Optimal Multicast Routing for ATM Networks

Maria C. Yuang, Y. G. Chen, and M. T. Yen

Department of Computer Science and Information Engineering
and Center for Telecommunications Research
National Chiao Tung University, Taiwan

Abstract

ATM networks are expected to efficiently provide multicast communication services such as video conferencing by means of feasible multicast routing algorithms. Several minimum–cost multicast routing algorithms with various degrees of computational complexity have been proposed. These algorithms, however, could be inappropriate for ATM networks due to their high complexities. In this paper, we initially present an optimal multicast routing algorithm, called LOPT (Load Optimal), which guarantees minimum load or a minimum number of cells to be generated throughout the network. LOPT simply corresponds to a minimum–cost multicast routing algorithm if the cost of each link is assumed to be unity. The algorithm efficiently determines the minimum–load multicast route by means of partition. Complexity analysis shows its superiority over the Balakrishnan's algorithm, one of promising multicast routing algorithms, especially when the number of destinations is much smaller than the size of the network. Experimental results further exhibit even better efficiency than its theoretical results. Moreover, based on LOPT, we further propose two variants of the algorithm for applications with different requirements. The first variant determines the multicast route satisfying the condition min[max(delay)] subject to the constraint of the minimum load. The second variant determines the minimum–load route subject to the constraint of a delay bound. The former can be applied for load–sensitive networks, whereas the latter can be applied for delay–sensitive applications.

1. Introduction

Asynchronous Transfer Mode (ATM) [1,2,10,17] has emerged as a promising transport technique for supporting multicast communication services such as video teleconferencing and message multicasting in broadband networks. To provide the multicast communication services, the design of efficient multicast routing algorithms becomes crucial. Generally for ATM networks, multicast routing can be realized on the bases of *delay optimization* and *load optimization*. Delay optimization determines a multicast route with a minimal delay from a source to multiple destinations, given a network with a delay value associated with each link. This problem actually corresponds to the *single–source–shortest–paths problem* [16] and has already been resolved.

On the other hand, load optimization determines a route incurring minimum load. The minimum–load route is defined as a route causing a minimum number of ATM cells copied and passed (i.e., a minimum number of links traversed) throughout the network to all destinations. Notice that the minimum–load multicast routing problem simply corresponds to a minimum–cost multicast routing problem with the unit–cost constraint. Consequently, the problem is then correspondent with the *Graphical Steiner Minimal Tree problem (GSMT)* [8,20] which has been proved to be NP–complete [12]. Efficient solving of the GSMT problem with the unit–cost constraint becomes the aim of our paper.

Several GSMT algorithms with various degrees of computational complexity have been proposed. The Hakimi algorithm [9] considered all possible choices for the Steiner nodes (i.e., the non–destination nodes) and for each choice computes the corresponding minimum–cost route. Levin [15] and Dreyfus and Wagner [6] employed dynamic programming techniques to determine the optimal route. Shore, Foulds and Gibbons [7,18] applied a branch–and–bound approach to this problem. The Balakrishnan and Patel algorithm [4] (referred to as B&P) generated spanning trees for a derived problem in increasing–cost order until a solution to the original problem has been inferred [11]. Aneja [3] reformulated the problem as a set–covering problem and employed several relaxation techniques. Beasley [5] proposed a branch–and–bound algorithm using Lagrangean relaxations of 0–1 linear programming reformulations for the bounds. Kou [14] proposed a Minimal–Spanning–Tree (MST)–based algorithm for approximate solutions. Wu [21] gave an implementation on the basis of the Kruskal's MST algorithm. Takahashi and Matsuyama [19] presented a heuristic method based on the Prim's MST algorithm. These algorithms, however, could be inappropriate for ATM networks due to their high complexities.

In this paper, we initially present an optimal multicast routing algorithm, called LOPT (Load Optimal), which guarantees minimum load or a minimum number of cells to be generated throughout an ATM network. The algorithm efficiently determines the minimum–load multicast route by means of partition. Complexity analysis shows its superiority over the simplified B&P algorithm with the unit–cost constraint [22], especially when the number of destinations is much smaller than the size of the network. Experimental results exhibit even better efficiency than its theoretical results.

Proceedings of the 20th Conference on Local Computer Networks (LCN '95)

Moreover, on the basis of LOPT, we further propose two variants of the algorithm for applications with different requirements. The first variant determines a multicast route satisfying the condition min[max(delay)] subject to the constraint of the minimum load. The second variant determines a minimum–load route subject to the constraint of a delay bound. The former can be applied for load–sensitive networks, whereas the latter can be applied for delay–sensitive applications.

This paper is organized as follows. Section 2 presents the minimum–load multicast routing algorithm including the complexity analysis of the algorithm. Section 3 demonstrates experimental results and comparisons of results. The two variants of the algorithm are then addressed in Section 4. Finally, conclusion remarks are given in Section 5.

2. Minimum–Load Multicast Routing Algorithm

In this section, we propose the minimum–load multicast routing algorithm, named as LOPT. The minimum–load route is defined as a route causing a minimum number of ATM cells [10] copied and passed (i.e., a minimum number of links traversed) to all destinations throughout the network. The worst–case time complexity of LOPT is also analyzed and compared with that of the simplified B&P algorithm.

2.1 The LOPT Algorithm

Generally, LOPT first determines the nodes appearing in the minimum–load multicast route by means of partition. The links establishing the route are then selected by performing a breadth–first search (BFS) from the source to the destinations through those pre–determined nodes. Networks are assumed to be undirected. In the following, LOPT is described in detail.

