Minimum-Cost Multiple Paths Subject to Minimum Link and Node Sharing in a Network

S. Q. Zheng, *Senior Member, IEEE*, Jianping Wang, *Member, IEEE*, Bing Yang, and Mei Yang, *Member, IEEE*

***Abstract*—In communication networks, multiple disjoint communication paths are desirable for many applications. Such paths, however, may not exist in a network. In such a situation, paths with minimum link and/or node sharing may be considered. This paper addresses the following two related fundamental questions. First, in case of no solution of disjoint multiple paths for a given application instance, what are the criteria for finding the best solution in which paths share nodes and/or links? Second, if we know the criteria, how do we find the best solution? We propose a general framework for the answers to these two questions. This framework can be configured in a way that is suitable for a given application instance. We introduce the notion of link shareability and node shareability and consider the problem of finding minimum-cost multiple paths subject to minimum shareabilities (MCMPMS problem). We identify 65 different link/node shareability constraints, each leading to a specific version of the MCMPMS problem. In a previously published technical report, we prove that all the 65 versions are mutually inequivalent. In this paper, we show that all these versions can be solved using a unified algorithmic approach that consists of two algorithm schemes, each of which can be used to generate polynomial-time algorithms for a set of versions of MCMPMS. We also discuss some extensions where our modeling framework and algorithm schemes are applicable.**

***Index Terms*—Algorithm, complexity, disjoint paths, graph, multiple paths, network, network flow, network planning, protection, protocol, reliability, routing, survivability.**

I. Introduction

IN A COMMUNICATION network, the connection between a source node and a destination node is a path between them. Currently, most networks employ protocols based on shortest-path routing algorithms that determine a single path with the minimum cost. Finding multiple paths between a source and a destination has been proposed. Potential benefits of multiple paths include improved reliability (e.g., [8], [13], [17], [18], and [20]–[24]), load balancing (e.g., [7] and [16]), higher network throughput (e.g., [9] and [16]), and alleviation of congestion (e.g., [2] and [8]). It is desirable that multiple paths are link- and/or node-disjoint.

In the literature, various versions of the problem of finding optimal disjoint paths between two nodes in a network have been investigated. Ford and Fulkerson proposed a polynomial-time algorithm for finding two paths with minimum total cost (the *Min-Sum 2-Path Problem*) based on minimum-cost network flow model [5]. Suurballe and Tarjan provided a different treatment and presented algorithms that are more efficient [14], [15]. Li *et al.* proved that the problem of finding two disjoint paths such that the cost of the longer path is minimized (the *Min-Max 2-Path Problem*) is NP-complete [11]. They also considered a generalized min-sum problem (the *G-Min-Sum k-Path Problem*), assuming that each link is associated with k different lengths where the jth link-cost is associated with the jth path. They showed that the G-Min-Sum k-Path Problem is NP-complete for $k \geq 2$ [12]. In [19]–[23], a set of optimal disjoint two-path problems with different objective functions, including the *Min-Min 2-path problem*, the *α-MIN-SUM 2-path problem*, and the *MinSum-MinMin 2-path problem*, were considered and proved to be NP-complete.

Given a pair of nodes, finding k, $k > 1$, disjoint paths, though desirable, may not always be possible in practical network applications for at least two reasons. First, if the network is too sparse, such paths may not physically exist. Second, if some links are overly saturated, additional traffic on these links may be prohibited so that two disjoint paths without using these links do not exist. When k disjoint paths do not exist, alternatively k paths from the source to the destination subject to minimum shared links/nodes should be found. In the context of network reliability, these paths can provide partial protection [24].

In the literature, limited work on multiple paths subject to minimum number of shared links/nodes has been reported. In [3], an algorithm based on minimum-cost network flow (MCNF) was given for finding the k-best paths (i.e., k paths with minimum node sharing). However, the algorithm can only be applied to trellis graphs. In [13], an algorithm was provided to transform an arbitrary graph to a trellis graph and then to obtain the k-best paths by the algorithm given in [3]. As analyzed in [10], this solution is only a heuristic one, and the complexity of the transformation is quite high. In [10], a MCNF-based algorithm was proposed for finding the k-best paths in arbitrary networks. However, only best link-disjoint paths were considered in [10].

This paper addresses the following two related fundamental questions. First, in case of no solution of disjoint multiple paths for a given application instance, what are the criteria for finding the best solution in which paths share nodes and/or links? Second, if we know the criteria, how do we find the best solution? It is clear that the answers to these two questions

Manuscript received July 05, 2007; revised July 10, 2008 and March 22, 2009; accepted January 31, 2010; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor D. Raz. Date of publication April 05, 2010; date of current version October 15, 2010. This work was supported in part by the National Science Foundation under Grant NSF-0714057 and the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CityU 121107).

