ELSEVIER

Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®
Journal of Network and
Computer Applications xx (2003) xxx–xxx

Journal of
NETWORK
and
COMPUTER
APPLICATIONS

www.elsevier.com/locate/jnca

A comparison of two optimal approaches for the MCOP problem

Wen-Lin Yang

Department of Information Technology, National Pingtung Institute of Commerce, No. 51, Ming-Sheng East Road, Pingtung City 900, Taiwan, ROC

Received 9 October 2002; received in revised form 20 August 2003; accepted 2 October 2003

Abstract

Providing guaranteed end-to-end quality of service (QoS) is a key issue for deploying multimedia applications on the broadband integrated services networks. To support QoS-based services, an optimal path determination problem, which concerns how to select a feasible end-to-end path to simultaneously satisfy multiple QoS constraints, has to be studied. This problem is often referred as the multiple constrained optimal path problem. Two optimal algorithms: branch-and-bound and extended Bellman–Ford algorithms (EBFA) are proposed in this paper for this NP-complete problem. A number of simulations for different network topologies were carried out for comparing performances between these two algorithms. The experimental results show that the branch-and-bound based algorithm not only outperforms the EBFA, but also is an efficient and stable method when it applies to random networks and practical networks like ANSNET.

Keywords: Branch-and-bound; Multiple constrained path problem; Quality of service; Multiple constrained path; Multiple constrained optimal path

1. Introduction

For real-time multimedia applications deployed on the broadband integrated services networks, various quality of service (QoS) requirements, such as bandwidth, delay, cost, delay jitter, packet loss rate, etc. must be supported in order to provide appropriate service quality (Almeroth and Ammar, 1996; Sitaram and Dan, 2000; Verma et al., 1998; Lorenz and Orda, 2002; Raz and Shavitt, 2000). To support such QoS-based services, the problem concerning how to select a feasible end-to-end path that simultaneously satisfies multiple QoS requirements needs to be studied (Chen and Nahrstedt, 1998; Korkmaz and Krunz,

E-mail address: wly@npic.edu.tw (W.L. Yang).

1084-8045/$ - see front matter © 2003 Published by Elsevier Ltd.
doi:10.1016/j.jnca.2003.10.003

2001; Korkmaz et al., 1999, 2002; Orda, 1999; Yuan and Liu, 2001; Guo and Matta, 2003; Youssef et al., 2002).

Considering a network with every link associated with a set of additive QoS parameters (e.g. delay and cost), it is necessary for us to study the multiple constrained path (MCP) problem, if our goal is to find a feasible path between two given nodes such that all the given QoS constraints are satisfied simultaneously (Chen and Nahrstedt, 1998; Korkmaz et al., 1999, 2002; Yuan and Liu, 2001; Yang, 2002). However, the MCP problem is only a special case of the multiple constrained optimal path (MCOP) problem (Wu et al., 2000). The latter is concerned about how to determine the least cost path among all the feasible paths satisfying the QoS constraints, assuming that each link is associated with a cost and a set of additive QoS parameters. For non-additive QoS parameters like bandwidth, a preprocessing can be taken to delete links with bandwidth less than requirement. The optimal path is then determined on the resulting network. Hence, only additive QoS parameters are considered in this study.

Both problems are known to be NP-complete (Jaffe, 1984). To cope with the NP-complete, a number of heuristics were developed for MCOP and MCP problems in the past (Chen and Nahrstedt, 1998; Korkmaz and Krunz, 2001; Korkmaz et al., 1999, 2002; Yuan and Liu, 2001; Wu et al., 2000). Most of them are for the MCP problem. The empirical results show that compared to the optimal solutions, the quality of their approximate solutions is good for networks with a small set of nodes. However, as far we know, no research about performance evaluations on optimal algorithms has been reported in literature. In this paper, a study about solving the MCOP problem based on the optimal algorithms is presented. Although the MCOP problem is NP-complete and no polynomial time algorithm exists for it, it would be an advantage to develop an optimal algorithm, which is so efficient that the optimal paths can be found easily for some practical networks with reasonable size. On the other hand, in order to evaluate the performances of heuristic algorithms more accurately (Yang, 2002; Wu et al., 2000), a good optimal algorithm is required because it can provide optimal solutions for MCOP problems with larger input-size.

