Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 718152, 11 pages
http://dx.doi.org/10.1155/2013/718152

Research Article

Many-to-Many Multicast Routing Schemes under a Fixed Topology

Wei Ding,[1] Hongfa Wang,[1] and Xuerui Wei[2]

[1] *Zhejiang Water Conservancy and Hydropower College, Hangzhou, Zhejiang 310018, China*
[2] *Department of Mathematics, Shaoxing University, Shaoxing, Zhejiang 312000, China*

Correspondence should be addressed to Wei Ding; dingweicumt@163.com

Received 11 January 2013; Accepted 31 January 2013

Academic Editors: A. Bogliolo and J. Zheng

Many-to-many multicast routing can be extensively applied in computer or communication networks supporting various continuous multimedia applications. The paper focuses on the case where all users share a common communication channel while each user is both a sender and a receiver of messages in multicasting as well as an end user. In this case, the multicast tree appears as a terminal Steiner tree (TeST). The problem of finding a TeST with a quality-of-service (QoS) optimization is frequently NP-hard. However, we discover that it is a good idea to find a many-to-many multicast tree with QoS optimization under a fixed topology. In this paper, we are concerned with three kinds of QoS optimization objectives of multicast tree, that is, the minimum cost, minimum diameter, and maximum reliability. All of three optimization problems are distributed into two types, the centralized and decentralized version. This paper uses the dynamic programming method to devise an exact algorithm, respectively, for the centralized and decentralized versions of each optimization problem.

1. Introduction

Multicast routing has been increasingly used in computer or communication networks supporting various multimedia applications, such as real-time audio and video conferences, entertainment, and distance learning, [1, 2]. Multicast routing is known as a *multicast tree* [3], which can reduce the usage of network resource, such as network cost and bandwidth. Multicast tree can be reduced to be a *Steiner tree* in mathematics [4].

In a real world, a large number of continuous multimedia applications drive the consumers to advance their *quality of service* (QoS) requirements [2, 5, 6] (e.g., cost, delay, and bandwidth). As we all know, the minimum cost multicast tree (Steiner tree) problem is NP-hard [7] and the minimum diameter multicast tree (Steiner tree) problem is polynomial solvable [8]. Furthermore, the multicast tree problem with additional QoS requirements is frequently harder to solve. For example, in past decade, a number of heuristics [9, 10] and distributed algorithms [11, 12] have been devised for finding a minimum cost delay-constrained multicast tree. Surprisingly, the *fixed topology* version of this problem, in which the configuration of multicast tree is given in advance with a tree topology, is easier than the classic version. Wang and Jia [13] designed a pseudo-polynomial-time algorithm, and Xue and Xiao [14] devised a full polynomial time approximation scheme. Not only that, we believe the idea of under a fixed topology will play an important role in exploring a desired multicast routing with a variety of QoS requirements.

In many practical settings, each destination is a *terminal*. A terminal means an end user, who takes charge of receiving and sending data but not branching them, that is, it can not serve as a *relay node* in charge of copying and branching data to other terminals. In a word, a terminal is a source, a receiver, or both. For instance, a member in video conference not only receives all the others' real-time images but also sends its real-time images to all the others. In fact, this is a type of general paradigm of *many-to-many* multicast routing. Provided that all terminals share a common communication channel, one many-to-many multicast tree can be reduced to a *terminal Steiner tree* (TeST) [15]. The minimum cost TeST problem is known to be NP-hard [15] and the minimum diameter TeST problem is polynomial solvable [8]. Many-to- many multicast

tree includes two types, the *centralized* and *decentralized*. In the former, a network node with a bootstrap serves as a server in charge of receiving a terminal's data and then copying and branching them to all the other terminals by using multicast, resulting in a centralized multicast tree, essentially a rooted TeST (with the root at the server node). The centralized multicast tree problem is in fact *one-to-many* multicast tree problem [13, 14, 16]. In the latter, a terminal sends its data directly to all the others using a common channel, resulting in a decentralized one, an unrooted TeST in essence.

To the best of our knowledge, there have been a number of studies on the unrestricted many-to-many multicast tree mentioned above (terminal Steiner tree) [8, 15, 17–19]; however few studies on the QoS restricted version [16] due to its vast difficulty. Fortunately, we discover that the idea of under a fixed topology could provide us with a new way of studying the restricted version. In this paper, we will study the unrestricted many-to-many multicast tree problem with the objective of cost minimized or delay minimized and/or reliability maximized under a fixed topology, respectively. The approaches and results presented in the rest of the paper will contribute to study the many-to-many multicast tree with QoS restrictions under a fixed topology in the recent future.

The rest of this paper is organized as follows. In Section 2, we introduce the architecture of many-to-many multicast tree under a fixed topology. In Section 3, we make preliminaries, including defining three kinds of metrics of QoS of multicast tree, three QoS optimization problems, and two Euclidean graphs. We present an exact algorithm, respectively, for the centralized and decentralized version of the minimum cost problem in Section 4, the minimum delay problem in Section 5, and maximum reliability problem in Section 6. In Section 7, we give an example to illustrate all the algorithms. In Section 8, we conclude the paper with some research topics.

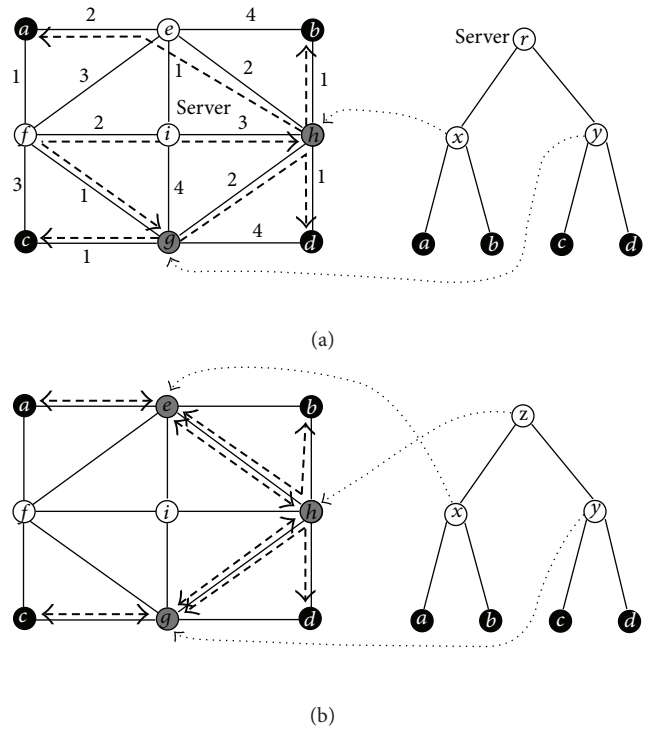

Figure 1: Ignore the dashed lines on the left top subfigure to obtain a sample graph where the number on each edge represents its length. The right top subfigure is a sample rooted TeST and the right bottom is a sample unrooted TeST. An example centralized many-to-many multicast tree under a fixed topology is distinguished by dashed lines on the left top subfigure and an example decentralized one is distinguished on the left bottom subfigure.

