Then, the first candidate path whose delay is distributed between [$\max(0, \max d - \delta)$, $\min(\min d + \delta, \Delta)$] is picked out from candidate-paths-set. Suppose this candidate path is the i^{th} least delay path, then the message <add, i, m'> is sent to the neighbor v' via which m' can be reached by the selected path. After updating the values of max$d$ and min$d$, the edge (s, v') is added into T.

The above operation continues as the multicast connection is extended to destinations one after another, until all destinations in M are included in T. When the add request reaches the last destination in M, it sends a finish message to s. The construction of multicast tree satisfying both delay and delay variation constraints is completed.

For ease of understanding, we consider an example network as shown in Fig. 1, where $s = \{0\}$ and $M = \{3, 5, 6, 7\}$, the number along each edge represents the delay for that edge. Given $\Delta = 7$ and $\delta = 3$. The candidate-paths-set of the example network, obtained by using the distributed kBF algorithm, is shown in Table 1, where $k = 5$.

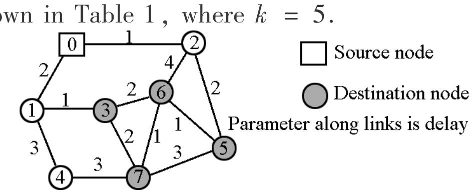

Fig. 1 The example network used in explaining the proposed algorithm

Fig. 2 shows the process of producing the multicast tree with both delay and delay variation constraints. The destination node 5 is firstly selected, and then the least delay path (0,2,5) from 0 to 5 is added. The resultant tree T so far is shown in Fig. 2(a). Then, node 6 is selected as the next adding destination. Since the max$d$ and min$d$ are equal to 3, we should choose the first candidate path whose delay is distributed between $[\max\{0,3-3\}, \min\{3+3,7\}]$ (i. e., $[0,6]$) toward 6. Therefore, the path (0,2,5,6) with a delay of 4 is added in T, shown in Fig. 2(b). Next, node 3 is included into T by the path (0,1,3) in the same way. We can see the final tree constructed so far in Fig. 2(c).

At this point of time when node 3 is added in T, the max$d$ and min$d$ are updated as 4 and 3 respectively. When considering the last destination 7, the first candidate path whose delay is distributed between $[\max\{0,4-3\}, \min\{3+3,7\}]$ (i. e. $[1,6]$) from 0 to 7 is (0,1,3,7). Thus destination 7 is linked to T by (0,1,3,7). Eventually, the final tree T which contains all destinations is produced, shown in Fig. 2(d).

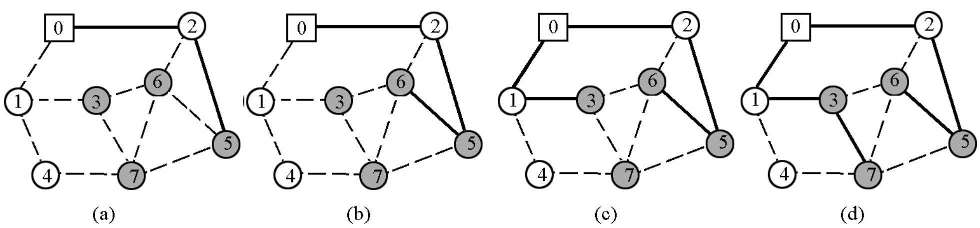

Fig. 2 The process of producing the multicast tree *T*

3.3 Discuss of the algorithm

Theorem 2 (Correctness of proposed algorithm) A delay and delay variation-bounded multicast routing tree will be always found if one exists.

Proof Suppose there exists a delay and delay variation-bounded tree for a source s, a set of destination M. During the construction of multicast tree, algorithm firstly adds the least delay path from s to one of selected destination into the tree, which obviously does not violate the two bounds. At each step, we always select a path satisfying Eq. (3) from s to a nontree destination in candidate-paths-set, and add it to the tree. According to Theorem 1, the tree, after this path is added, is still feasible. As a result, our algorithm can always find a delay and delay variation-bounded multicast tree if one exists.

