A Heuristic for Multi-Constrained Multicast Routing

Gang Feng

Department of Electrical Engineering
University of Wisconsin, Platteville, WI 53818, Email: fengg@uwplatt.edu.

Abstract— **Opposed to the situation that the constrained minimum Steiner tree (CMST) problem has been attracting much attention in quality of service (QoS) routing area, little work has been done on multicast routing subject to multiple additive constraints even though the corresponding applications are obvious. In this paper, we propose a heuristic HMCMC to solve this problem. The basic idea of HMCMC is to construct the multicast tree step by step, which is done essentially based on the latest research results on multi-constrained unicast routing. Computer simulations demonstrate that the proposed heuristic can find a feasible multicast tree with a fairly high probability if there is one.**

I. Introduction

Delivering only one copy of information on links shared by multiple paths to different destinations, multicast routing makes a good example for the efficient utilization of network resources. Initial research on multicast routing mainly focused on the problem of finding a minimum Steiner tree (MST) [14], which has proved to be NP-complete [6]; major efforts were then dedicated to developing efficient heuristics that can produce a low-cost tree in reasonable time complexity [9]. The protocols based on these heuristics cannot provide additional quality of service (QoS) guarantees, and thus are called *QoS-oblivious protocols* [3]. In recent years, however, many newly emerging multimedia applications have very stringent QoS requirements, necessitating great research efforts for developing *QoS-sensitive protocols*.

Most prior research on QoS-aware multicast routing is limited to the constrained minimum Steiner tree (CMST) problem, i.e., to find a delay-constrained multicast tree such that the delay between the source and each destination meets certain delay upper bound, and the total cost of the tree is minimal. Heuristics proposed to solve this NP-complete problem range from those that have extremely low cost solutions [17] to those with very low time complexities [12]. There are also algorithms that have both excellent cost performance and time performance as well [4].

Opposed to the great progress made to solve the CMST problem, little work has been done on multicast routing subject to multiple constraints, even though the corresponding applications are obvious like the multi-constrained unicast routing. Probably as the only previous work, Kuipers and Mieghem [10] proposed a Multicast Adaptive Multiple Constraints Routing Algorithm (MAMCRA) to solve this problem, but unfortunately the performance of this algorithm is unknown (there is neither theoretical proof nor simulations to evaluate the performance of MAMCRA in [10]).

In this paper, we propose a brand new heuristic HMCMC to solve the multi-constrained multicast QoS routing problem. The basic idea of our heuristic is to construct the multicast tree step by step, which is done essentially based on the latest research results on multi-constrained unicast routing [8], [11], [13]. Simulations indicate that the proposed heuristic, even though simple (and therefore responsive), can find a feasible multicast tree with a fairly high probability if there is one.

In the following, we first give formal definitions of the multicast constrained multicast routing problem, i.e., the multi-constrained minimum Steiner tree (MCMST) problem, in Section II. We then elaborate the motivation of our work and review the related work in Section III. Heuristic HMCMC is discussed at length in Section IV, and in Section V the performance of HMCMC is investigated through a large number of computer simulations. Section VI concludes the paper.

This work was partially supported by the SAIF grant of University of Wisconsin, Platteville.

II. Notation and Problem Definition

A *network* is represented by a directed graph $G(V, E)$, where V is the set of nodes, and E is the set of links. Associated with each link e there are J non-negative weights $w_j(e), j = 1, 2, \cdots, J$ and a cost $c(e)$.

A path is a sequence of non-repeated nodes $\mathbf{p} = (v_1, v_2, \cdots, v_k)$ such that for a given $1 \le i < k$ there exists a link from v_i to v_{i+1}, i.e., $(v_i, v_{i+1}) \in E$. The notation $e \in \mathbf{p}$ means that path $\mathbf{p}$ passes through link e. The w_j-weight and cost of path $\mathbf{p}$ are given by

$$w_j(\mathbf{p}) = \sum_{e \in \mathbf{p}} w_j(e) \text{ and } c(\mathbf{p}) = \sum_{e \in \mathbf{p}} c(e),$$

respectively.