2. Architecture of Many-to-Many Multicast Tree under a Fixed Topology

A computer or communication network is frequently modeled as an undirected graph [20]. Let $G = (V, E, \omega)$ be an undirected edge-weighted graph with a subset $S \subset V$ of terminals and $T = (U, F)$ be a sample TeST. Here, V and E denote the node set and edge set of G, U and F denote the node set and edge set of T, and $\omega(e)$ denotes the weight on e for every $e \in E$. For any edge $f = \{u_i, u_j\} \in F$, we say that f is realized in G once its two endpoints u_i and u_j are mapped to two different nodes v_i and v_j in G. Note that u_i and u_j are not allowed to be mapped to a same node of G. In essence, f is mapped to a simple path in G connecting v_i and v_j, often a so-called optimal path with respect to some optimization objective, for example, a shortest path if $\omega(e)$ represents the length of e. Also, we use $R(f)$ to denote a realization of f in G, and use $\pi^*[v_i, v_j]$ to denote an optimal path in G between v_i and v_j for any a pair of nodes v_i and v_j.

Given any two different nodes x and y of T, there is a unique path in T between them. Specially, if x and y are both leaves, we call the path a *leaf-to-leaf path* of T, denoted as $L[x, y]$. In addition, the path in T between the root node r and a leaf x is called a *root-to-leaf path* of T, denoted as $C[r, x]$. Let $R(x, y)$ denote a realization of $L[x, y]$ in G and $R(r, x)$ a realization of $C[r, x]$. It is clear that $R(x, y)$ (resp. $R(r, x)$) is composed of all the realizations of edges on $L[x, y]$ (resp. $C[r, x]$) respectively. Furthermore, all the realizations of the edges on T form a realization of T in G, denoted as $R(T)$. A realization of T in G forms a many-to-may multicast tree under a fixed topology T in G essentially.

In this paper, we are concerned with the centralized and decentralized multicast trees under a fixed topology, namely, the realization of a given rooted and unrooted TeST topology; see Figure 1. On the top subfigure, the left graph shows a network with each edge having a weight denoting its length and the right graph shows a given rooted TeST topology. A realization of the rooted TeST in the network is distinguished by dashed edges. Considering that every leaf of the TeST is required to be mapped to a fixed node (terminal) and its root is required to be mapped to the server node, the essential work of realizing the TeST in the network is to map all the nonleaves except the root of the TeST to some nonterminals. If we arrange all the nonleaves in the order of from the 2nd level to top and from left to right on a level and label them by numbers $1, 2, \ldots, |U \setminus S|$ in sequence, we can denote all the nonleaves by an ordering (x, y, r), and then use another

ordering (h, g, i) to record all the selected nonterminals in the graph. Likewise, on the bottom subfigure, the left shows a same network as above and the right shows an unrooted TeST topology. A realization of the TeST is distinguished by dashed edges. Since each leaf of the TeST is required to be mapped to a fixed node (terminal), the essence of realizing the TeST in the network is to map all the nonleaves of the TeST to some nonterminals. We can transform an unrooted tree into a rooted tree by assigning any a nonleaf as the root, and similarly use (x, y, z) to denote all the nonleaves and (e, g, h) to record all the selected nonterminals in the graph. In general, we can use an ordering $(u_1, \ldots, u_{|U\setminus S|})$, $u_i \in U \setminus S$ to denote all the nonleaves of a given TeST and use another ordering $(v_1, \ldots, v_{|U\setminus S|})$, $v_i \in V \setminus S$ to record all the selected nonterminal nodes in sequence, that is, a multicast tree under a fixed topology. In this paper, we always use $|\cdot|$ to denote the cardinality of a set.

Let T be a rooted tree, T_u the subtree of T rooted at u, and $R(T_u)$ a realization of T_u. When u is a nonleaf node, we let $\Delta(u)$ be the set of all the children of u, and then immediately obtain that

$$\sum_{u\in U\setminus S} |\Delta(u)| = |F|. \tag{1}$$

3. Fundamental Preliminaries

In the section, we make some fundamental preliminaries, which will help us to analyze the problems and understand the algorithms proposed in the following.

3.1. Metrics. First of all, we define three kinds of metrics of QoS of tree, including the *cost, delay*,and *reliability* of tree.

3.1.1. Cost. We are given an undirected graph $G = (V, E, c)$ where $c(e)$ represents the cost on e for every $c(\cdot)$. Let π be a path in G. The cost of π is equal to the sum of all the costs of edges on π, that is, $c(\pi) = \sum_{e\in\pi} c(e)$. Hence, the cost of edge realization $R(f)$ of $f = \{u_i, u_j\}$ in which u_i and u_j are mapped to two different nodes v_i and v_j in G, respectively, is denoted as $c(\pi^*[v_i, v_j])$ where $\pi^*[v_i, v_j]$ is a shortest path in G between v_i and v_j.

The *cost* of $R(T)$ is defined as the sum of all the costs of edge realizations of T, that is,

$$c(R(T)) = \sum_{\{u_i,u_j\}\in F} c\left(\pi^*\left[v_i, v_j\right]\right). \tag{2}$$

Let $R^c(T)$ denote a minimum cost realization of T. Thus,

$$c\left(R^c(T)\right) = \min_{R(T)} \sum_{\{u_i,u_j\}\in F} c\left(\pi^*\left[v_i, v_j\right]\right). \tag{3}$$

3.1.2. Delay. We are given an undirected graph $G = (V, E, d)$ where every edge $e \in E$ has a weight $d(e)$ representing the delay on e. The delay of π is equal to the sum of all the delays of edges on π, that is, $d(\pi) = \sum_{e\in\pi} d(e)$. Then, the delay of edge realization $R(f)$ is denoted as $d(\pi^*[v_1, v_2])$ in which $\pi^*[v_i, v_j]$ is a minimum delay path in G between v_i and v_j.

The delay of $R(x, y)$ is equal to the sum of all the delays of edge realizations of $L[x, y]$, that is,

$$d(R(x, y)) = \sum_{\{u_i,u_j\}\in L[x,y]} d\left(\pi^*\left[v_i, v_j\right]\right). \tag{4}$$

Similarly, we have

$$d(R(r, x)) = \sum_{\{u_i,u_j\}\in C[r,x]} d\left(\pi^*\left[v_i, v_j\right]\right). \tag{5}$$

The maximum delay of leaf-to-leaf path realization of T is called the *diameter* of $R(T)$, that is,

$$\operatorname{diam}(R(T)) = \max_{\forall x,y\in S, x\neq y} d(R(x, y)). \tag{6}$$

Let $R^d(T)$ denote a minimum delay realization of T. So,

$$\operatorname{diam}\left(R^d(T)\right) = \min_{R(T)} \max_{\forall x,y\in S, x\neq y} d(R(x, y)). \tag{7}$$

The maximum delay of root-to-leaf path realization of T is called the *radius* of $R(T)$, that is,

$$\operatorname{radi}(R(T)) = \max_{\forall x\in S} d(R(r, x)). \tag{8}$$

Likewise,

$$\operatorname{radi}\left(R^d(T)\right) = \min_{R(T)} \max_{\forall x\in S} d(R(r, x)). \tag{9}$$

Note that realizations with a minimum diameter or radius are both denoted by $R^d(T)$. We can differentiate according to the context where it appears.

3.1.3. Reliability. We are given an undirected graph $G = (V, E, p)$ where every edge $e \in E$ has an independent working probability $p(e)$ while all the nodes are immune to failures. The working probability of π is equal to the product of all the working probabilities of edges on π, that is, $p(\pi) = \prod_{e\in\pi} p(e)$. Then, the working probability of edge realization $R(f)$ is denoted as $p(\pi^*[v_i, v_j])$ where $\pi^*[v_i, v_j]$ is a maximum reliability path in G between v_i and v_j.