Theorem 3 In the worst case, the message complexity of our proposed algorithm is $O(mn)$, and the time complexity is $O(k^2 m^2 n \log k)$, where m is group size, n is network size, and k is the number of paths generated by using kBF.

Proof In our algorithm, the add message for each destination will be sent at most n times. Since there are m destinations, there will be at most $O(mn)$ number of add messages. Other messages will not be set more than m times. Therefore, the worst message complexity of our algorithm is $O(mn)$.

In terms of time complexity, generating k least delay paths for m destinations by using kBF costs $O(k^2 m^2 n \log k)$, and the rest of our algorithm costs $O(m)$. So, the worst time complexity of our algorithm is $O(k^2 m^2 n \log k)$[4].

Table 1 The candidate-paths-set of example network, $k=5$

dest	3		5		6		7	
k^{th}	path	d	path	d	path	d	path	d
1	0,1,3	3	0,2,5	3	0,2,5,6	4	0,1,3,7	5
2	0,2,5,6,3	6	0,1,3,6,5	6	0,1,3,6	5	0,2,5,6,7	5
3	0,2,6,3	7	0,2,6,5	6	0,2,6	5	0,1,3,6,7	6
4	0,2,5,6,7,3	7	0,1,3,7,6,5	7	0,1,3,7,6	6	0,2,6,7	6
5	0,2,6,7,3	8	0,1,3,7,5	8	0,2,5,7,6	7	0,2,5,7	6

4 Dynamic reconstruction of the tree

For certain multicast applications, multicast participants may join or leave the multicast group dynamically during the lifetime of the multicast connection. It is important to ensure that any change of multicast memberships will minimize both the cost incurred during the transition period and the disruption caused to the receivers, and the routing tree after the change will always satisfy the constraints (1) and (2) for the current destination set.

In our method, when a destination node $m \in M$ decides to leave the multicast group, if m is not a leaf node, then no action needs to be taken. The new tree can be the same as the current tree T, with the only difference being that node m will stop forwarding the multicast packets to its local user and perform only switching operations. If, however, m is a leaf node, then a leave request is sent upward (to the source direction) along the tree, node by node, until it reaches the source node or another destination. At each node this request passes through, the connection is released. As the result, the new tree is essentially the same as T except in parts of the path from the source to m.

When a node $v \notin M$ wants to join an existing multicast group, it sends a join request to the source. We distinguish following three cases:

If $v \notin V_T$, we get k least delay paths from source to v. Then select the first path satisfying Eq. (3) and add it to T, which is similar to the main steps of our algorithm. If this fails to discover such a path, then deny the participation of node v in the multicast session and discard its join request.

If $v \in V_T$, and the path from source to v is such that the delay variation constraint (2) is satisfied for the new multicast group $M \cup \{v\}$. T is then a feasible tree for the new group, and can be used without any change other than having node v now forward multicast packets to its user, in addition to forwarding them to the downstream nodes.

If $v \in V_T$, but the path from source to v is such that constraint (2) is not satisfied for the new group $M \cup \{v\}$. It shows that v must be an intermediate node in the path from source to other destination or destinations. As a result, we will delete the paths which contain v and the destination(s). Then add v and the destination(s) to the routing tree, one by one, until all of them are included in the tree.

5 Simulation

In the following simulations, we will compare the performance of our algorithm with other four delay and delay variation-bounded routing algorithms. Five algorithms, namely a distributed version of Bellman-Ford Shortest Path Algorithm (SPA), DVMA, DDVCA, SP-DVMA, and the one we proposed have been implemented in a QoS routing simulator (QRSIM) designed by us and written in C ++ [1-3, 5]. All simulations are run on a Pentium IV 2.8 GHz, 512 MB RAM, DELL PC.

Generating network topology is based on the random link generator (based on Waxman's generator with some modifications) developed by Salama, which yields networks with an average node degree of 4 [6, 7]. The positions of the nodes are fixed in a rectangle of size 4 000 km × 2 400 km. The Euclidean metric is then used to determine the distance between each pair of nodes. Edges are introduced between pairs of nodes u, v with a probability that depends on the distance between them. The edge probability is given by $P(u,v) = \beta \exp\left[\frac{-l(u,v)}{\alpha L}\right]$, where $l(u, v)$ is the distance from node u to v, L is the maximum distance between two nodes. α and β are parameters, and are set to 0.15 and 2.2 respectively. Larger values of β result in graphs with higher edge densities, while small values of α increase the density of short edges relative to longer ones.