Two optimal algorithms are proposed in this paper for the MCOP problem. The first algorithm is developed based on the branch-and-bound technique. It is named BB algorithm in this paper. The second method is modified from the Bellman–Ford algorithm (EBFA) presented in Yuan and Liu (2001) for the MCP problem. Based on different network topologies and different number of QoS constraints, a number of experiments are designed to study the performance of these two algorithms. Although the time complexity of the BB algorithm is $O(d^h)$ at the worst case, where h denoting the number of hops of the desired path and d representing the largest node-degree in the network, the experimental results show that our BB method not only is superior to the EBFA based method, but also is a practical method for large-scale random networks.

2. The MCOP problem

We assume that a network is modeled by a directed graph $G(V, E)$, where V is a set of nodes and E is a set of links. Each link $(u, v) \in E$ is associated with a cost cos $t(u, v)$ and k

additive QoS parameters: $w_i(u,v)$, $i = 0, 1, \ldots, (k-1)$. Given k constraints C_i, $0 \le i \le (k-1)$, and a pair of nodes S and T, which represent a source node and a destination node, respectively, the goal of our study is to find a path P from S to T such that the cost of path P is minimized and all constraints are satisfied on the path P. The problem can be mathematically formulated as follows.

$$\text{Minimize} \sum_{(u,v)\in P} \cos t(u,v) \tag{1}$$

$$\text{Subject to } W_i = \sum_{(u,v)\in P} w_i(u,v) \le C_i, \quad 0 \le i \le (k-1) \tag{2}$$

If $\cos t(u,v)$ is assumed to be 0 for all the links, the above equations can be used to describe the MCP problem. Hence, two optimal algorithms proposed in this paper can be easily extended to solve the MCP problem.

3. The BB algorithm for the MCOP problem

3.1. The construction process of a state-space tree

In order to solve the MCOP problem, a data structure called state-space tree is generated from a given network G to record all the feasible paths. Based on the state-space tree, an algorithm developed based on branch-and-bound technique is applied to search for the optimal solution. This algorithm is referred as the BB algorithm in this paper.

The state-space tree is constructed in the following ways. The source node of network G is the root of the state-space tree. For each state node S_j of the state-space tree, two labels mark it: one is the node number j in the original network G and the other one is an attribute vector Y_j. Y_j can be defined as follows:

$$Y_j = (\eta_j, W_{j,0}, \ldots, W_{j,k-1}),$$

where $W_{j,i} = \sum_{(u,v)\in P} w_i(u,v)$ and $\eta_j = \sum_{(u,v)\in P} \cos t(u,v)$, $0 \le i \le (k-1)$.

Assume that P represents the path from root to state node S_j, k is the number of QoS constraints, and $w_i(u,v)$ represents the value of QoS parameter i on link (u,v).

For any state node S_u with node-label u, a new state node S_v is created for each downstream node v, if the link $(u,v) \in E$, These new state nodes are made to be children of S_u, and they are at the same level in the state-space tree. For any state node S_u with node-label u, S_u becomes a leaf node if $W_{u,i} > C_i$ for some QoS constraint C_i. Since at least one of QoS constraints is violated on the path from root to state node S_u, branching process stops at S_u.

By applying the above branching process recursively, the entire state-space tree is then obtained for the MCOP problem. The destination node must be at some of the leaf nodes of the state-space tree. However, not all the paths from root to leaf nodes with destination-label are feasible solutions for the MCOP problem. Only the paths satisfying constraints are valid.