The working probability of $R(x, y)$, named its *reliability*, is equal to the product of all the working probabilities of edge realizations of $L[x, y]$, that is,

$$p(R(x, y)) = \prod_{\{u_i,u_j\}\in L[x,y]} p\left(\pi^*\left[v_i, v_j\right]\right). \tag{10}$$

Likewise,

$$p(R(r, x)) = \prod_{\{u_i,u_j\}\in C[r,x]} p\left(\pi^*\left[v_i, v_j\right]\right). \tag{11}$$

The minimum reliability of leaf-to-leaf path realization of T is called the *diameter reliability* of $R(T)$, that is,

$$\mathrm{dia}p\left(R\left(T\right)\right) = \min_{\forall x,y\in S, x\neq y} p\left(R\left(x,y\right)\right). \tag{12}$$

Let $R^r(T)$ denote a maximum reliability realization of T. Thus,

$$\mathrm{dia}p\left(R^r\left(T\right)\right) = \max_{R(T)} \min_{\forall x,y\in S, x\neq y} p\left(R\left(x,y\right)\right). \tag{13}$$

The minimum reliability of root-to-leaf path realization of T is called the *radius reliability* of $R(T)$, that is,

$$\mathrm{rad}p\left(R\left(T\right)\right) = \min_{\forall x\in S} p\left(R\left(r,x\right)\right). \tag{14}$$

Similarly,

$$\mathrm{rad}p\left(R^r\left(T\right)\right) = \max_{R(T)} \min_{\forall x\in S} p\left(R\left(r,x\right)\right). \tag{15}$$

Note that realizations with a maximum diameter or radius reliability are both denoted by $R^r(T)$. We can differentiate according to the context where it appears.

3.2. Problem Definitions. The focus of this paper is three optimization problems of many-to-many multicast tree under a fixed topology from the perspective of three metrics above, which are formally defined as follows.

First of all, we use an INPUT (abbreviated to I) to simplify the statements of problems: an undirected graph $G = (V, E)$, a subset $S \subset V$ of terminals, and a sample TeST topology as $T = (U, F)$ (see Figure 1). Let

$$|V| = n, \quad |E| = m, \quad |U| = \alpha, \quad |F| = \beta, \quad |S| = \lambda. \tag{16}$$

Moreover, we define $\overline{S} = S \cup \{r\}$.

Problem 1. Given an INPUT with every edge $e \in E$ having a nonnegative weight $c(e) \geq 0$ representing the cost on e, the *minimum cost many-to-many multicast tree under a fixed TeST topology problem* (MCMP) aims to find an ordering $(v_1, \ldots, v_{m-k})^*$, $v_i \in V \setminus S$ with a minimum cost.

Problem 2. Given an INPUT with every edge $e \in E$ having a positive weight $d(e) > 0$ denoting the delay on e, the *minimum delay many-to-many multicast tree under a fixed TeST topology problem* (MDMP) aims to find an ordering $(v_1, \ldots, v_{m-k})^*$, $v_i \in V \setminus S$ with a minimum delay.

Problem 3. Given an INPUT with every edge $e \in E$ having a weight $0 < p(e) < 1$ representing the working probability on e, the *maximum reliability many-to-many multicast tree under a fixed TeST topology problem* (MRMP) aims to find an ordering $(v_1, \ldots, v_{m-k})^*$, $v_i \in V \setminus S$ with a maximum reliability.

3.3. Euclidean Graphs. Given an undirected graph $G = (V, E, \omega)$ with every edge $e \in E$ having a weight $\omega(e)$, we construct two types of *Euclidean graphs* based on G in the following.

3.3.1. Shortest-Path Graph. When $\omega(e)$, $\forall e \in E$ in $G = (V, E, \omega)$ represents the cost $c(e)$ or delay $d(e)$ on e, the path between v_i and v_j with a minimum weight (e.g., cost or delay) is called a *shortest path* (SP), denoted as $\pi^*[v_i, v_j]$. So,

$$\omega\left(\pi^*\left[v_i, v_j\right]\right) = \min_{\pi[v_i,v_j]} \sum_{e\in\pi[v_i,v_j]} \omega\left(e\right). \tag{17}$$

All-pairs SPs in G form a set, denoted as Σ, that is,

$$\Sigma = \left\{\pi^*\left[v_i, v_j\right] : \forall v_i,\ v_j \in V, v_i \neq v_j\right\}. \tag{18}$$

Next we construct a new graph $\Sigma(G) = (V, \Sigma, \omega)$ from G such that the edge between v_i and v_j just represents an SP $\pi^*[v_i, v_j]$ in G. We call $\Sigma(G)$ as a *Shortest-Path Graph* (SPG). Evidently, $\Sigma(G)$ is a complete graph.

3.3.2. Maximum-Reliability-Path Graph. When $\omega(e)$, $\forall e \in E$ in $G = (V, E, \omega)$ represents the working probability $p(e)$ on e, the path between v_i and v_j with a maximum working probability is called a *maximum reliability path* (MRP), denoted as $\pi^*[v_i, v_j]$. So,

$$p\left(\pi^*\left[v_i, v_j\right]\right) = \max_{\pi[v_i,v_j]} \prod_{e\in\pi[v_i,v_j]} p\left(e\right). \tag{19}$$

All-pairs MRPs in G form a set, denoted as Π, that is,

$$\Pi = \left\{\pi^*\left[v_i, v_j\right] : \forall v_i,\ v_j \in V,\ v_i \neq v_j\right\}. \tag{20}$$

Next, we construct a complete graph $\Pi(G) = (V, \Pi, p)$ from G such that the edge between v_i and v_j just represents an MRP $\pi^*[v_i, v_j]$. We call $\Pi(G)$ a *Maximum- Reliability-Path Graph* (MRPG).

4. Minimum Cost Many-to-Many Multicast Tree under a Fixed Topology

In this section, we study the centralized and decentralized MCMP, respectively.

4.1. Centralized MCMP. According to discussions above in Section 2, the essence of finding a minimum cost multicast tree under a given TeST in the centralized MCMP is to find a minimum cost realization of the rooted TeST topology. In this subsection, we devise a polynomial-time exact algorithm for the centralized MCMP.

Let $R(T_u)$ be a realization of T_u with u mapped to v in the centralized MCMP. Let $C[u][v]$ denote the cost of $R(T_u)$.

Step_1: Use the Floyd's algorithm to find all-pairs shortest paths and then obtain $\Sigma_c(G) = (V, \Sigma_c, c)$;
Step_2: **for** {all $u \in U$, $v \in V$} **do**
$C[u][v] \leftarrow 0$;
end for
for all $u \in U \setminus S$ **do**
if $u = r$ **then**
Compute $C[r][r]$ by (21);
end for all $v \in V \setminus \overline{S}$ **do**
Compute $C[u][v]$ by (21);
end for
end if
end for
Step_3: Trace out $(v_1, \ldots, v_{\alpha-\lambda})^*_{\mathrm{C}}$ top-down from r of T;

Algorithm 1: $(v_1, \ldots, v_{\alpha-\lambda})^*_{\mathrm{C}} = \mathrm{MCCT}[I]$.

Step_1: Use the Floyd's algorithm to find all-pairs shortest paths and then obtain $\Sigma_c(G) = (V, \Sigma_c, c)$;
Step_2: **for** {all $u \in U$, $v \in V$} **do**
$C[u][v] \leftarrow 0$;
end for
for {all $u \in U \setminus S$, $v \in V \setminus S$} **do**
Compute $C[u][v]$ by (23);
end for
Step_3: Trace out $(v_1, \ldots, v_{\alpha-\lambda})^*_{\mathrm{D}}$ top-down from r of T;

Algorithm 2: $(v_1, \ldots, v_{\alpha-\lambda})^*_{\mathrm{D}} = \mathrm{MCDT}[I]$.