Given a graph (G) with a source (s) and a set of destinations (T), let Demand–nodes (*D–nodes*) represent the nodes to be connected, i.e., s and all nodes in T, and Steiner–nodes (*S–nodes*) represent the remaining nodes in G. Initially, we perform a two–step procedure to reduce the search space of determining the minimum–load route: (i) each S–node with only one incident link is removed; (ii) if D–nodes are distributed to different biconnected components [16] of G, all the S–nodes connecting these biconnected components (i.e., the articulation points) are transformed to D–nodes since these S–nodes must be included in the route.

Now we are at the stage of determining the nodes which would appear in the minimum–load route. Let R be a set of nodes which have currently been included in the minimum–load route. R is initially set with s only. R is repeatedly expanded by including all the D–nodes directly connected with the nodes in R. If R does not include all D–nodes, at least one S–node is required to connect R with the remaining D–nodes excluded from R. Thus, let ML_R be the best possible minimum load with respect to current R. ML_R can be expressed as :

$$ML_R = (|R| - 1) + 1 + (\text{number of remaining D–nodes excluded from } R),$$

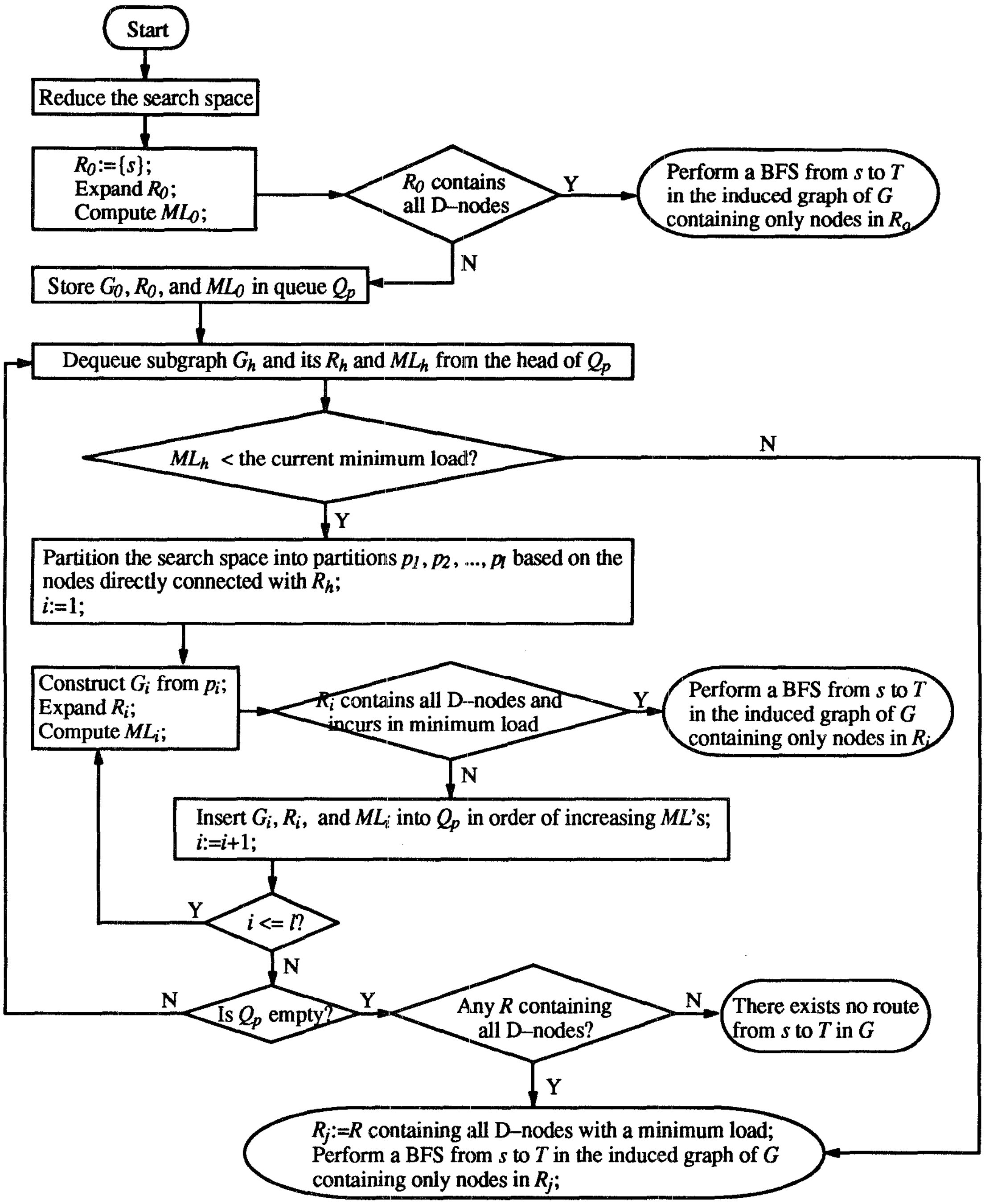
where $|R|$ represents the size of R. Now, given a set of nodes, $\{n_1, n_2, ..., n_l\}$, which represents the S–nodes directly connected with R, the search space (P) containing all possible routes can be partitioned as:

$$P = [n_1] \bigcup [\bar{n}_1 n_2] \bigcup ... \bigcup [\bar{n}_1 ... \bar{n}_{l-1} n_l] \bigcup [\bar{n}_1 ... \bar{n}_{l-1} \bar{n}_l],$$

where partition $[\bar{n}_1 \bar{n}_2 ... \bar{n}_{i-1} n_i]$ ($i = 1$ to l) represents the routes with nodes n_1 through n_{i-1} excluded, and node n_i included. If there is no S–node connected with R, and there exists at least a D–node still excluded from R, then there exists no route from s to T in G. The algorithm hence terminates in failure. Otherwise, the algorithm continues and each partition $[\bar{n}_1 \bar{n}_2 ... \bar{n}_{i-1} n_i]$, i=1 to l, is used to construct a subgraph from G with nodes n_1 to n_{i-1} removed. Notice that partition $[\bar{n}_1 \bar{n}_2 ... \bar{n}_{l-1} \bar{n}_l]$ is useless with respect to the problem solving owing to the disconnection of R from the remaining D–nodes. Node set R associated with each subgraph is then expanded with node n_i and then the D–nodes or the single S–node directly connected with R. ML_R corresponding to the newly–expanded R is then computed. These subgraphs are then repeatedly processed by means of the same partition in order of increasing ML_R's until all D–nodes are included in R with a load less than or equal to the minimum $ML_{R'}$ of any existing pending subgraph possessing R'. Upon having determined the nodes in the minimum–load route, the optimal links in the route can be easily selected by performing a BFS algorithm from s to T. The LOPT algorithm is depicted in the flowchart shown in Figure 1 .

An example of determining the minimum–load multicast route based on LOPT is shown in Figure 2. Two subgraphs, G_1 and G_2, are first generated by partitioning based on the S–nodes directly connected with R_0. R_1 is expanded by including *Node 2* and its directly connected D–node, *Node 5*. Similarly, R_2 is expanded with *Nodes 3*, *6*, and *5* included. Since ML_1 is less than ML_2, G_1 becomes the next subgraph to be processed. Three subgraphs, G_3, G_4, and G_5, are similarly generated with associated R's and ML's computed. Notice that although R_5 contains all D–nodes, it does not contribute to the minimum load owing to the existence of G_2, G_3, and G_4 with smaller ML's. The same procedure continues until G_{11} which includes all D–nodes in its R_{11} and results in the smallest minimum load of all pending subgraphs, G_6, G_7, G_8, and G_{10}. Finally, the links establishing the minimum–load route are determined by performing a BFS from source *Node 1* to destinations *Node 5* and *Node 8* in the induced graph of G through the nodes in R_{11}.

Proceedings of the 20th Conference on Local Computer Networks (LCN '95)
0742-1303/95 $10.00



Legend:

- s: the source;
- T: the set of destinations;
- G_k: subgraph k (G_0 = original graph);
- R_k: R for subgraph k;
- ML_k: ML_{R_k} for subgraph k;
- Q_p: a queue storing current pending subgraphs;

Figure 1. The LOPT algorithm.

Proceedings of the 20th Conference on Local Computer Networks (LCN '95)
0742-1303/95 $10.00 © 1995 IEEE