Given a multicast tree T spanning a source s and a set of destinations D, let $\mathbf{p}_T(s, v)$ denote the path on T from s to destination $v \in D$. The cost of the multicast tree T is defined by $c(T) = \sum_{e \in T} c(e)$. The upper bound of destination v corresponding to weight w_j is denoted by Δ_j^v.

Definition 1—MCMST Problem: Given a source s, a set of destinations D, a set of upper bounds $\Delta_j^v, j = 1, 2, \cdots, J$ for destination $v \in D$, the MCMST problem needs to find a tree T^* spanning $D \cup \{s\}$ such that

(i) $w_j(\mathbf{p}_{T^*}(s, v)) \le \Delta_j^v, \forall v \in D, j = 1, 2, \cdots, J$, and
(ii) $c(T^*) \le c(T)$ for any tree T that satisfies (i).

If a tree T only satisfies (i) in the above definition, it is called a *feasible multicast tree* (or *feasible solution*).

We also give a formal definition of the multi-constrained path (MCP) problem [1] since the proposed heuristic searches for a feasible solution to the MCMST problem by converting it to multiple MCP unicast routing problems, which will be elaborated later.

Definition 2—MCP Problem: Given a routing request between a source s, a destination t, a set of upper bounds $\Delta_j^t, j = 1, 2, \cdots, J$, the MCP problem is to find a path $\mathbf{p}$ between s and t such that $w_j(\mathbf{p}) \leq \Delta_j^t, \forall j = 1, 2, \cdots, J$.

III. Motivation and Related Work

The heuristic to be proposed is motivated by the recent progress of research on the CMST and MCP problems. As briefly mentioned in Section I, various heuristics have been proposed to solve the CMST problem in the past few years, and some of them have been demonstrated to have excellent cost performance as well as reasonably low time complexities. For instance, the bounded shortest multicast algorithm (BSMA) proposed by Zhu *et al.* has proved to be capable of locating a feasible tree with an extremely low cost. Even though the original implementation of BSMA was very time consuming, an alternative implementation recently proposed by Feng *et al.* [4] shows the time complexity can be considerably reduced without sacrificing the cost performance. With the alternative implementation, it runs a lot faster than many renowned heuristics such as KPP [7], CAO [16], etc., which were demonstrated to have worse cost performance yet lower time complexities than the original implementation of BSMA [14].

BSMA [17] solves the CMST problem by starting with a feasible solution and then gradually improving the solution. It first runs Dijkstra's shortest path algorithm [2] to find a minimum-delay tree as the initial solution. Then, certain type of partial path on the tree (called "super-edge" in [17]) is replaced by another path if the cost of the resulting tree is lower. This procedure repeats until the cost cannot be reduced any more.

Obviously the basic idea of BSMA can be used to solve the MCMST problem as long as we can find an initial feasible tree. However, unlike the case for the CMST problem, to find a feasible solution to the MCMST problem is NP-complete. In view of this, we concentrate our attention on how to find an initial feasible solution to the MCMST problem in this paper. Our basic idea is to construct a partial feasible tree spanning the source and a subset of destinations, and then join the remaining destinations to the partial tree through a feasible path. The whole procedure can be completed by repeatedly running a heuristic for the MCP unicast routing problem.

Like the CMST problem, the MCP problem has been attracting a great deal of attention in recent years due to the fact that there are many QoS routing problems in which the search for a feasible path to the MCP problem is a fundamental step. For such reason, a slight change of the performance of a MCP heuristic may have a significant impact on the performance of any higher-level algorithm. Currently, almost all known heuristics demonstrated to have good performance for the MCP problems [8], [11] originated from the pioneering work conducted by Neve and Mieghem [13], in which they proposed a tunable accuracy multiple constraints routing algorithm (TAMCRA). The essential idea of TAMCRA is to use a modified Dijkstra's algorithm to search for a feasible path by determining the predecessor of a node based on a nonlinear length function given by

$$g(\mathbf{p}) = \max\left\{\frac{w_j(\mathbf{p})}{\Delta_j^t}, j = 1, 2, \cdots, J\right\}.$$

The probability that TAMCRA can find a feasible solution is tunable by adjusting the number of (k) paths stored in the queue of each intermediate node. Simulations indicate that the probability of finding feasible solutions is already satisfactorily high even when $k = 2$ [11].