The link delay function $d(e)$ is defined as the propagation delay of the link, and queuing and transmission delays are negligible. The propagation speed through the links is taken to be two thirds the speed of light. At each simulation point, we run the simulation 500 times and the result is the mean value of the results produced by these 500 runs. Each time, the source node and the destination nodes are randomly picked up from the network graph. Note that δ is kept constantly at 0 in DVMA algorithm (it forces DVMA to return the smallest delay variation that it can find).

Fig. 3(a) shows the delay of varying network size with group size $m = 5$, $\Delta = 35$ ms, and $\delta = 20$ ms. Fig. 3(b) shows the delay performance measures versus group size for a 50-node network with $\Delta = 35$ ms and $\delta = 20$ ms. It is easy to see from two subfigures in

Fig. 3 that our algorithm has the best delay performance among all algorithms. SPT algorithm gives slightly higher delay than our algorithm, but is much lower than that of other three algorithms. As the number of network nodes and group members increase, the maximum end-to-end delay of all algorithms increases, but below the 35 ms delay bound.

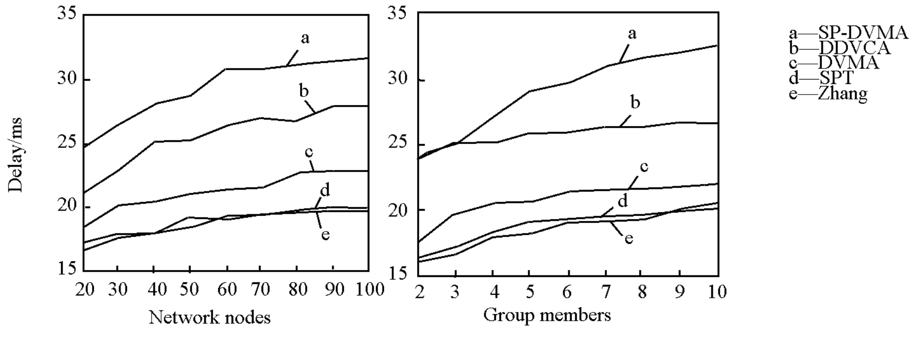

Fig. 3 A comparison on the delay performance

Fig. 4 shows the delay variation of tree for all algorithms in above simulation environment. It can be seen that the DVMA has the optimum delay variation performances as expected. Our algorithm gives slightly higher delay variation performance than SPT. Fig. 4(b) shows the delay variation of all algorithms increases as the group size increases. This is expected since, the larger the size of the multicast group, the larger number of the destination nodes physically closer or farther to the source, which results in the increase of the delay variation between destination nodes.

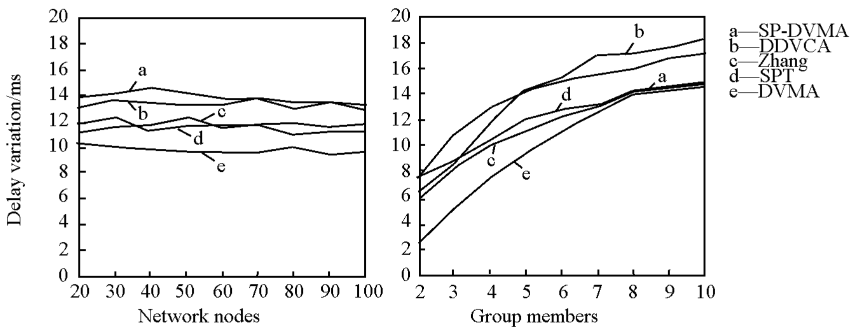

Fig. 4 A comparison on the delay variation performance

Finally, we compare the routing request success ratio (SR) for three algorithms (DDVCA, SPT and Zhang's). SR is defined as the ratio of the number of multicast routing requests accepted and the total number of requests generated. Fig. 5(a) and Fig. 5(b) show the SR of routing requests for different number of