For example, based on the network given in Fig. 1, we assume that the nodes 0 and 5 represent the source node and destination node respectively, and the paths between these

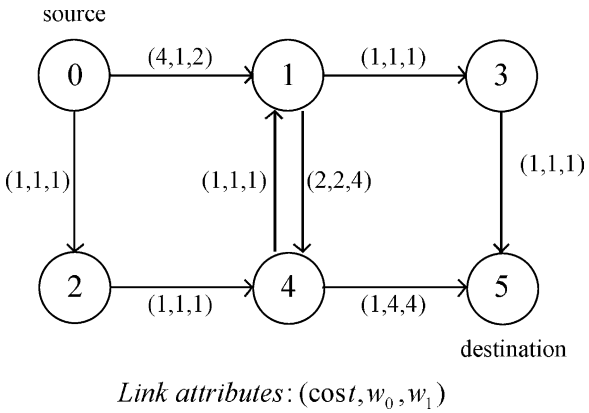

Fig. 1. An example of six-node network.

two nodes must satisfy two QoS constraints, C_0 and C_1, which are no more than 5. A state-space tree is then constructed in Fig. 2. Two feasible paths can be found in Fig. 2. One is the path between s_0 and s_7, and the other is the path between s_0 and s_{10}. Because of the smaller cost of the first path, it is the optimal solution for the MCOP problem.

To find an optimal solution, it is time consuming if considering all the feasible paths in the state space tree. In fact, by traversing only a portion of the state-space tree, the optimal solution can be determined by our best-first selection method proposed in Section 3.2. The main idea behind this method is based on the criteria for selecting next node for branching.

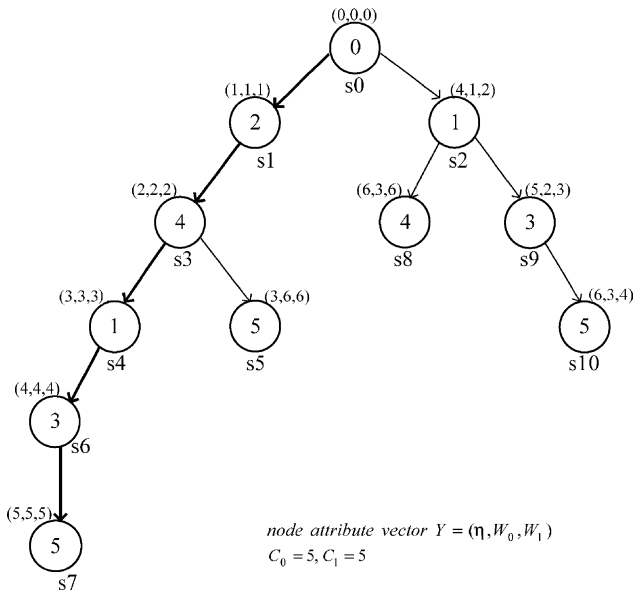

Fig. 2. State-space tree construction using the BB algorithm.

3.2. The selection process of next branching node

The search strategy of our BB algorithm is a best-first searching method. That is, during the state-space-tree constructing step, the next state node selected for branching is a node with the smallest cost.

In order to support this lowest-cost-first searching strategy, a heap is maintained to record all the state nodes that are eligible for branching. According to the attribute vector associated with each state node, it is easy to determine whether a state node is eligible for branching. Only eligible branching nodes are inserted into the heap. The next node selected for branching is at the top of the heap, since its cost value is the smallest among all the nodes in the heap. Constructing the state-space tree begins from the source node (root of the tree) and stops when the heap is empty. In addition, a current best cost 'Best_cost' is kept in the algorithm. Initially, 'Best_cost' is set at a large positive number, and it is tested for updating whenever a feasible path is found (i.e. destination is reached). That is, when the algorithm terminates, 'Best_cost' keeps the cost value of an optimal path found by the algorithm, unless no feasible solution exists for the given network. The algorithm is presented in Fig. 3.