Korkmaz *et al.* proposed a heuristic H_MCOP [8], in which the searching process is finished in two steps. A modified Dijkstra's algorithm is first executed in reverse direction with a linear length function to find a *post-path* from an intermediate node to destination t; then, it is run in forward direction with the nonlinear length function $g(\cdot)$ to find a *pre-path* from source s to an intermediate node. A feasible solution can thus be retuned if at any intermediate node the *pre-path* joined with the *post-path* satisfies all constraints.

Even though the question which of these two heuristics, TAMCRA and H_MCOP, is better is still controversial (see [8] and [11]), the overall performance of both of the two heuristics is of very high standard. In the following Section, we first describe a modified heuristic for the MCP problem, based on which the heuristic HMCMC for the multi-constrained multicast routing problem is discussed in detail.

IV. Heuristic HMCMC

A. Heuristic H_MCP

Heuristic H_MCOP was originally proposed for solving the multi-constrained optimal path (MCOP) problem, i.e., to find a path with the minimal cost that satisfies all constraints. Even though it can be directly used to solve a MCP problem by skipping all codes regarding the cost c, it does not achieve the best performance due to the particularity of the MCP problem, for which we are concerned with feasible solutions.

Considering the above reasons, we present a heuristic H_MCP for the MCP problem, which is modified from H_MCOP. As shown by the top-level description in Fig. 1, H_MCP first runs Dijkstra's algorithm in reverse direction to find the shortest path with respect to $g(\cdot)$ from any other node to destination t (line 1). The path from s to t found by Reverse_Dijkstra will be returned if it is a feasible path (lines 2 - 3). Otherwise, Dijkstra's algorithm is executed in the forward direction to search for a feasible solution, which may be a combination of the *pre-path* (from s to a node u) found by Look_Ahead_Dijkstra and the *post-path* (from u to t) found by Reverse_Dijkstra (lines 4 - 5). However, even if no feasible solution can be found, the path found by Look_Ahead_Dijkstra will still be returned as the best available path. The reason for returning the best available tree will be explained later.

Figs. 2 and 3 are the relaxation procedures for subroutines Reverse_Dijkstra and Look_Ahead_Dijkstra, respectively. Fig. 4 is a preference rule which arbitrates whether a node u should become the predecessor of a node v. With such preference rule, a feasible solution takes precedence over all other options.

The notations used in H_MCP are defined as follows. $r[u]$ denotes the length w.r.t. $g(\cdot)$ of the *post-path* from u to t found by Reverse_Dijkstra, while $f[u]$ is the length of the *pre-path* from s to u found by Look_Ahead_Dijkstra. Labels $R_j[u]$, $j = 1, 2, \cdots, J$ represent the individual accumulated link weights along the *post-path* with the predecessor of u stored in $\pi_r[u]$. Correspondingly, labels $G_j[u]$, $j = 1, 2, \cdots, J$ represent the individual accumulated link weights along the *pre-path* with the predecessor of u stored in $\pi_g[u]$.

When comparing H_MCP with H_MCOP, one should notice that there are two major modifications in addition to that all codes for processing cost c have been ignored and that the best available path will be returned even if there is no feasible solution. First, Reverse_Dijkstra in H_MCP uses the same nonlinear length function as the one used in Look_Ahead_Dijkstra, unlike in H_MCOP a linear function is used. Second, in Look_Ahead_Dijkstra, the length function $f[\cdot]$ is evaluated only for a *pre-path* from s to u instead of a complete path from s to t. The reason for making these two modifications is due to our recent finding that such modifications can increase the probability of finding a feasible solution [5].