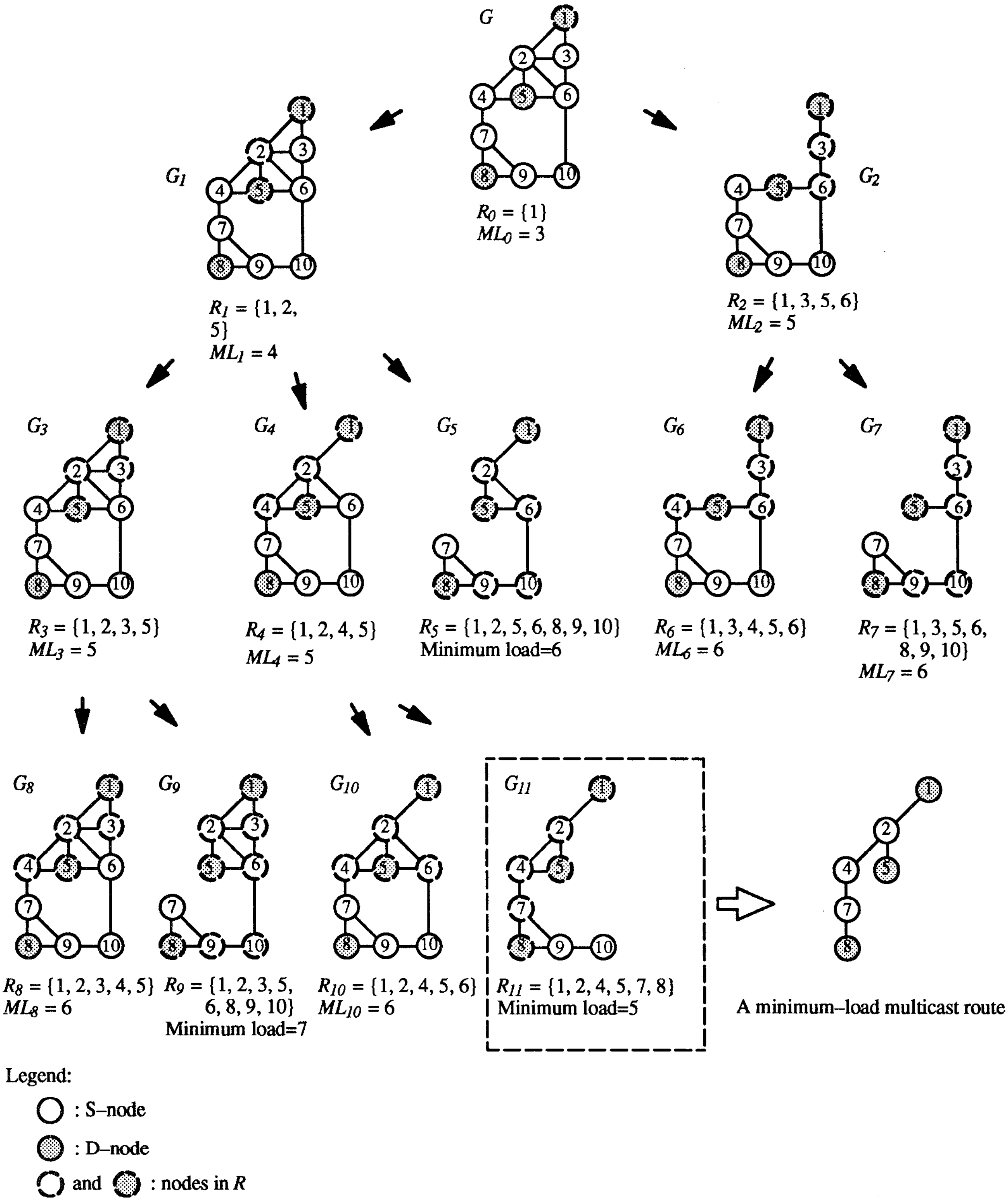


Figure 2. An example of minimum–load multicast routing based on LOPT.

Since LOPT considers all possible routes and terminates when all D–nodes has been reached with a load no higher than the best possible minimum load of the pending subgraphs, LOPT thus guarantees the minimum–load route. The formal justification of the algorithm has been presented in detail in [22].

2.2 Complexity Analysis

The time complexity of LOPT hinges on the number of S–nodes appearing in the minimum–load route, and thus the total number of subgraphs generated. Unfortunately, the total number of generated subgraphs for a general network depends on the topology of the network, the number of D–nodes, and how the D–nodes are distributed in the network. This fact renders the analysis to be considered under the worst case of a network, in which all D–nodes, except for the one located far distant from the source (s), are directly connected to s, as shown in Figure 3. In this case, a maximum number of S–nodes (or subgraphs) between s and the distant D–node are examined. We now evaluate the worst–case time complexity of LOPT by computing the total number of subgraphs generated for such network.

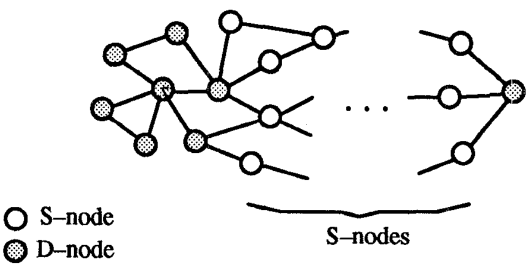


Figure 3. Network topology for worst coplexity of LOPT

Lemma 1: Given an R consisting of n nodes, there exist at most $(dn-2(n-1))$ nodes directly connected with R, where d is the highest link degree of the nodes in the network.

Proof: Since the sum of the link degrees of the nodes in R is at most dn and at least $n-1$ links is required to make R connective, the number of nodes directly connected with R is thus no greater than $(dn-2(n-1))$. ■

Theorem 1: The maximum number of subgraphs to be generated by LOPT is $O((d|D|)^m)$, where d is the highest link degree of the nodes, $|D|$ is the number of D–nodes, and m is the largest distantance (in nodes) between the source and any D–node in the network.

Proof: Given a graph (G) with a set of D–nodes to be connected in which s is the source and t is the most distant D–node. Initially, the node set R associated with G consists of $(|D|-1)$ nodes, including s and all D–nodes except for t. By Lemma 1, there are $(d(|D|-1)-2(|D|-2)) \approx d|D|$ generated subgraphs by partitioning. Among these subgraphs, the first one without excluding any S–node possesses an R including $|D|$ nodes, and subsequently generates $(d|D|-2|D|+2)$ new subgraphs. Consequently, the total number of subgraphs generated from those $d|D|$ subgraphs becomes $((d|D|-2|D|+2) + (d|D|-2|D|+1) + ... + (d-1)) \approx (d|D|)^2$. The same justification can be applied for m steps until the distant D–nodes (t) is reached. Ultimately, the total number of subgraphs generated is $(1 + d|D| + (d|D|)^2 + ... + (d|D|)^m) \approx (d|D|)^m$. ■

Notice that even though several existing minimum–cost multicast routing algorithms provided complexity analysis, it is unachievable to make a comparison between LOPT and any of them due to the employment of different parameters. To approach this, we simplified the B&P algorithm by considering the unit–cost constraint. We have also showed that the algorithm generated a maximum of $O(|S|^m)$ subgraphs [22], where $|S|$ represents the number of S–nodes in the network. Since $d|D|$ is much smaller than $|S|$ in realistic networks, LOPT thus outperforms the simplified B&P algorithm.