The selection process of next branching node has two determination rules. The first rule is determined by the QoS constraints in an attribute vector. The branching-status of a state node is marked as 'NO', if any of QoS constraint is violated. After the first feasible path is computed and the value of 'Best_cost' is set, the branching-status of any new state node whose cost is greater than 'Best_cost' is also marked as 'NO'. This is the second rule for selecting a state node for branching. These two rules are shown at lines 16–20 in Fig. 3.

To illustrate our best-first BB algorithm, a numerical example is given in Fig. 2, where each state node is associated with a node number and an attribute vector. In Fig. 2, the state nodes are numbered based on their creating sequence. For example, after all state nodes at level one are generated, two state nodes s_1 and s_2 are in the heap, where the state node s_1 is at the top of the heap since the cost of s_1 is smaller than the cost of s_2. As a result, the state node s_3 is generated earlier than s_8 or s_9. The same reason also applies to the generation of state nodes s_4, s_5, and s_6. That is, for any two state nodes s_i and s_j with different father-nodes, s_i is generated earlier than s_j if $i < j$. For state nodes s_6 and s_2, since they have equal cost values and two QoS constraints are also hold for both of them, we made s_6 being selected for branching before s_2. Finally, for state node s_5, although the path stored at it reaches the destination node (node 5), this path is not valid since $w_0 > C_0$ and $w_1 > C_1$.

3.3. Time complexity

The time complexity of our BB algorithm presented in Fig. 3 is bounded by $O(d^h)$ where h denotes the number of hops of the desired path and d represents the largest node-degree in the network, since at the worst case the height of the state-space tree is at most h and the number of the children of any state node is not greater than the largest node-degree. For real networks, the node degree tends to be a small integer (Doar and Leslie, 1993). Based on our simulations given in Section 5, the optimal paths can be found in less than 1 s for large-scale random networks with 200 nodes. Hence, the actual performance of the BB algorithm is much better than what the time complexity indicates.

The BB algorithm {
(1) Initialize a node heap R;
(2) Let S denote the source node, and T denote the destination node;
(3) Let C_j = the $(j+1)th$ constraint, $0 \le j \le (k-1)$;
(4) Let $w_k(u,v)$ = the value of QoS parameter k on link (u,v); $Best\_\cos t$ = a large positive number;
(5) Let S_0 represent the source node S and be the root of the state-space tree;
(6) Add S_0 to heap R;
(7) While ($R \neq \phi$) {
(8) $S_u = remove\_top(R)$;
(9) For each node v adjacent to node u of S_u {
(10) If (v is not on the path from S_0 to u, and v has been not visited from u) {
(11) Create a new state node S_v for v based on the information stored in state node S_u of u;
(12) $S_v->branch = YES$;
(13) $S_v->path = S_u->path + (u,v)$;
(14) Let P denote the path from S_0 to v, and $\eta = \sum_{(u,v)\in P} \cos t(u,v)$;
(15) For each constraint C_k {
(16) If ($\sum_{(u,v)\in P} w_k(u,v) > C_k$) {
(17) $S_v->branch = NO$; break;
(18) } }
(19) If ($v \neq T$ and $\eta > Best\_\cos t$)
(20) $S_v->branch = NO$;
(21) If ($S_v->branch = YES$) {
(22) If ($v == T$ and $\eta < Best\_\cos t$) {
(23) $Best\_\cos t = \eta$; $S_{best} = S_v$;
(24) }
(25) If ($v \neq T$ and $\eta < Best\_\cos t$) {
(26) Add v to heap R;
(27) Make the new state node S_v to be a child-node of state node S_u;
(28) } } } }
(29) Output $Best\_\cos t$ and the optimal path stored in S_{best};
(30) }

Fig. 3. The BB algorithm.