H_MCP($G(V, E), s, t, w_j, \Delta_j^t, j = 1, 2, \cdots, J$)
Reverse_Dijkstra ($G(V, E), t, w_j, j = 1, 2, \cdots, J$)
if ($R_j[s] \leq \Delta_j^t, \forall j = 1, 2, \cdots, J$) **then**
    return the path found by Reverse_Dijkstra
Look_Ahead_Dijkstra($G(V, E), s, w_j, \Delta_j^t, j = 1, 2, \cdots, J$)
return the path found by Look_Ahead_Dijkstra
return NULL

Fig. 1. The heuristic algorithm H_MCP for the MCP problem

Reverse_Dijkstra_Relax(u, v)
if $r[u] > \max\left\{\frac{R_j[v]+w_j(u,v)}{\Delta_j^t}, j = 1, 2, \cdots, J\right\}$ **then**
    $r[u] = \max\left\{\frac{R_j[v]+w_j(u,v)}{\Delta_j^t}, j = 1, 2, \cdots, J\right\}$
    $R_j[u] = R_j[v] + w_j(u, v)$
    $\pi_r[v] = u$

Fig. 2. The relaxation procedure of subroutine Reverse_Dijkstra

Look_Ahead_Dijkstra_Relax(u, v)
Let tmp be a temporary node
$f[tmp] = \max\left\{\frac{G_j[u]+w_j(u,v)}{\Delta_j^t}, j = 1, 2 \cdots J\right\}$
$G_j[tmp] = G_j[u] + w_j(u, v)$ for $j = 1, 2, \cdots, J$
$R_j[tmp] = R_j[v]$ for $j = 1, 2, \cdots, J$
if (Prefer_the_best(tmp, v) $= tmp$) **then**
    $f[v] = f[tmp]$
    $G_j[v] = G_j[tmp]$ for $j = 1, 2, \ldots, J$
    $\pi_g[v] = u$

Fig. 3. The relaxation procedure of subroutine Look_Ahead_Dijkstra

B. Heuristic HMCMC

Heuristic HMCMC searches for a feasible solution to the MCMST problem by first finding a feasible partial tree that spans the source and some of the destinations, and builds up the tree by joining the remaining destinations through the paths found by H_MCP. As depicted in Fig. 5, HMCMC first finds an initial tree T_1, which will be returned if it is a feasible solution (lines 1 - 3). Otherwise, the path on T_1 from s to each destination node is checked for feasibility. For any node $v \in D$, if the corresponding path $\mathbf{p}_{T_1}(s, v)$ satisfies all constraints imposed on node v, $\mathbf{p}_{T_1}(s, v)$ is included in T_2 as a component of the partial feasible tree, and the incoming links to all nodes on path $\mathbf{p}_{T_1}(s, v)$ except those links that form the path are pruned (lines 4 - 9). On the contrary, for any node $v \in D$, if the corresponding path $\mathbf{p}_{T_1}(s, v)$ does not meet all the constraints, a new path $\mathbf{p}$ between s and v found by H_MCP is included in T_2, followed by the same pruning operation (lines 10 - 16). As a result, T_2 is returned as the final solution after all destination nodes are included.

Prefer_the_best (a, b)
if ($\forall j = 1, 2, \cdots, J, G_j[a] + R_j[a] \leq \Delta_j^t$) **then return** ($a$)
if ($\forall j = 1, 2, \cdots, J, G_j[b] + R_j[b] \leq \Delta_j^t$) **then return** ($b$)
if ($f[a] < f[b]$) **then return** (a)
return (b)

Fig. 4. The preference rule used in H_MCP

The initial tree T_1 (line 1) is found by running a modified Dijkstra's algorithm starting from the source node s with a nonlinear length function given by

$$h(\mathbf{p}) = \max\left\{\frac{w_j(\mathbf{p})}{\Delta_j}, j = 1, 2, \cdots, J\right\},$$

where $\Delta_j = \frac{1}{|D|}\sum_{v\in D}\Delta_j^v$, and $|D|$ is the number of destinations. The modified Dijkstra's algorithm used here is quite similar to the Look_Ahead_Dijkstra described in H_MCP. The only difference is that the predecessor of a node will always be determined based on function $h(\cdot)$. We expect that by selecting this length function the impact of the constraints of all destinations can be partially taken into account, and therefore the probability of finding a feasible tree T_1 could be increased.

Theorem 1: Heuristic HMCMC must be able to return a tree spanning s and all destinations as long as there is one.