3. Experimental Results

We have implemented LOPT and performed experiments on random networks possessing four parameters: the network size (denoted as N), the number of D–nodes (denoted as $|D|$), the average link degree (denoted as d), and the maximum distance (denoted as m) between the source and the destinations. The experimental performance of LOPT is also compared with the theoretical performance and that of the B&P algorithm. Each experimental result was computed by averaging the results of 120 try–outs on a given random network with D–nodes randomly distributed in the network in each try–out. The variance of the link degree of each random network was set as 1.0.

Figures 4 to 7 present comparisons of the performance in terms of the four parameters defined above, respectively. In particular, Figure 4 shows that the number of subgraphs generated by LOPT moderately grows with the size of networks. Experimental results of LOPT exhibit better efficiency than its theoretical results. In Figure 5, theoretical analyses show that the numbers of subgraphs generated by LOPT and the simplified B&P algorithm increase with the number of D–nodes. Moreover, the performance of LOPT is worse than that of the simplified B&P algorithm if $d|D| > |S|$. However, experimental realistic results show, when $|D|$ crosses a threshold, the number of subgraphs generated by LOPT drastically decreases with the number of D–nodes. This is because when D–nodes are clustered, these D–nodes are rapidly included into R after partitioning. This thus speeds up the search of ultimate R with all D–nodes included.

Proceedings of the 20th Conference on Local Computer Networks (LCN '95)
0742-1303/95 $10.00 © 1995 IEEE

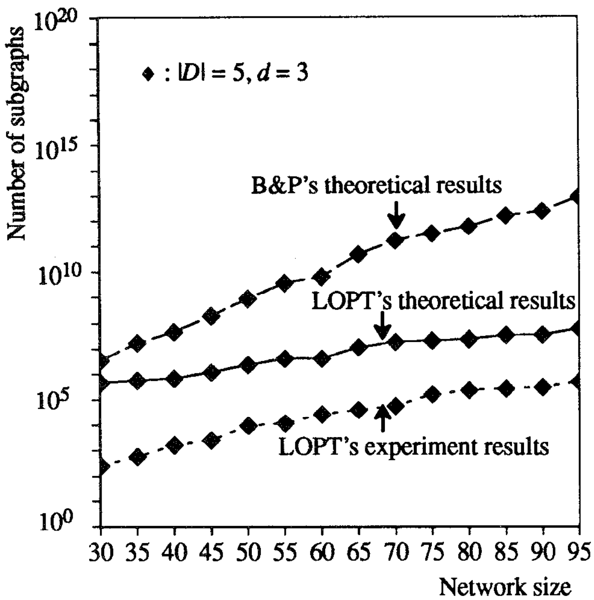


Figure 4. Performance with respect to network size

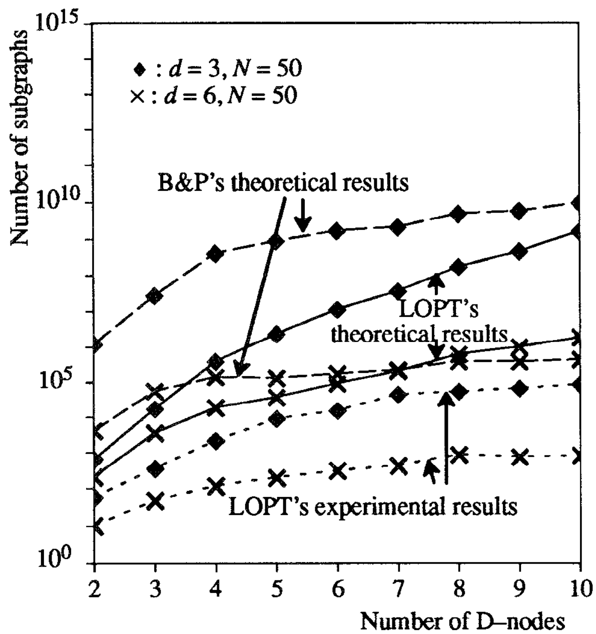


Figure 5. Performance with respect to number of D–nodes.

In Figure 6, theoretical results show that the numbers of subgraphs generated by LOPT and B&P decrease with the average link degree. In addition, experimental results show that the more complex the network is, the more rapidly all D–nodes are included into R, consequently, the smaller number of subgraphs LOPT generates. Finally, in Figure 7,

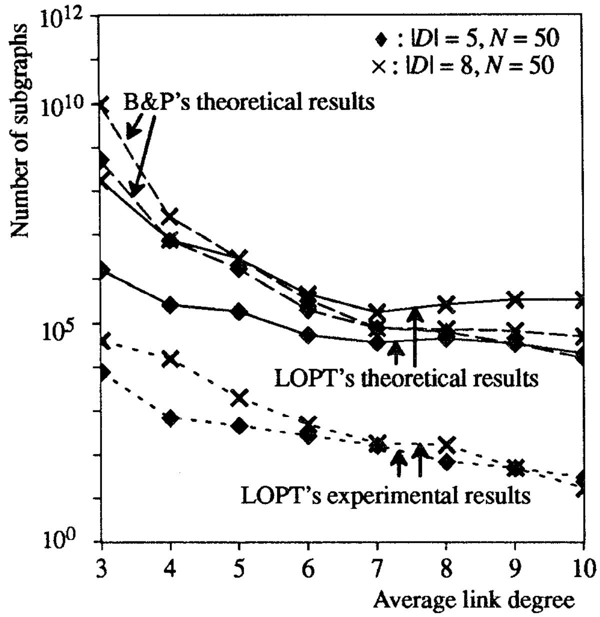


Figure 6. Performance with respect to average link degree.