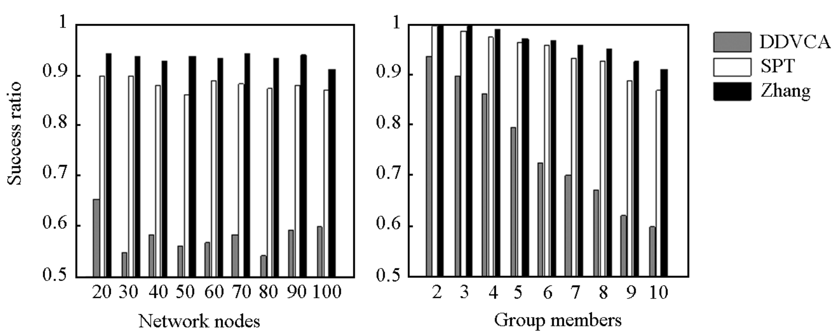

Fig. 5 A comparison on the success ratio of routing request

network nodes (from 20 to 100 in steps of 10, m = 10) and different group sizes (from 2 to 10 in steps of 1, n = 100), respectively. Δ and δ are kept constant at 35 ms and 20 ms. We observe from two subfigures that our algorithm achieves higher SR than other two algorithms for all scenarios we tested. It is obvious from Fig. 5(b) that as the group size increases, the SR of all algorithms decreases. This is because the delay variations between destinations increase as the group size increases, and then possibility of satisfying δ will decrease.

6 Conclusion

In this paper, we discussed the problem of constructing multicast routing trees satisfying the end-to-end delay bound and delay variation bound, which is called DVBMT problem and has been proved to be NP-complete. We have presented an efficient distributed dynamic multicast routing algorithm for obtaining such trees. We firstly compute candidate least paths in terms of delay from source to each destination. Then starting from an empty tree, we iteratively add a candidate path satisfying specific condition to the selected destination into the tree. This operation repeats until all destinations are included in the tree.

The proposed algorithm is always able to find a delay and delay variation-bounded multicast tree if one exists. In addition to the fully distributed nature, our algorithm has two special features. First, it allows dynamical reorganizing the multicast tree in response to changes for the destination, and guarantees the minimal disruption to the multicast session. Then, a large amount of simulation has been done to show that it is better in terms of tree delay and routing success ratio as compared with other existing algorithms, and performs excellently in delay variation performance under lower time complexity, which ensures it to support the requirements of real-time multimedia communications more effectively.

References

[1] Rouskas G N, Baldine I. Multicast routing with end-to-end delay and delay variation constraints[J]. IEEE Journal on Selected Areas in Communications, 1997, 15(3): 346-356.

[2] Sheu P R, Chen S T. A fast and efficient heuristic algorithm for the delay and delay variation-bounded multicast tree problem[J]. Computer Communications, 2002, 25(8): 825-833.

[3] Yu Yanping, Qiu Peiliang. A heuristic of multicast routing with delay and delay variation constraints[J]. Journal on Communications, 2003, 24(2): 132-137. (in Chinese)

[4] Jia Z F, Varaiya P. Heuristic methods for delay constrained least cost routing using k-shortest-paths[C]. Proc. of IEEE INFOCOM'01, 2001.

[5] Bellman R E. Dynamic Programming[M]. NJ: Princeton University, 1997.

[6] Waxman B M. Routing of multipoint connections[J]. IEEE Journal on Selected Areas in Communications, 1988, 6(9): 1617-1622.

[7] Salama H F. Multicast routing for real-time communication on high-speed networks[D]. North Carolina State University, Department of Electrical and Computer Engineering, 1996.

Author

Zhu Baoping, male, was born in 1964. He graduated from Nanjing University, mathematic logic major, in 1987 and got his doctor's degree on computer science application in 2007. He is an associate professor of Nanjing University of Science & Technology and his research focuses on information security and theory. He can be reached by E-mail: zbp2068@126.com