Proof: HMCMC either returns T_1 or T_2. T_1 is a tree because it is composed of the shortest paths (w.r.t. $h(\cdot)$) from s to each destination. T_2 is a tree because each time when a path is included in T_2, the incoming links to all the nodes of that path except those forming it are eliminated, making sure that when searching for a path $\mathbf{p}$ from s to another destination, $\mathbf{p}$ will not be able to reenter T_2 once it leaves T_2. ■

The tree returned by HMCMC may not be a feasible solution because H_MCP will always return a path even if it is not feasible. The reason for returning the best available tree is based on the consideration that we may still be interested in or even accept such tree in the case that a feasible tree is very hard, if not impossible, to find.

Theorem 2: It is possible that there does not exist a feasible multi-constrained multicast tree even if a feasible multi-constrained path from s to each destination is available.

Proof: Just consider the case where there is only one feasible multi-constrained path from s to each destination and these paths do not form a tree. ■

Since HMCMC runs modified Dijkstra's algorithm once to find the initial tree T_1 and runs H_MCP at most $|D|$ times to

```
Heuristic HMCMC(G, s, D, Δ_j^v, v ∈ D, j = 1, 2, ···, J)
Input:
    G(V, E): graph
    s: source; D: set of destinations
    Δ_j^v: the jth upper bound for destination v ∈ D
Output:
    a multi-constrained tree spanning D ∪ {s}
    Find an initial tree T_1
    if T_1 is a feasible tree then
        return T_1
    T_2 ← {} /* empty set */
    for v ∈ D do{
        if path p_{T_1}(s, v) satisfies all constraints for v then
            T_2 ← T_2 ∪ p_{T_1}(s, v)
            prune all incoming links to the nodes on p_{T_1}(s, v)
            except those links that form p_{T_1}(s, v)
    }
    for v ∈ D do{
        if p_{T_1}(s, v) does not satisfy all constraints for v then
            p ← H_MCP(G, s, v, w_j, Δ_j^v, j = 1, 2, ···, J)
            T_2 ← T_2 ∪ p
            prune all incoming links to the nodes on path p
            except those links that form path p
    }
    return T_2
```

Fig. 5. The heuristic algorithm HMCMC for the MCMST problem

construct tree T_2, and considering that modified Dijkstra's algorithm has a time complexity of $O(n \log n + m)$, the time complexity of HMCMC can be expressed as $O(|D|(n \log n + m))$.

V. Performance Evaluation

The performance of heuristic HMCMC is investigated through simulations on network topologies with different number of nodes and various methods for generating routing requests. Two types of networks with 100 and 200 nodes respectively are generated using Waxman's method [15]. The number of destinations in a multicast routing request varies from 4 to 20.

For a given size of network with a specific number of destinations, we generate 50 network topologies, for each of which 10 instances of link weights are randomly generated, and for each instance of a topology, 50 different routing requests are produced. For each routing request, heuristics HMCMC and the simple heuristic described in the following subsection are executed independently to find a solution. Based on the results returned by each heuristic, the *success ratio* of finding a feasible solution with 95% confidence intervals is computed.

A. A simple heuristic for performance comparison

In order to make the performance evaluation more convincing, we also implement a simple heuristic to compare with HMCMC. This heuristic first finds the least-w_1 tree using Dijkstra's algorithm, and if the tree is a feasible solution, it is returned. Otherwise, it continues to check least-w_2 tree, least-w_3 tree, $\cdots$, least-w_J tree for feasibility.

In the following simulation results, we use T_0 to denote the solution of the simple heuristic described above, T_1 the initial tree found by HMCMC, and T_2 the final solution returned by HMCMC. It should be noted that $T_2 = T_1$ if T_1 is a feasible solution. To simplify the simulation, we only consider two link weights w_1 and w_2.

B. Success ratio vs. the size of multicast group

We first investigate the impact of the size of the multicast group on the performance measures. In this case, the upper bound of the jth constraint for destination v is given by

$$\Delta_j^v = \mathcal{R}[2,4] \times w_j\left(\mathrm{p}_{T_j}(s,v)\right),$$

where $\mathcal{R}[2,4]$ is a random number uniformly distributed on $[2,4]$, and T_j is the least-w_j tree.

Fig. 6 shows the success ratio with the number of destinations being 4, 8, 12, 16, and 20, respectively. In Fig. 6a, both of the two link weights w_1 and w_2 are uniformly distributed on $[1,1000]$, while in Fig. 6b they are distributed on $[1,1000]$ and $[500,2000]$, respectively. Notice that for a specific number of destinations the results for 100- and 200-node networks are purposely put in different positions to clearly show the confidence intervals.