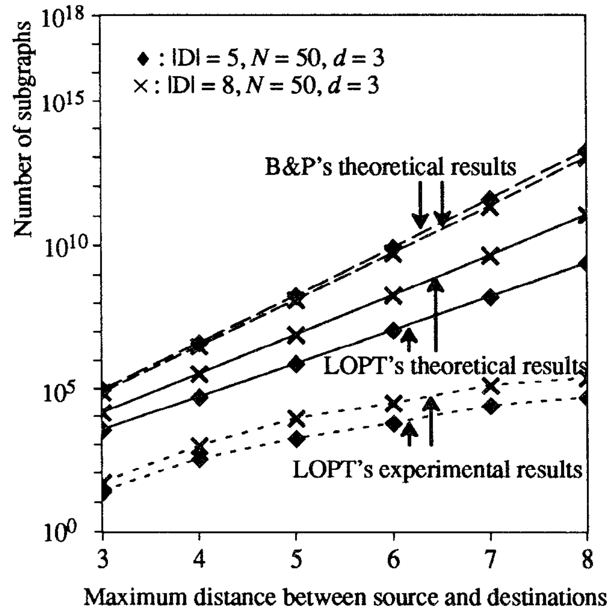


Figure 7. Performance with respect to maximum distance between source and destinations.

theoretical results show the superiority of LOPT over the simplified B&P algorithm. Moreover, experimental results of LOPT exhibit better efficiency than theoretical results.

4. Two Variants of LOPT

On the basis of the LOPT algorithm presented above, this section presents two variants of LOPT for applications with

Proceedings of the 20th Conference on Local Computer Networks (LCN '95)
0742-1303/95 $10.00 © 1995 IEEE

different requirements. The first variant (named as *MLMD*) determines the multicast route satisfying the condition min[max(delay)] subject to the constraint of the minimum load. The second variant (named as *MDML*) determines the minimum–load route subject to the constraint of a delay bound. Each algorithm is presented in detail as follows.

4.1 MLMD Algorithm

Recall that LOPT terminates immediately when a multicast route with the minimum load has been discovered. Different with LOPT, MLMD terminates when all minimum–load multicast routes have been discovered, i.e., the partition process terminates when the best possible minimum loads of all pending subgraphs are greater than the minimum load having currently been found. Among these minimum–load routes, MLMD then selects the subgraph with the minimum height representing the min[max(delay)] multicast route.

Figure 8 presents an example of determining an MLMD route for a given network. In this example, four ultimate *R*'s resulting in the minimum load are found, i.e., R_3, R_5, R_6, and R_7. After performing a BFS, four corresponding minimum–load multicast routes are determined. Since the depth (maximum delay) of these four routes are two, four, four, and four, respectively, the minimum height (or min[max(delay)]) is two. Therefore G_3 is the subgraph satisfying the min[max(delay)] condition. The MLMD route is then constructed from R_3.