4. The EBFA and MCOP problem

Based on the Constrained Bellman–Ford algorithm in Widyono (1994), an optimal algorithm referred as the extended Bellman–Ford algorithm (EBFA) was proposed in

```
Extended Bellman-Ford Algorithm {
  For i = 0 to |V| −1 { PATH (i) = ϕ ;}
  PATH (source) = {source} ;
  For i = 0 to |V| −1
    For each edge (u, v) ∈ E
      Relax (u, v) ;
  OPT = ϕ ;
  For each partial path p stored in PATH (destination)
    If ( (W(p)_i = Σ_{(x,y)∈P} w_i(x, y)) ≤ C_i, ∀i )
      OPT = OPT ∪ {p} ;
  If ( OPT ≠ ϕ )
    Output the path p with the minimal cost;
}
Relax (u, v) {
  For each partial path p stored in PATH (u) {
    If ( (Σ_{(x,y)∈P} w_i(x, y) + w_i(u, v)) ≤ C_i, ∀i ) {
      If ( {p + (u, v)} ∉ PATH (v) )
        PATH (v) = PATH (v) ∪ {p + (u, v)} ;
} } }
```

Fig. 4. The extended Bellman–Ford algorithm.

Yuan and Liu (2001) for solving the MCP problem. The goal of EBFA is to find a feasible path, which can satisfy k path constraints.

In this section, we present a new version of EBFA in Fig. 4 for solving the MCOP problem. The main idea behind this new version of EBFA is to store all the feasible partial paths in each node. For each link (u, v) and a partial path p stored in node u, path $p + (u, v)$ can be stored in node v if the summations of k QoS parameters along this new path satisfying their k path constraints respectively. Finally, a set of feasible paths is stored in the destination. The one with minimum cost found in the set is the optimal path. The EBFA is an exponential time algorithm, since the number of feasible partial paths stored in each node may grow exponentially with respect to the number of nodes and edges in the network.

5. Experimental results

In this section, we have several sets of experiments on the BB and EBFA algorithms for solving the MCOP problem. The purpose of this study is to compare executing performance between these two algorithms. All the experiments of this study are done with the following experimental parameters: PIII 866 MHz CPU, 512 MB RAM, Linux OS, and programs are developed by C++.

5.1. Random network generation

For the experiments conducted in this section, a procedure is needed for generating random networks, which can be modeled by directed graphs. The procedure begins with generating a random undirected graph based on the following equation:

$$P(\{u,v\}) = \beta \exp \frac{-d(u,v)}{L\alpha}, \tag{3}$$

where $P(\{u,v\})$ is the probability for creating a link between nodes u and v, and $d(u,v)$ is the distance of nodes u and v, and L is the maximum distance of any two nodes in the graph, and α and β are two positive and less than one numbers (Waxman, 1988). However, one problem with this model, it is that the number of degrees of each node is increased, as the number of nodes is increased. However, in real network, the number of degrees of each node tends to be a small value. Hence, a modified model reported in Doar and Leslie (1993) is used to generate a random undirected graph with a small node degree. The modification implemented is to scale $P(\{u,v\})$ by a factor kg/n, where k is an empirical parameter, g is the mean degree and n is the number of nodes in the graph. In this study, the mean degree of nodes in the network is set to be around 6. After an undirected graph is created, we transform it to be a directed graph by making each undirected edge to be two directed-links pointing two different end-nodes.

5.2. Performance comparisons

Three network topologies: mesh, random, and ANSNET, are used in experiments for comparing performance between the BB and EBFA algorithms. For a given network topology, we can generate different network configurations by assigning any link j a set of values for QoS parameters $w_{j,i}$ and cos t_j, where $0 \leq w_{j,i} \leq 100$ and $0 \leq \cos t_j \leq 100$. The QoS constraints C_i are then set to a value such that at least one optimal path exists for each network configuration tested in experiments. The 'success rate' is said to be 1 for these networks.

(a) *Mesh networks*

In Table 1, a set of benchmarks for mesh networks with two QoS constraints is simulated. The source and destination nodes are assumed to be two nodes located on two end-points on the longest diagonal line of the mesh. Thus, the number of hops for any path between these two nodes is at least $2 \times (n-1)$ for a $n \times n$ mesh. Let $\text{hops}_{\min}$ represents $2 \times (n-1)$.