From Fig. 6a, we may have the following observations: For any heuristic the success ratio decreases with the increase of the number of destinations. This is understandable since the more the number of destinations, the less the number of feasible solutions, and hence the harder it is to find a feasible solution. We should also notice that with a specific number of destinations, HMCMC can find a feasible initial tree T_1 with a success ratio much higher than that for the simple heuristic to find a feasible tree, and the success ratio can be further improved by 15% to 30% (or even more) if the final solution returned by HMCMC is evaluated. In addition, the success ratio decreases with the increase of the number of nodes.

We may obtain very similar observations from Fig. 6b except that in this case there are more feasible solutions available with a specific number of destinations, and therefore the success ratio is much higher than that for the corresponding case in Fig. 6a. For instance, the success ratio in Fig. 6b for the case of 200-node networks with 20 destinations is around 90%, while it is 45% in Fig. 6a. Clearly, the increase of the number of feasible solutions is attributed to the increase of the lower limit of the interval on which w_2 is distributed.

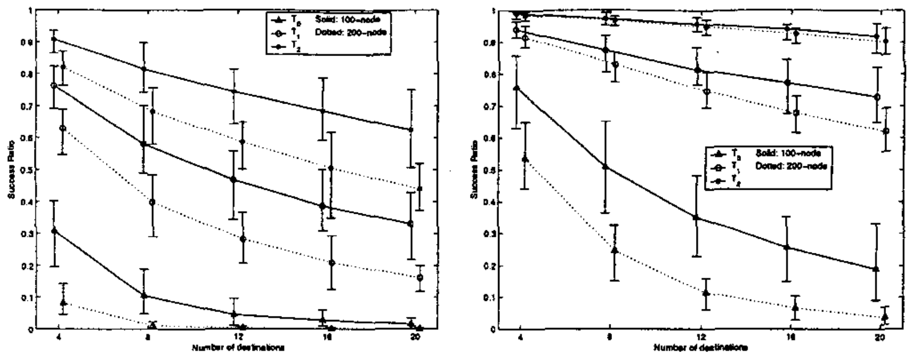
C. Success ratio vs. the upper bound

We also did experiments to show the influence of upper bounds on the success ratio by fixing the number of destinations to be 8 while generating upper bounds by

$$\Delta_j^v = \mathcal{R}[x, x+0.5] \times w_j\left(\mathrm{p}_{T_j}(s,v)\right),$$

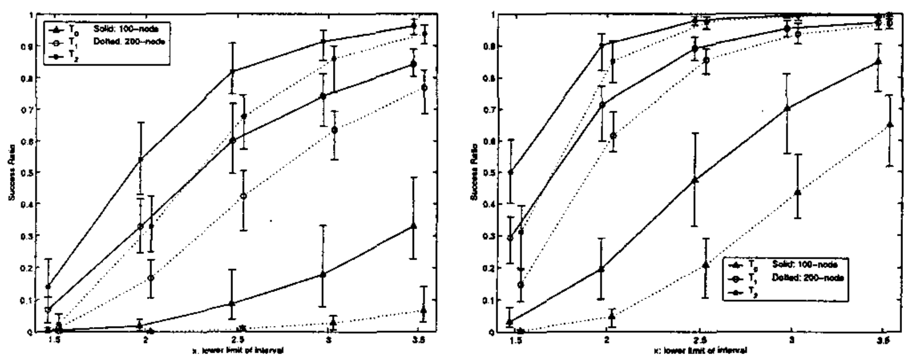
where x is the lower limit of the interval on which the random number $\mathcal{R}$ is distributed. We let x take several values between 1.5 and 3.5, and the result is shown in Fig. 7.

From Fig. 7a, we can see that the success ratio increases with the increase of x as excepted. We should also notice the clear difference between the success ratios for T_0, T_1, and T_2. Similar results are shown in Fig. 7b, and in this case we also notice that the success ratio is higher than that for the corresponding case in Fig. 7a because the number of feasible solutions increases drastically with x.