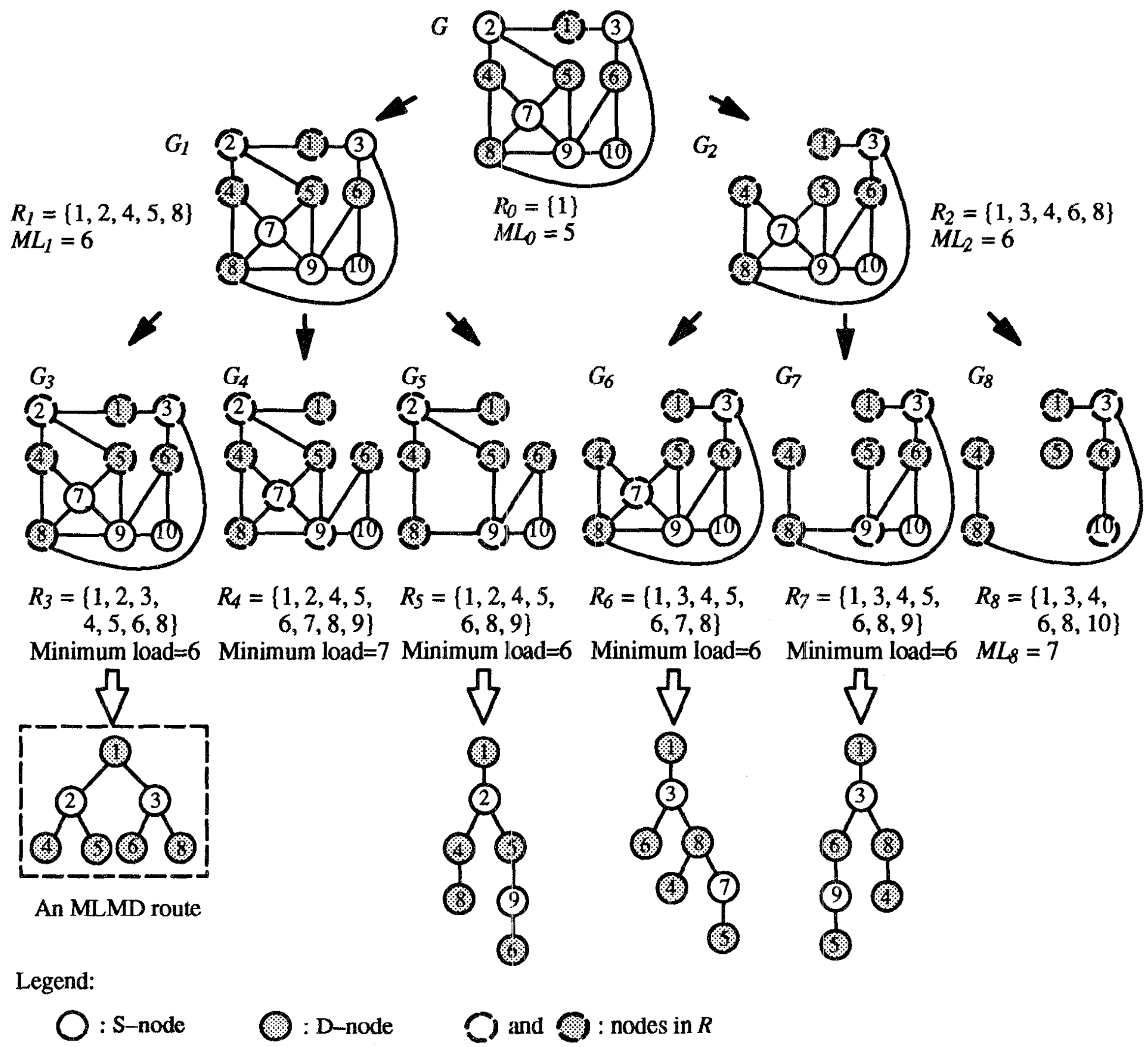


Figure 8. An example of determining an MLMD route.

Proceedings of the 20th Conference on Local Computer Networks (LCN '95)
0742-1303/95 $10.00 © 1995 IEEE

4.2 MDML Algorithm

In contrast with MLMD, MDML determines the multicast route (denoted as an MDML route) subject to the constraint of a delay bound. Initially, the delay bound is determined by performing a BFS from the source to the destinations. Let h be the height of the BFS tree. The MDML algorithm then aims at determining the minimum-load route with delay bound h. MDML initially sets multiple R's each of which consists of the nodes in a specific shortest-path route from the source to the farthest destinations. To ensure that R is expanded without violating the constraint of a delay bound, R is expanded by only considering the S-nodes resulting in delays bounded by h. Thus, by applying the same partition procedure employed in LOPT, MDML determines the minimum-load multicast route subject to the constraint of delay bound h.

Figure 9 presents an example of determining an MDML route for a given network. In this example, three initial R's, R_0, R_1, and R_2, which correspond to all possible combinations of the shortest paths from the source to the farthest destinations are considered. Since ML_1 represents the best possible minimum load, its corresponding R_1 is then processed. The search terminates when R_4 includes all D-nodes with the

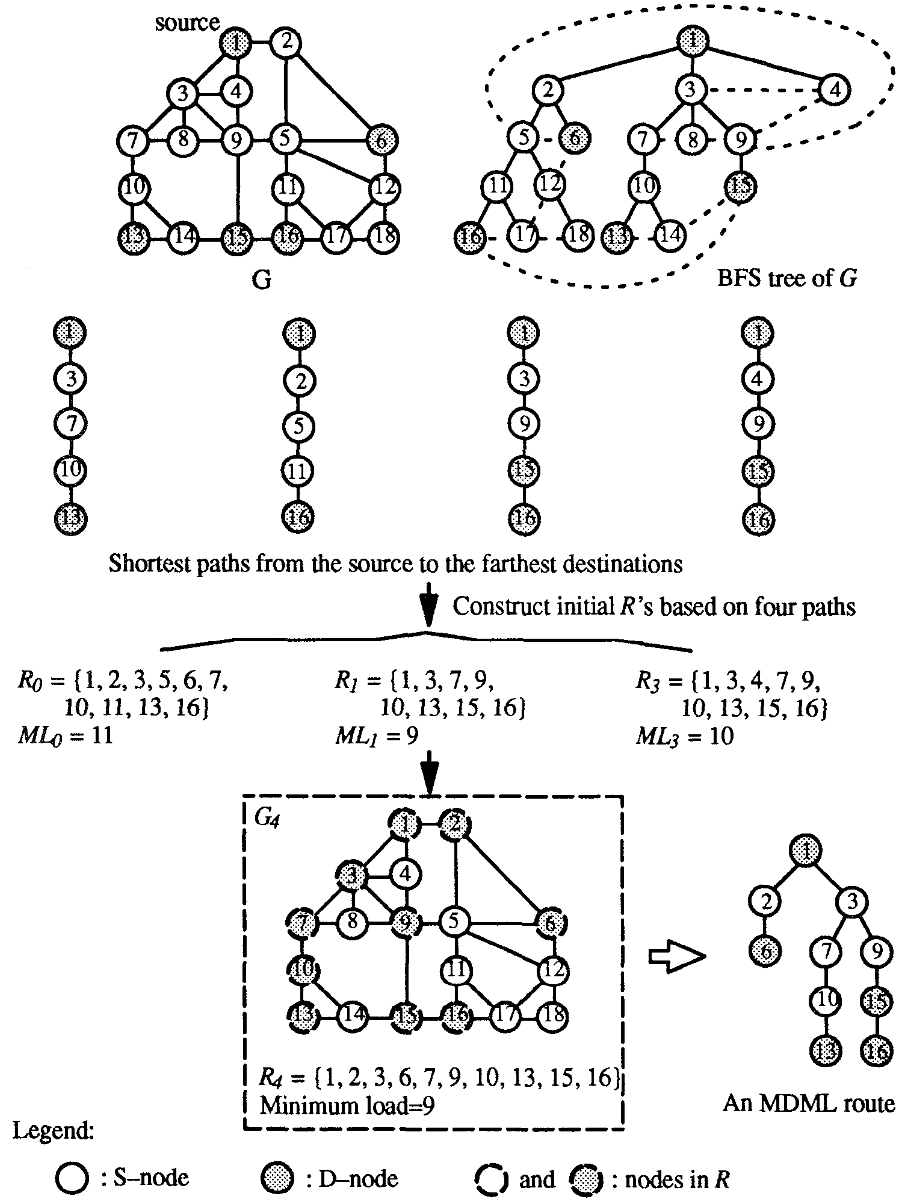


Figure 9. An example of determining an MDML route.