For each QoS parameter $w_{j,i}$ on any link j, its value is randomly selected from 0 to 100. As a result, the average value of each QoS parameter on a link is 50. Hence, for the first five rows in Table 1, the constraint C_i is then set based on the equation: $\text{hops}_{\min} * 50$. For last three rows in Table 1, the network QoS constraints are made tighter by setting a constraint C_i on each link based on the equation: $(\text{hops}_{\min} * 50) * 80\%$. As for w_0 and w_1 in the third column, they represent the summations of QoS requirements on the links along the optimal path.

For a mesh topology with the same source and destination nodes, the running time in Table 1 is an average running time based on 10 different network configurations.

Table 1
Performance comparisons between EBFA and BB algorithms for meshes with two QoS parameters

Nodes	C_0/C_1	W_0/W_1	Optimal cost	BB cpu time (s)	EBFA cpu time (s)
36	500/500	457/453	377.7	0.01	0.33
49	600/600	529/532	373.9	0.03	12.01
64	700/700	609/649	455.0	1.48	1455.7
81	800/800	747/756	488.7	18.73	n/a
100	900/900	766/837	553.9	1511.7	n/a
64	560/560	526/535	549.5	0.29	8.26
81	640/640	619/621	677.2	10.31	137.41
100	720/720	684/688	720.8	950.62	n/a

The number of nodes in a mesh is $n \times n$, where $6 \leq n \leq 10$; CPU executing time is marked as 'n/a' if it is over 1 h; Each running time is an average value of 10 different configurations; For any link j, $0 \leq w_{j,0} \leq 100$, $0 \leq w_{j,1} \leq 100$ and $0 \leq \cos t_j \leq 100$.

Obviously, executing performance of the BB algorithm is much more efficient than the EBFA algorithm for all cases tested. When the mesh size is no more than 64 and the number of hops of the optimal path between source and destination nodes is at least 14, the BB algorithm can give outstanding performance. Hence, the executing performance of our branch-and-bound algorithm is greatly affected by the number of hops of the optimal path. Hence, our BB algorithm is an efficient method for networks where the number of hops between two end-points is less than 15.

For the simulations listed in the last three rows in Table 1, the running time is decreased for these two methods when the value of each C_i is decreased. This result is from the fact that the number of feasible partial paths found in the searching space is decreased when the value of QoS constraint is reduced.

In Table 2, a set of simulations on three QoS constraints is given. For the EBFA algorithm, the running time is decreased as the number of QoS constraints is increased. For example, applying EBFA algorithm to solve the 64-node mesh network based on the same value on C_i, the running time is 1445.7 s in Table 1, but it is decreased to 766.2 s in Table 2. However, the phenomenon is not so apparent for the BB algorithm except for the 100-node mesh.

Table 2
Performance comparisons between EBFA and BB algorithms for mesh networks with three QoS parameters

Nodes	$C_0/C_1/C_2$	$w_0/w_1/w_2$	Optimal cost	BB cpu time (s)	EBFA cpu time (s)
36	500/500/500	436/439/436	386.9	0.01	0.27
49	600/600/600	573/561/542	468.2	0.10	8.91
64	700/700/700	621/645/623	492.9	3.33	766.2
81	800/800/800	745/709/731	533.2	33.5	n/a
100	900/900/900	785/768/798	582.3	562.9	n/a

The number of nodes in a mesh network is $n \times n$, where $6 \leq n \leq 10$; CPU executing time is marked as 'n/a' if it is over 1 h; Each CPU time is an average value of 10 different configurations; For any link j, $0 \leq w_{j,0} \leq 100$, $0 \leq w_{j,1} \leq 100$, $0 \leq w_{j,2} \leq 100$ and $0 \leq \cos t_j \leq 100$.