When $u \in U$ is a leaf of T, considering that T_u contains a single node, we set $C[u][u] = 1$. Otherwise, for every $u \in U \setminus \overline{S}$, we can use (21) to compute $C[u][v]$ for all $v \in V \setminus \overline{S}$,

$$C[u][v] = \sum_{u_k \in \Delta(u)} \min_{v_k \in V \setminus \overline{S}} \{C[u_k][v_k] + c(v_k, v)\}, \quad (21)$$

where $c(v_k, v)$ denotes the cost of $\pi^*[v_k, v]$. We use (21) to compute all of $C[u][v]$ by the dynamic programming method until $C[r][r]$ is obtained, and then we can trace out an exact solution of the centralized MRMP easily if some bookkeeping information is saved during the computation.

Above analysis leads to algorithm MCCT that can find a minimum cost centralized multicast tree under a TeST, which is denoted as $(v_1, \ldots, v_{\alpha-\lambda})^*_{\mathrm{C}}$, in a polynomial time, see Theorem 1. We can use the approach in [21] to implement algorithm MCCT. So as to compute all $C[u][v]$, we need to get all-pairs shortest paths in G beforehand, namely, $\Sigma_c(G) = (V, \Sigma_c, c)$ where Σ_c denotes (18) with a cost on each edge of G. For the purpose, we can use the Floyd's algorithm, n times Dijkstra's algorithm, and so forth.

Theorem 1. *Given an INPUT as I, algorithm MCCT can find an optimal solution of the centralized MCMP correctly in $O(n^3 + \beta(n-\lambda)^2)$ time.*

Proof. Step_1 takes $O(n^3)$ time to find all-pairs shortest paths by using the Floyd's algorithm. Step_2 first spends $O(\alpha n)$ time to initialize $C[u][v]$ for all $u \in U$ and $v \in V$, and then uses (21) to compute $C[u][v]$ for all $\in U \setminus S$, whose time complexity is at most

$$\begin{aligned}
&O\left(|\Delta(r)| \cdot \left|V \setminus \overline{S}\right|\right) + \sum_{u \in U \setminus \overline{S}} \sum_{v \in V \setminus \overline{S}} O\left(|\Delta(u)| \cdot \left|V \setminus \overline{S}\right|\right) \\
&= O\left(|\Delta(r)| \cdot \left|V \setminus \overline{S}\right|\right) + O\left(\sum_{u \in U \setminus \overline{S}} |\Delta(u)| \cdot \sum_{v \in V \setminus \overline{S}} \left|V \setminus \overline{S}\right|\right) \\
&= O\left(|\Delta(r)| \cdot \left|V \setminus \overline{S}\right|\right) + O\left(\left|V \setminus \overline{S}\right|^2 \cdot \sum_{u \in U \setminus \overline{S}} |\Delta(u)|\right) \\
&\leq O\left(\left|V \setminus \overline{S}\right|^2 \cdot \sum_{u \in U \setminus S} |\Delta(u)|\right) \\
&\overset{(1)}{=} O\left(|F|(n-\lambda-1)^2\right) \\
&= O\left(\beta(n-\lambda)^2\right).
\end{aligned} \quad (22)$$

Step_3 only takes $O(\alpha - \lambda)$ time if we save some bookkeepings information during the computation in Step_2. □

4.2. Decentralized MCMP. By the discussions above in Section 2, to find a minimum cost multicast tree under a given

Step_1: Use the Floyd's algorithm to find all-pairs shortest paths and then obtain $\Sigma_d(G) = (V, \Sigma_d, d)$;
Step_2: **for** {all $u \in U$, $v \in V$} **do**
$Y[u][v] \leftarrow 0$;
end for
for all $u \in U \setminus S$ **do**
if $u = r$ **then**
Compute $Y[r][r]$ by (24);
else for all $v \in V \setminus \overline{S}$ **do**
Compute $Y[u][v]$ by (24);
end for
end if
end for
Step_3: Trace out $(v_1, \ldots, v_{\alpha-\lambda})^{\#}_{\mathrm{C}}$ top-down from r of T;

Algorithm 3: $(v_1, \ldots, v_{\alpha-\lambda})^{\#}_{\mathrm{C}} = \mathrm{MDCT}[I]$.

TeST in the decentralized MCMP is essentially to find a minimum cost realization of the unrooted TeST topology. Since an unrooted TeST can be transformed into a rooted TeST by assigning it any nonleaf as its root, we can adapt the algorithm MCCT for finding a minimum cost decentralized multicast tree under an unrooted TeST, which is denoted as $(v_1, \ldots, v_{\alpha-\lambda})^{*}_{\mathrm{D}}$. The resultant algorithm is named as MCDT. The main differences between MCDT and MCCT are that $v_k \in V \setminus \overline{S}$ in (21) is changed to $v_k \in V \setminus S$ in

$$C[u][v] = \sum_{u_k \in \Delta(u)} \min_{v_k \in V \setminus S} \{C[u_k][v_k] + c(v_k, v)\}, \quad (23)$$

and MCDT removes the restriction of r mapped to r from MCCT. Based on Theorem 1., we calculate the time complexity of algorithm MCDT, shown in Theorem 2.

Theorem 2. *Given an INPUT as I, algorithm MCDT can find an optimal solution of the decentralized MCMP correctly in* $O(n^3 + \beta(n-\lambda)^2)$ *time.*

Proof. It is similar to the proof of Theorem 1. □

5. Minimum Delay Many-to-Many Multicast Tree under a Fixed Topology

In this section, we study the centralized and decentralized MDMP, respectively.

5.1. Centralized MDMP. In the centralized MDMP, to find a minimum delay multicast tree under a given TeST topology is to find a minimum delay realization of the rooted TeST. In this subsection, we design a polynomial-time exact algorithm for the centralized MDMP.

Let $R(T_u)$ be a realization of T_u with u mapped to v in the centralized MDMP. The delay of $R(T_u)$ is equal to the radius of $R(T_u)$. We let $Y[u][v]$ denote the radius of $R(T_u)$. When $u \in U$ is a leaf of T, we set $Y[u][u] = 0$. Otherwise, for each $u \in U \setminus \overline{S}$, we can use (24) to compute $Y[u][v]$ for all $v \in V \setminus \overline{S}$,

$$Y[u][v] = \max_{u_k \in \Delta(u)} \min_{v_k \in V \setminus \overline{S}} \{Y[u_k][v_k] + d(v_k, v)\}, \quad (24)$$

where $d(v_k, v)$ denotes the delay of $\pi^*[v_k, v]$. We use (24) to compute $Y[u][v]$ recursively until $Y[r][r]$ is achieved, and then we can trace out an exact solution of the centralized MDMP easily if some bookkeeping information is saved during the whole computation.

Based on above analysis, we devise algorithm MDCT to find a minimum delay centralized multicast tree under a TeST, which is denoted as $(v_1, \ldots, v_{\alpha-\lambda})^{\#}_{\mathrm{C}}$, in a polynomial time; see Theorem 3. We also can use the way in [21] to execute algorithm MDCT. In order to compute all $Y[u][v]$, we need to get all-pairs shortest paths in G beforehand, namely, $\Sigma_d(G) = (V, \Sigma_d, d)$ in which Σ_d denotes (18) with a delay on every edge of G. Here, we can use the Floyd's algorithm to get $\Sigma_d(G)$.