minimum load less than *ML*'s of all pending subgraphs. The links establishing the MDML route are then determined by performing a BFS from the source to destinations in the induced graph of G containing only the nodes in R_4.

5. Conclusions

The paper has proposed a minimum-load multicast routing algorithm which guarantees minimum load or a minimum number of cells to be generated throughout the ATM network. We demonstrated experimental and theoretical results on random networks based on four parameters (the network size, the number of D-nodes, the average link degree, and the maximum distance between the source and the destinations). Complexity analysis showed its superiority over the simplified B&P algorithm especially when the number of destinations is smaller than the size of the network. Experimental realistic results further exhibit even better efficiency than its theoretical results. This paper has also presented two variants of the algorithm for applications with different requirements. The first variant determines a multicast route satisfying condition min[max(delay)] subject to the constraint of the minimum load, and can be applied for load-sensitive networks. The second variant determines a minimum-load route subject to the constraint of a delay bound, and can be applied for delay-sensitive applications.

References

[1] 1990 CCITT Study Group XVIII Recommendation I.150, "B-ISDN Asynchronous Transfer Mode Functional Characteristics," CCITT, Geneva, 1990.

[2] CCITT SG XVIII Draft Recommendation I.150: "B-ISDN ATM functional characteristics," Geneva, May 1990.

[3] Y. P. Aneja, "An integer linear programming approach to the Steiner problem in graphs," *Networks*, vol. 10, pp. 167–178, 1980.

[4] A. Balakrishnan and N. R. Patel, "Problem reduction methods and a tree generation algorithm for the Steiner network problem," *Networks*, vol. 17, pp. 65–85, 1987.

[5] J. E. Beasley, "An algorithm for the Steiner problem in graphs," *Networks*, vol. 14, pp. 147–159, 1984.

[6] S. E. Dreyfus and R. A. Wagner, "The Steiner problem in graphs," *Networks*, vol. 1, pp. 195–207, 1972.

[7] L. R. Foulds and P. B. Gibbons, "A branch and bound approach to the Steiner problem in graphs," *Proceedings of the 14th Annual Conference of the Operational Research Society of New Zealand*, vol. 1, pp. 61–70, 1978.

[8] E. N. Gilbert and H. O. Pollak, "Steiner minimal trees," *SIAM. Journal for Applied Mathematics*, vol. 16, no. 1, pp. 1–29, 1968.

[9] S. L. Hakimi, "Steiner's problem in graphs and its implications," *Networks*, vol. 1, pp. 113–133, 1971.

[10] R. Handel et. al., *ATM Networks – Concepts, Protocols, Applications*, second Edition, Addison Wesley, 1994.

[11] F. K. Hwang and Dana S. Richards, "Steiner tree problems," *Networks*, vol. 22, pp. 55–89, 1992.

[12] R. M. Karp, *Complexity of computer computations*, Plenum Press, New York, pp. 85–103, 1972.

[13] V. P. Kompella, J. c. Pasquale, and G. C. Polyzos, "Multicasting for multimedia applications," *INFOCOM '92*, pp. 2078–2085.

[14] L. Kou, G. Markowsky, and L. Berman, "A fast algorithm for Steiner trees," *Acta Info.*, vol. 15, pp. 141–145, 1981.

[15] A. Ju. Levin, "Algorithm for the shortest connection of a group of graph vertices," *Soviet Math. Doklady*, vol. 12, pp. 1477–1481, 1971.

[16] Udi Manber, *Introduction to algorithms*, Addision Wesley, pp. 201–208, 1989.

[17] S. E. Minzer, "Broadband ISDN and Asynchronous Transfer Mode," *IEEE Communication Magazine*, vol. 27, no. 9, pp. 17–24, 1989.

[18] M. L. Shore, L. R. Foulds, and P. B. Gibbons, "An algorithm for the Steiner problem in graphs," *Networks*, vol. 12, pp. 323–333, 1982.

[19] H. Takahashi and A. Matsuyama, "An approximate solution for the Steiner problem in graphs," *Math. Jpn.*, vol. 24, pp. 573–577, 1980.

[20] P. Winter, "Steiner problem in networks: A survey," *Networks*, vol. 17, pp. 129–167, 1987.

[21] Y. F. Wu, P. Widmayer, and C. K. Wong, "A faster approximation algorithm for the Steiner problem in graphs," *Acta Info.*, vol. 23, pp. 223–229, 1986.

[22] Maria C. Yuang and M. T. Yen, "Optimal multicast routing for ATM networks," Technical Report, National Chiao Tung University, Taiwan, 1995.

Proceedings of the 20th Conference on Local Computer Networks (LCN '95)
0742-1303/95 $10.00