Table 3
Performance comparisons between EBFA and BB algorithms for random networks with two QoS parameters

Nodes	Config.	Mean degree	Mean hops	BB cpu time (s)	EBFA cpu time (s)
100	#1	5.72	3.45	0.38	748.3
	#2	6.02	3.5	0.37	2331
	#3	5.52	3.7	0.25	1273
	#4	5.3	3.15	0.05	430.8
	#5	5.42	3.5	0.14	267.9
200	#1	4.81	4.15	0.18	155
	#2	6.32	3.65	0.35	n/a
	#3	5.99	4.00	0.51	n/a
	#4	5.3	4.25	0.29	1104
	#5	5.52	3.6	0.39	1982

Random networks are generated with $\alpha = 0.2$, $\beta = 0.25$; CPU executing time is marked as 'n/a' if it is over 1 h; each running time is an aggregate value of 20 runs, and each run has different source and destination; $C_0 = 300$, and $C_1 = 300$; For any link j, $0 \leq w_{j,0} \leq 100$, $0 \leq w_{j,1} \leq 100$, and $0 \leq \cos t_j \leq 100$.

(b) *Random networks*

Two random networks with 100 and 200 nodes respectively are tested in this set of simulations. For each network, there are five configurations generated in Table 3. In addition, for each configuration, 20 distinct pairs of source and destination nodes are selected randomly for test. Hence, the running times shown in last two columns are the aggregate values of 20 runs. As for the data of 'mean-hops' and 'mean-degree' listed in Table 3, the 'mean-hops' represents the average number of hops for 20 optimal paths computed for each configuration, while the 'max-hops/min-hops' represents the largest and smallest numbers of hops of the optimal paths generated in the 20 runs. For two QoS constraints C_0 and C_1, they are set to be 300 such that the 'success rate' is one for all runs.

The experimental data shows that the performance of our BB algorithm is much more efficient than EBFA algorithm for large-scale networks. The outstanding performance of branch-and-bound algorithm is due to the maximum number of hops of optimal paths is not greater than 7 in Table 3.

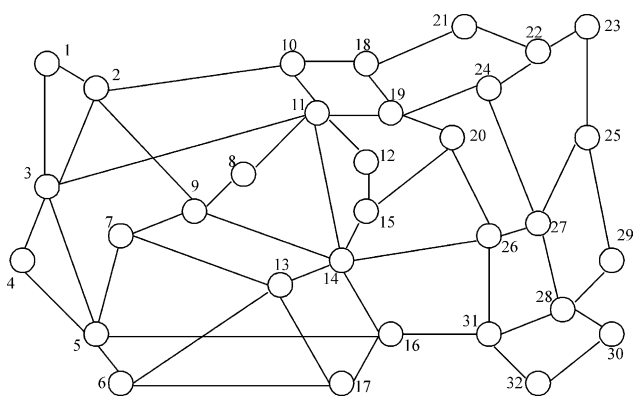

Fig. 5. A network topology.

Table 4
Performance comparisons between EBFA and BB algorithms for ANSNET with two QoS parameters

Nodes	Config.	Mean degree	Mean hops	Max. hops /min. hops	BB cpu time (s)	EBFA cpu time (s)
32	#1	3.38	3.45	6/1	0.01	1.2
	#2	3.38	3.2	6/1	0.01	0.84
	#3	3.38	3.4	6/1	0.01	0.61
	#4	3.38	3.85	8/1	0.01	1.89
	#5	3.38	3.45	6/1	0.01	0.57

Each running time is an aggregate value of 20 runs, and each run has different source and destination; BB denotes branch-and-bound algorithm; $C_0 = 400$, and $C_1 = 400$; For any link j, $0 \leq w_{j,0} \leq 100$, $0 \leq w_{j,1} \leq 100$, and $0 \leq \cos t_j \leq 100$.

(c) *ANSNET*

A network topology shown in Fig. 5 is modified from ANSNET (Comer, 1995), which was studied in Chen and Nahrstedt (1998) and Korkmaz et al. (1999). With the same simulation environment described in (b), a set of simulation results is shown in Table 4. The experimental data shows again that on the average our BB algorithm is around 100 times faster than the EBFA method for this small-scale real network.