Theorem 3. *Given an INPUT as I, algorithm MDCT can find an optimal solution of the centralized MDMP correctly in* $O(n^3 + \beta(n-\lambda)^2)$ *time.*

Proof. It is similar to the proof of Theorem 1. □

5.2. Decentralized MDMP. The essence of finding a minimum delay multicast tree under a given TeST in the decentralized MDMP is to find a minimum delay realization of the unrooted TeST topology.

First of all, we can always use the method in [22] to transform an unrooted tree into a rooted tree and further into a binary tree, denoted as $T^{\mathrm{B}} = (U^{\mathrm{B}}, F^{\mathrm{B}})$. Clearly, $|U^{\mathrm{B}} \setminus S| = |S| - 1 = \lambda - 1$. For any nonleaf $u \in U^{\mathrm{B}} \setminus S$, let u_l and u_r denote its left and right child, respectively. Let $R(T^{\mathrm{B}}_u)$ be a realization of T^{B}_u with u mapped to v in the decentralized MDMP. The delay of $R(T^{\mathrm{B}}_u)$ is equal to the diameter of $R(T^{\mathrm{B}}_u)$ and denoted by $X[u][v]$. When $u \in U^{\mathrm{B}}$ is a leaf of T^{B}_u, we set $X[u][u] = 0$

Step_1: Use the Floyd's algorithm to find all-pairs shortest paths and then obtain $\Sigma_d(G) = (V, \Sigma_d, d)$;
Step_2: **for** {all $u \in U^B$ and $v \in V$} **do**
$X[u][v] \leftarrow 0; Y[u][v] \leftarrow 0;$
end for
for {all $u \in U^B \setminus S$ and $v \in V \setminus S$} **do**
Compute $Y[u][v]$ by (26) and $X[u][v]$ by (25);
end for
Step_3: Trace out $(v_1, \ldots, v_{\alpha-\lambda})^{\#}_D$ top-down from r of T;

Algorithm 4: $(v_1, \ldots, v_{\alpha-\lambda})^{\#}_D = \mathrm{MDDT}[I]$.

since T^B_u has a single node. Otherwise, for each $u \in U^B \setminus S$, we can use (25) to compute $X[u][v]$ for all $v \in V \setminus S$,

$$X[u][v] = \min_{v_l, v_r \in V \setminus S} \max \{X[u_l][v_l], X[u_r][v_r], Y[u_l][v_l] + Y[u_r][v_r] + d(v_l, v) + d(v_r, v)\}, \quad (25)$$

where u_l is mapped to v_l and u_r is mapped to v_r, $d(v_l, v)$ and $d(v_r, v)$ denote the delay of $\pi^*[v_l, v]$ and $\pi^*[v_r, v]$, respectively, and both of $Y[u_l][v_l]$, $Y[u_r][v_r]$ can be derived from

$$Y[u][v] = \max \left\{ \begin{array}{l} \min_{v_l \in V \setminus S} \{Y[u_l][v_l] + d(v_l, v)\} \\ \min_{v_r \in V \setminus S} \{Y[u_r][v_r] + d(v_r, v)\} \end{array} \right\}. \quad (26)$$

We can use (25) to compute $X[u][v]$ recursively, and then we can trace out an exact solution of the decentralized MDMP, denoted as $(v_1, \ldots, v_{\alpha-\lambda})^{\#}_D$, if some bookkeeping information are is during the whole computation. The resulting algorithm is called MDDT, whose time complexity is presented in Theorem 4.

Theorem 4. *Given an INPUT as I, algorithm MDDT can find an optimal solution of the decentralized MDMP correctly in $O(n^3 + \lambda(n-\lambda)^3)$ time.*

Proof. It is similar to the proofs of Theorems 1 and 2. Step_2 takes $O((2\lambda - 1)n)$ time to initialize $X[u][v]$ and $Y[u][v]$ for all $u \in U^B$ and $v \in V$ and then computes $X[u][v]$ and $Y[u][v]$ for all $u \in U^B \setminus S$ and $v \in V \setminus S$, whose time complexity is at most

$$\begin{aligned} &\sum_{u \in U^B \setminus S} \sum_{v \in V \setminus S} \left(O(|V \setminus S|) + O\left(|V \setminus S|^2\right)\right) \\ &= O\left(\left|U^B \setminus S\right| \cdot |V \setminus S|^3\right) \\ &= O\left(\lambda(n-\lambda)^3\right). \end{aligned} \quad (27)$$

Therefore the time complexity of algorithm MDDT is no more than $O(n^3 + \lambda(n-\lambda)^3)$. □

Input: an undirected edge-weighted graph $G = (V, E, p)$;
Output: all-pairs MRP's of G (namely $\Pi(G) = (V, \Pi, p)$);
Step_1: **for** {$i = 1, 2, \ldots, |V|$ and $j = 1, 2, \ldots, |V|$} **do**
$p^{(0)}(i, j) \leftarrow P_{i,j}$;
end for
Step_2: **for** $k = 1, 2, \ldots, |V|$ **do**
for {$i = 1, 2, \ldots, |V|$ and $j = 1, 2, \ldots, |V|$} **do**
if $i = j$ **then**
$p^{(k)}(i, j) \leftarrow \infty$;
else
Compute $p^{(k)}(i, j)$ by (29);
end if
end for
end for

Procedure 1: Procedure CMRP.

6. Maximum Reliability Many-to-Many Multicast Tree under a Fixed Topology

In this section, we study the centralized and decentralized MRMP, respectively.

6.1. Constructing an MRPG. MRPG based on G will play an important role in the design of algorithm for MRMP in G. According to the discussions in Section 3, MRPG based on G is an Euclidean graph comprising all-pairs MRPs in G. So the key work of constructing MRPG is to devise an efficient algorithm for finding all-pairs MRPs. In this section, we present such an algorithm with a cubic time.

Firstly, we introduce a fundamental Lemma 5.

Lemma 5. *Given an undirected graph $G = (V, E, p)$ with every edge $e \in E$ having an independent working probability, for any path $\pi[v_i, v_j]$ composed of two subpaths $\pi[v_i, v_k]$ and $\pi[v_k, v_j]$, $\pi[v_i, v_j]$ is an MRP in G if and only if both $\pi[v_i, v_k]$ and $\pi[v_k, v_j]$ are MRPs in G.*

Proof. On one hand, if $\pi[v_i, v_j]$ is an MRP, we can verify that the combination of $\pi'[v_i, v_k]$ and $\pi[v_k, v_j]$ forms a more reliable path than $\pi[v_i, v_j]$ provided that $\pi'[v_i, v_k]$ is a more reliable path than $\pi[v_i, v_k]$ or the combination of $\pi[v_i, v_k]$ and $\pi'[v_k, v_j]$ forms a more reliable path than $\pi[v_i, v_j]$ provided

Step_1: Use procedure CMRP to find all-pairs maximum reliability paths and then obtain $\Pi(G) = (V, \Pi, p)$;
Step_2: **for** {all $u \in U$, $v \in V$} **do**
$\varphi[u][v] \leftarrow 0$;
end for
for all $u \in U \setminus S$ **do**
if $u = r$ **then**
Compute $\varphi[r][r]$ by (30);
else for all $v \in V \setminus \overline{S}$ **do**
Compute $\varphi[u][v]$ by (30);
end for
end if
end for
Step_3: Trace out $(v_1, \ldots, v_{\alpha-\lambda})_C^{\triangle}$ top-down from r of T;

Algorithm 5: $(v_1, \ldots, v_{\alpha-\lambda})_C^{\triangle} = \mathrm{MRCT}[I]$.

that $\pi'[v_k, v_j]$ is a more reliable path than $\pi[v_k, v_j]$. This causes a contradiction. On the other hand, if $\pi[v_i, v_k]$ and $\pi[v_k, v_j]$ are both MRP's, we can verify that either $\pi'[v_i, v_k]$ is more reliable than $\pi[v_i, v_k]$ or $\pi'[v_k, v_j]$ is more reliable than $\pi[v_k, v_j]$ provided that $\pi'[v_i, v_j]$ consisting of $\pi'[v_i, v_k]$ and $\pi'[v_k, v_j]$ is a more reliable path than $\pi[v_i, v_j]$. This causes a contradiction. □

From Lemma 5, we claim that the most reliable paths in G satisfy the *triangle inequality*, based on which we can design a dynamic programming algorithm for finding all-pairs MRP's.