(a) $w_1 \sim [1, 1000]$, $w_2 \sim [1, 1000]$ (b) $w_1 \sim [1, 1000]$, $w_2 \sim [500, 2000]$

Fig. 6. Success ratio vs. the size of multicast group with link weights.



(a) $w_1 \sim [1, 1000]$, $w_2 \sim [1, 1000]$ (b) $w_1 \sim [1, 1000]$, $w_2 \sim [500, 2000]$

Fig. 7. Success ratio vs. the upper bound with 8 destinations.

VI. Conclusions

A QoS multicast routing algorithm that can find qualified solutions in a reasonably low time complexity is very important for high-speed networks to provide services with stringent and diverse QoS requirements. In this paper, we proposed a heuristic algorithm HMCMC to search for feasible solutions to the MCMST problem. Since nearly no previous work has been done on this problem, our work provides an excellent reference for future work. Our basic idea is to convert this multi-constrained multicast routing problem to a multi-constrained unicast routing problem, for which a variety of heuristics are available. Computer simulations demonstrate that the proposed heuristic can achieve a very satisfactory success ratio of finding feasible solution. It should also be noted that the proposed heuristic can be easily modified for solving the MCMST problem when cost is also concerned.

References

[1] S. Chen and K. Nahrstedt, "An overview of quality of service routing for next-generation high-speed networks: problems and solutions," *IEEE Network*, pp.64-79, Nov./Dec. 1998.

[2] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol.1, pp.269-271, 1959.

[3] M. Faloutsos, A. Banerjea, and R. Pankaj, "QoSMIC: Quality of service sensitive multicast Internet protoCol", *SIGCOMM '98*, pp.144-153, September 1998.

[4] G. Feng, K. Makki, and N. Pissinou, "Efficient implementations of a delay-constrained least-cost multicast algorithm," *Journal of Communications and Networks*, Vol. 4, No. 3, pp. 246 -255, Sep. 2002.

[5] G. Feng, "Nonlinear Lagrange relaxation based QoS routing revisited," submitted for publication.

[6] M. R. Garey and D. S. Johnson, *Computers and Intractability, A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, 1979.

[7] V. Kompella, J. C. Pasquale, and G. Polyzos, "Multicast routing for multimedia communication," *IEEE/ACM Trans. Networking*, vol. 1, no. 3, pp. 286-292, 1993.

[8] T. Korkmaz and M. Krunz, "Multi-constrained optimal path selection," *INFOCOM'2001*, Alaska, 2001.

[9] L. Kyou, G. Markowsky, and L, Berman, "A fast algorithm for Steiner trees," *Acta Info.*, vol. 15, no. 2, pp. 141-145, 1981.

[10] F. Kuipers, and P. Mieghem, "MAMCRA: a constrained-based multicast routing algorithm," *Computer Communications*, vol. 25, pp. 802-811, 2002.

[11] F. Kuipers, T. Korkmaz, M, Krunz, and P. Mieghem, "A review of constraint-based routing algorithm," Tech Rep, TU Delft, 2002.

[12] I. Matta, and L. Guo, "QDMR: An efficient QoS dependent multicast routing algorithm," *Journal of Communications and Networks*, vol. 2, no. 2, 2000.

[13] H. Neve and P. Mieghem, "A multiple quality of service routing algorithm for PNNI," *Proc. Of the ATM Workshop*, pp. 324-328, May 1998.

[14] H. F. Salama, D. S. Reeves, and Y. Viniotis, "Evaluation of multicast routing algorithms for real-time communication on high-speed networks," *IEEE JSAC*, vol. 15, no. 3, pp. 332-345, April 1997.

[15] B.M. Waxman, "Routing of multipoint connections," *IEEE JSAC*, 6(9): 1617-1622, Dec. 1988.

[16] R. Widyono, "The design and evaluation of routing algorithms for real-time channels," Tech. Rep. ICSI TR-94-024, University of California at Berkley, International Computer Science Institute, June 1994.

[17] Q. Zhu, M. Parsa, and J. J. Garcia-Luna-Aceves, "A source-based algorithm for delay-constrained minimum-cost multicasting," *IEEE INFOCOM'95*, vol. 1, pp. 377-385, 1995.