6. Conclusions

Two optimal algorithms, BB and EBFA, are presented for the MCOP problem in this paper. In spite of the time complexity of BB algorithm is $O(d^h)$ at the worst case, the simulation results show that it not only outperforms the EBFA method, but also is an efficient method when it applies to random networks and practical networks like ANSNET. From simulations carried out for the mesh networks, it is observed that the executing performance of our BB algorithm is greatly affected by the number of hops of the optimal path. Our BB algorithm is efficient only for networks where the number of hops of the desired MCOP path is less than 15.

Acknowledgements

This work was supported by the National Science Council, R.O.C. (NSC-91-2213-E-251-001).

References

Almeroth KC, Ammar MH. The use of multicast delivery to provide a scalable and interactive video-on-demand service. IEEE J Sel Areas Commun 1996;14(6):1110–22.

Chen S, Nahrstedt K. On finding multi-constrained paths. Proceedings of the ICC'98 Conference, IEEE; 1998. p. 874–979.

Comer DE, Internetworking with TCP/IP, vol. I. New Jersey: Prentice Hall; 1995.

Doar M, Leslie I. How bad is naïve multicast routing. Proceedings of the IEEE INFOCOM Conference; 1993. p. 82–9.

Jaffe JM. Algorithms for finding paths with multiple constraints. Networks 1984;14:95–116.

Korkmaz T, Krunz M. Multi-constrained optimal path selection. Proceedings of the IEEE INFOCOM Conference, IEEE; 2001. p. 834–43.

Korkmaz T, Krunz M. A randomized algorithm for finding a path subject to multiple QoS constraints. Proceedings of the IEEE Global Telecommunications Conference, IEEE; 1999. p. 1694–8.

Korkmaz T, Krunz M, Tragoudas S. An efficient algorithm for finding a path subject to two additive constraints. Comput Commun 2002;25:225–38.

Orda A. Routing with end-to-end QoS guarantees in broadband networks. IEEE/ACM Transact Networking 1999;7(3):365–74.

Sitaram D, Dan A. Multimedia servers. Los Altos, CA: Morgan Kaufmann; 2000.

Verma S, Pankaj RK, Leon-Garica A. QoS based multicast routing algorithms for real time applications. Perform Eval 1998;34:273–94.

Waxman BM. Routing of multipoint connections. IEEE J Sel Areas Commun 1988;6(9):1617–22.

Widyono R. The design and evaluation algorithm for real-time channels, TR-94-024. UC Berkeley: International Computer Science Institute; 1994.

Yuan X, Liu X. Heuristic algorithms for multi-constrained quality of service routing. Proceedings of the IEEE INFOCOM Conference, IEEE; 2001. p. 844–53.

Lorenz DH, Orda A. Optimal partition of QoS requirements on unicast paths and multicast trees. IEEE/ACM Transact Networking 2002;10(1):102–14.

Raz D, Shavitt Y. Optimal partition of QoS requirements with discrete cost functions. IEEE J Sel Areas Commun 2000;18(12):2593–602.

Yang W-L. Exact and heuristic algorithms for multi-constrained path selection problem. Advances in multimedia information processing-PCM2002. Lecture notes in computer science, vol. 2532. Berlin: Springer; 2002. p. 952–9.

Wu J-W, Hwang R-H, Lu H-I. Multicast routing with multiple QoS constraints in ATM networks. Inf Sci 2000; 124:29–57.

Guo L, Matta I. Search space reduction in QoS routing. Comput Networks 2003;41:73–88.

Youssef H, Al-Mulhem A, Sait SM, Tahir MA. QoS-driven multicast tree generation using Tabu search. Comput Commun 2002;25:1140–9.

Wen-Lin Yang received the PhD degree in Computer Science from the Pennsylvania State University, University Park, in 1993. He is an Associate Professor in the Department of Information Technology at the National Ping-Tung Institute of Commerce, Taiwan, ROC. His primary research interests are in the areas of multimedia communication, network optimization and computer-aided design.