For any edge $e = \{v_i, v_j\} \in E$, we rewrite $p(e)$ to be $p(v_i, v_j)$. We can use (28) to construct a probability matrix $P = (P_{i,j})_{n \times n}$,

$$P_{i,j} = \begin{cases} 1 & \text{if } i = j, \\ p\left(v_i, v_j\right) & \text{if } i \neq j, \left\{v_i, v_j\right\} \in E, \\ 0 & \text{if } i \neq j, \left\{v_i, v_j\right\} \notin E. \end{cases} \tag{28}$$

We use $p^{(k)}(i, j)$ to denote the working probability of a current MRP between v_i and v_j after v_k is introduced. We set $p^{(0)}(i, j) = P_{i,j}$ initially and then use (29) to compute $p^{(k)}(i, j)$ for k from 1 to n.

$$p^{(k)}(i, j) = \max\left\{p^{(k-1)}(i, j), p^{(k-1)}(i, k) \times p^{(k-1)}(k, j)\right\}. \tag{29}$$

Finally, $p^{(n)}(i, j)$ is the working probability of the MRP in G between v_i and v_j.

In essence, above idea of using (29) recursively forms our dynamic programming algorithm for finding all-pairs MRP's, namely, constructing $\Pi(G) = (V, \Pi, p)$, which is described as procedure CMRP (Procedure 1). The time complexity of CMRP is shown in Lemma 6.

Lemma 6. *Given an undirected edge-weighted graph $G = (V, E, p)$ with n nodes and m edges in which every edge has an independent working probability, procedure CMRP can find all-pairs maximum reliability paths in $O(n^3)$ time.*

Proof. Step_1 spends $O(n^2)$ time to initialize $p^{(0)}(i, j)$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$. For each $k = 1, 2, \ldots, n$, Step_2 spends $O(n^2)$ time to compute $p^{(k)}(i, j)$ for $i = 1, \ldots, n$ and $j = 1, \ldots, n$. Therefore, the time complexity of CMRP is $O(n^3)$. □

6.2. Algorithms. In this section, we present an exact algorithm for the centralized and decentralized MRMP, respectively.

6.2.1. Centralized MRMP. The essence of finding a maximum reliability multicast tree under a TeST topology in the centralized MRMP is to find a maximum reliability realization of the rooted TeST.

Let $R(T_u)$ be a realization of T_u with u mapped to v in the centralized MRMP. The reliability of $R(T_u)$ refers to its radius reliability, which is denoted as $\varphi[u][v]$. When $u \in U$ is a leaf of T, we set $\varphi[u][u] = 0$. Otherwise, for each $u \in U \setminus \overline{S}$, we can use (30) to compute $\varphi[u][v]$ for all $v \in V \setminus \overline{S}$,

$$\varphi[u][v] = \min_{u_k \in \Delta(u)} \max_{v_k \in V \setminus \overline{S}} \left\{\varphi[u_k][v_k] \times p(v_k, v)\right\}, \tag{30}$$

where $p(v_k, v)$ denotes the reliability of $\pi^*[v_k, v]$. We use (30) to compute $\varphi[u][v]$ recursively until $\varphi[r][r]$ is got, and then we can trace out an exact solution of the centralized MRMP if some bookkeeping information is saved during the whole computation.

Above analysis can be described as algorithm MRCT. It can find a maximum reliability centralized multicast tree under a TeST, denoted as $(v_1, \ldots, v_{\alpha-\lambda})_C^{\triangle}$, in a polynomial time; see Theorem 7. We use the method in [21] to perform algorithm MRCT. In order to compute $\varphi[u][v]$, we need to get all-pairs maximum reliability paths in G beforehand, namely, $\Pi(G) = (V, \Pi, p)$. This work can be accomplished by procedure CMRP.

Step_1: Use procedure CMRPto find all-pairs maximum reliability paths and then obtain $\Pi(G) = (V, \Pi, p)$;
Step_2: **for** {all $u \in U^{\mathrm{B}}$ and $v \in V$} **do**
$\varphi[u][v] \leftarrow 0; \psi[u][v] \leftarrow 0;$
end for
for {all $u \in U^{\mathrm{B}} \setminus S$ and $v \in V \setminus S$} **do**
Compute $\varphi[u][v]$ by (32) and $\psi[u][v]$ by (31);
end for
Step_3: Trace out $(v_1, \ldots, v_{\alpha-\lambda})^{\triangle}_{\mathrm{D}}$ top-down from r of T;

Algorithm 6: $(v_1, \ldots, v_{\alpha-\lambda})^{\triangle}_{\mathrm{D}} = \mathrm{MRDT}[I]$.

Theorem 7. *Given an INPUT as I, algorithm MRCT can find an optimal solution of the centralized MRMP correctly in* $O(n^3 + \beta(n-\lambda)^2)$ *time.*

Proof. It is similar to the proof of Theorem 1. □

6.2.2. Decentralized MRMP. To find a maximum reliability multicast tree under a TeST in the decentralized MDMP is essentially to find a maximum reliability realization of the unrooted TeST. We can use the way in [22] to transform an unrooted tree into a rooted tree and further into a binary tree $T^{\mathrm{B}} = (U^{\mathrm{B}}, F^{\mathrm{B}})$. And some definitions and notations therein are still used here. Let $R(T^{\mathrm{B}}_u)$ be a realization of T^{B}_u with u mapped to v in the decentralized MRMP. The reliability of $R(T^{\mathrm{B}}_u)$ refers to its diameter reliability, which is denoted as $\psi[u][v]$. When $u \in U^{\mathrm{B}}$ is a leaf of T^{B}_u, we set $\psi[u][u] = 0$. Otherwise, for each $u \in U^{\mathrm{B}} \setminus S$, we can use (31) to compute $\psi[u][v]$ for all $v \in V \setminus S$,

$$\begin{aligned}\psi[u][v] &= \max_{v_l, v_r \in V \setminus S} \min\{\psi[u_l][v_l], \psi[u_r][v_r], \\ &\qquad \varphi[u_l][v_l] \times \varphi[u_r][v_r] \times p(v_l, v) \times p(v_r, v)\},\end{aligned} \tag{31}$$

where u_l is mapped to v_l and u_r is mapped to v_r, $p(v_l, v)$ and $p(v_r, v)$ denote the reliability of $\pi^*[v_l, v]$ and $\pi^*[v_r, v]$, respectively, and both of $\varphi[u_l][v_l]$, $\varphi[u_r][v_r]$ can be obtained by using

$$\varphi[u][v] = \min\left\{\begin{array}{l}\max\limits_{v_l \in V \setminus S}\{\varphi[u_l][v_l] \times p(v_l, v)\} \\ \max\limits_{v_r \in V \setminus S}\{\varphi[u_r][v_r] \times p(v_r, v)\}\end{array}\right\}. \tag{32}$$

We can use (31) to compute $\psi[u][v]$ recursively, and then we can trace out an exact solution of the decentralized MRMP, denoted as $(v_1, \ldots, v_{\alpha-\lambda})^{\triangle}_{\mathrm{D}}$, if some bookkeeping information is saved during the whole computation. This leads to algorithm MRDT, whose time complexity is shown in Theorem 8.

Theorem 8. *Given an INPUT as I, algorithm MRDT can find an optimal solution of the decentralized MRMP correctly in* $O(n^3 + \lambda(n-\lambda)^3)$ *time.*

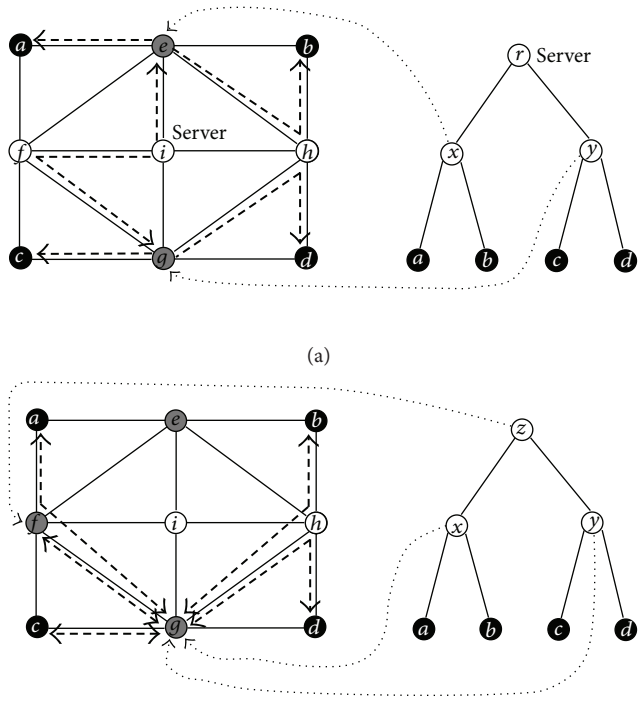

Figure 2: For ease of view, we neglect the numbers on edges. The example network for MCMP and the binary tree topology are both the same as those shown in Figure 1 and the integer on every edge represents its quantity of cost. The minimum cost multicast tree for the centralized MCMP is drawn with the bold dashed edges on the top subfigure and the one for the decentralized MCMP is drawn on the bottom subfigure.

Proof. It is similar to the proof of Theorem 4. □

7. Illustrative Examples

In this section, we take the network and the binary tree topology shown in Figure 1 as an example to illustrate the six algorithms proposed above (Algorithms 1–6).

Suppose the integer on every link of the network in Figure 1 represents the quantity of cost on the link. For instance, each unit of cost is 0.8 dollar; then the cost on edge $\{b, e\}$ is 3.2 dollars. We apply algorithm MCCT to solve the centralized MCMP. MCCT first uses the Floyd's algorithm to find all-pairs minimum cost paths in the given network, that is, constructing the SPG $\Sigma_c(G) = (V, \Sigma_c, c)$ with respect to the link costs of network then computes the values of (21) recursively, and finally terminates with an optimal ordering (e, g, i) of the centralized MCMP. So we know that x, y, r are mapped to e, g, i, respectively, and then derive an optimal solution from $\Sigma_c(G)$ distinguished by bold dashed edges on the top subfigure of Figure 2; for example, the communication between x and b is established by the minimum cost path e-h-b and the communication between r and y is established by the minimum cost path i-f-g. Similarly we can apply algorithm MCDT to solve the decentralized MCMP. MCDT first constructs $\Sigma_c(G)$, then computes the values of (23)

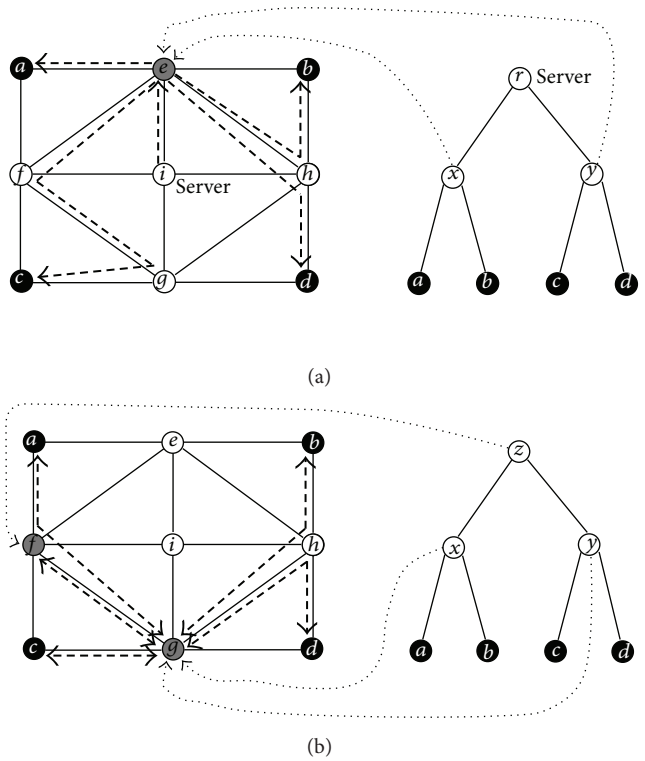

Figure 3: The example network for MDMP and the binary tree topology are both the same as those shown in Figure 1 and the integer on every edge represents its amount of delay. The minimum delay multicast tree for the centralized MDMP is drawn with the bold dashed edges on the top subfigure and the one for the decentralized MDMP is drawn on the bottom subfigure.

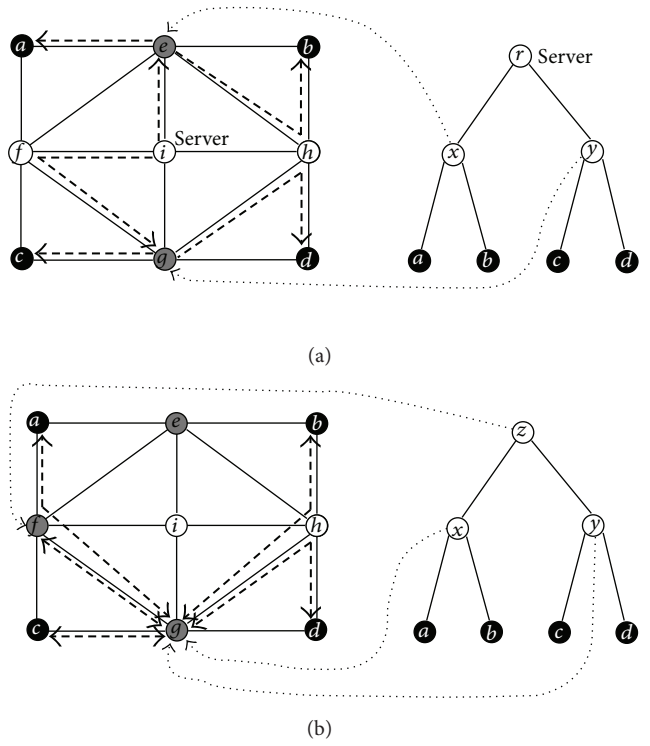

Figure 4: The example network for MRMP and the binary tree topology are both the same as those shown in Figure 1 and the integer on every edge represents the parameter of the linear probability function. The maximum reliability multicast tree for the centralized MRMP is drawn with the bold dashed edges on the top subfigure and the one for the decentralized MRMP is drawn on the bottom subfigure.

recursively, and finally ends with (g, g, f). Hence we can derive an optimal solution to the decentralized MCMP from $\Sigma_c(G)$, distinguished by bold dashed edges on the bottom subfigure of Figure 2.

Suppose the integer on every link of the network in Figure 1 represents its amount of delay. For example, every unit of delay is 1 ms, then the delay on link $\{e, f\}$ is 3 ms. We apply algorithm MDCT to solve the centralized MDMP. MDCT first uses the Floyd's algorithm to find all-pairs minimum delay paths in the given network, that is, constructing the SPG $\Sigma_d(G) = (V, \Sigma_d, d)$ with respect to the link delays of network, then computes the values of (24) recursively, finally ends with an optimal ordering (e, e, i) of the centralized MDMP. So we know that x, y, r are mapped to e, e, i, respectively, then derive an optimal solution from $\Sigma_d(G)$ distinguished by dashed edges on the top subfigure of Figure 3. Similarly, we apply MDDT to the decentralized MDMP. MDDT first constructs $\Sigma_d(G)$, and then computes the values of (25) and (26) recursively, finally terminates with (g, g, f). We can derive an optimal solution to the decentralized MDMP from $\Sigma_d(G)$, which is distinguished by bold dashed edges on the bottom subfigure of Figure 3 and equal to the optimal solution to the decentralized MCMP.

Suppose that every link $e \in E$ of the network given in Figure 1 has an independent working probability whose value is a linear function in the integer $\delta(e)$ on e, formulated as $p(e) = 1 - 0.01 \times \delta(e)$. For instance, the working probability of $\{a, e\}$ is 0.98. We apply algorithm MRCT to solve the centralized MRMP. MRCT first uses procedure CMRP to find all-pairs maximum reliability paths in the network, that is, constructing the MRPG $\Pi(G) = (V, \Pi, p)$ with respect to the link probabilities of the given network, then computes the values of (30) recursively, finally terminates with an optimal ordering (e, g, i) of the centralized MRMP. We can get an optimal solution from $\Pi(G)$ distinguished by bold dashed edges on the top subfigure of Figure 4. Similarly, we can apply algorithm MRDT to solve the decentralized MRMP. MRDT first constructs $\Pi(G)$, then computes the values of (31) and (32) recursively, and finally ends with (g, g, f). Hence we can derive an optimal solution to the decentralized MRMP from $\Pi(G)$, distinguished by bold dashed edges on the bottom subfigure of Figure 4. Evidently, the optimal solution to the centralized MRMP is same as that to the centralized MCMP as well as the optimal solution to the decentralized MRMP is same as that to the decentralized MCMP.

8. Conclusions

This paper introduces the architecture of a many-to-many multicast tree with fixed topology, reduces it to a realization of

the given TeST topology, and applies the idea of under a fixed topology to deal with three optimization problems, that is, the minimum cost, minimum delay, and maximum reliability multicast tree under a fixed topology problem. Each problem includes the centralized and decentralized versions. For the both versions of each problem, an exact algorithm is devised using the dynamic programming approach, respectively. On the condition that we are given a collection of alternative tree topologies, it is of interests to explore a best topology from all the alternative tree topologies.

Moreover, if we consider two or more weights on every link of a network, it is an interesting and important research topic how to devise an efficient algorithm for the multicast tree problem under a fixed topology with multiple objectives or with a single objective and at least one constraint.

Acknowledgments

This research was supported in part by the Key Research Grant xkyzd201207 (Ding) and the 2010 Excellent Young Teacher Grant 19093-88 (Ding) of Zhejiang Water Conservancy and Hydropower College.

References

[1] F. Kuo, W. Effelsberg, and J. J. Garcia-Luna-Aceves, *Multimedia Communications: Protocols and Applications*, Prentice Hall, Englewood Cliffs, NJ, USA, 1998.

[2] Z. Wang and J. Crowcroft, "Quality-of-service routing for supporting multimedia applications," *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 1228–1234, 1996.

[3] K. Bharath-Kumar and J. M. Jaffe, "Routing to multiple destinations in computer networks," *IEEE Transactions on Communications*, vol. 31, pp. 343–351, 1983.

[4] D. Z. Du, J. M. Smith, and J. H. Rubinstein, *Advances in Steiner Trees*, Kluwer Academic Publishers, Dordrecht, Netherlands, 2000.

[5] G. Apostolopoulos, R. Guerin, S. Kamat, and S. Tripathi, "Quality of service based routing: a performance perspective," in *Proceedings of the Applications, Technologies, Architectures, and Protocols for Computer Communication (ACM SIGCOMM '98)*, vol. 98, pp. 17–28, 1998.

[6] B. Wang and J. C. Hou, "Multicast routing and its QoS extension problems," *IEEE Network Algorithms, and Protocols*, vol. 14, pp. 22–36, 2000.

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

[8] W. Ding and K. Qiu, "Algorithms for the minimum diameter terminal steiner tree problem," *Journal of Combinatorial Optimization*, 2013.

[9] V. P. Kompella, J. C. Pasquale, and G. C. Polyzo, "Multicast routing for multimedia communication," *IEEE/ACM Transactions on Networking*, vol. 1, pp. 286–292, 1993.

[10] Q. Zhu, M. Parsa, and J. Garcia-Luna-Aceves, "A source-based algorithm for delay-constrained minimal-cost multicasting," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM '95)*, pp. 377–384, 1995.

[11] X. Jia, "A distributed algorithm of delay-bounded multicast routing for multimedia applications in wide area networks," *IEEE/ACM Transactions on Networking*, vol. 6, no. 6, pp. 828–837, 1998.

[12] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, "Two distributed algorithms for multicasting multimedia information," in *Proceedings of the IEEE Computer Communications and Networks (ICCCN '93)*, pp. 343–349, 1993.

[13] L. Wang and X. Jia, "Note fixed topology steiner trees and spanning forests," *Theoretical Computer Science*, vol. 215, pp. 359–370, 1999.

[14] G. Xue and W. Xiao, "A polynomial time approximation scheme for minimum cost delay-constrained multicast tree under a steiner topology," *Algorithmica*, vol. 41, no. 1, pp. 53–72, 2004.

[15] G. Lin and G. Xue, "On the terminal steiner problem," *Information Processing Letters*, vol. 84, pp. 103–107, 2002.

[16] M. Moh and B. Nguyen, "QoS-guaranteed one-to-many and many-to-many multicast routing," *Computer Communications*, vol. 26, pp. 652–669, 2003.

[17] Y. H. Chen, "An improved approximation algorithm for the terminal steiner tree problem," in *Computational Science and Its Applications (ICCSA '11)*, B. Murgante et al., Ed., vol. 6784 of *Lecture Notes in Computer Science*, pp. 141–151, 2011.

[18] D. E. Drake and S. Hougrady, "On approximation algorithms for the terminal steiner tree problem," *Information Processing Letters*, vol. 89, pp. 15–18, 2004.

[19] F. V. Martineza, J. C. D. Pinab, and J. Soares, "Algorithm for terminal steiner trees," *Theoretical Computer Science*, vol. 389, pp. 133–142, 2007.

[20] J. A. Bondy and U. S. R. Murty, *Graph Theory with Application.*, Macmillan, London, UK, 1976.

[21] W. Ding and G. Xue, "A linear time algorithm for computing a most reliable source on a tree network with faulty nodes," *Theoretical Computer Science*, vol. 412, no. 3, pp. 225–232, 2011.

[22] A. Tamir, "An $O(pn^2)$ algorithm for the p-median and related problems on tree graphs," *Operations Research Letters*, vol. 19, pp. 59–64, 1996.