S. Q. Zheng is with the Department of Computer Science, University of Texas at Dallas, Richardson, TX 75080 USA.

J. Wang is with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong (e-mail: jianwang@cityu.edu.hk).

B. Yang is with Cisco Systems, Richardson, TX 75082 USA.

M. Yang is with the Department of Electrical and Computer Engineering, University of Nevada, Las Vegas, NV 89154 USA.

Digital Object Identifier 10.1109/TNET.2010.2044514

are application-dependent, and they are not unique. Instead of aiming at finding answers for a particular application, though the aspect of network reliability and survivability is used as an application example, we propose a general framework for the answers to these two questions. This framework can be configured in a way that is suitable for a given specific application instance.

We first introduce the notion of *link shareability* and *node shareability*, which are variations of the concept of vulnerability defined in [17]. We use this notion to characterize the degree of link/node sharing among different paths. Larger link/node shareability implies more link/node sharing among a set of paths. A set of paths are link-disjoint if the link shareability of the paths is 0, and they are node-disjoint if their node shareability is 0 (in this case, their link shareability is also 0). We define a collection of minimum-cost multiple paths problems with prioritized minimization objectives. Shareability minimization objectives are treated as constraints for finding minimum-cost paths. We first define five basic shareability constraints: minsum link shareability constraint, minsum node shareability constraint, minmax link shareability constraint, minmax node shareability constraint, and empty constraint (no restriction on shareabilities). Based on these basic shareability constraints, we identify 65 shareability constraints, which are obtained by selections and permutations of basic shareability constraints. We investigate the problem of finding minimum-cost paths subject to these shareability constraints. For convenience, we use *MCMPMS* problem to refer to the problem of finding *M*inimum *C*ost *M*ultiple *P*aths between a pair of nodes in a network subject to various *M*inimum *S*harability constraints. Unless otherwise specified, the number of paths is assumed to be k. The MCMPMS problem has many versions, each corresponding to a particular shareability constraint where we have proved that the 65 versions are pair-wisely inequivalent in [25].

The MCMPMS problem is a generalization of the classical fundamental problem of finding minimum-cost k paths from a source node to a destination node that has received considerable attention in the context of protecting a network against link/node failure. This result leads to the following general framework for a given application instance: Based on application requirement, select a set of shareability constraints, and then compute optimal solutions under these constraints. These optimal solutions form a space of candidate solutions. One can evaluate the candidate solutions in this space of manageable size by considering tradeoffs among cost, link shareability, and node shareability and select the best solution. To make this framework complete, we will show that all the 65 versions of the MCMPMS problem can be solved in polynomial time using a unified algorithmic approach.

II. Problem Description

We restrict our discussions to directed graphs. All our algorithms and claims are applicable to undirected graphs since undirected graphs can be converted to directed graphs easily. In the rest of the paper, the terms graph and network are used interchangeably, as are the terms of edge and link.

Let $G = (V, E)$ be a directed graph with node set V, link set E, and nonnegative *cost* $l(e)$ associated with each link $e = (u, v)$. Furthermore, we assume that G is simple; i.e., it has no self-loop or parallel links. Given a source node s and a destination node t in G, let $P = \{P_1, P_2, \ldots, P_k\}$ be a set of k paths (s-t paths) in graph $G = (V, E)$ from s to t.

For a link e, we define

$$\delta(e, P_i) = \begin{cases} 1, & e \in P_i \\ 0, & e \notin P_i \end{cases}$$

and

$$\delta(e, P) = \sum_{P_i \in P} \delta(e, P_i).$$

Then, $\delta(e, P)$ represents the number of times e appears in P. Without causing too much ambiguity, we may slightly abuse the notation $e \in P$ to refer to the case $\delta(e, P) \geq 1$, though P is the set of paths. We thus define

$$\Delta(P) = \sum_{e \in P} (\delta(e, P) - 1)$$

and

$$\delta(P) = \max_{e \in P} \{\delta(e, P) - 1\}.$$

$\Delta(P)$ is called the *total link shareability* of P, and $\delta(P)$ is called the *maximum link shareability* of P. Clearly, $\Delta(P) = 0$ if and only if $\delta(P) = 0$, and either $\Delta(P) = 0$ or $\delta(P) = 0$ indicates that all paths in P are link-disjoint.

Similarly, for a node v, we define

$$\gamma(v, P_i) = \begin{cases} 1, & v \in P_i \\ 0, & v \notin P_i \end{cases}$$

and

$$\gamma(v, P) = \sum_{P_i \in P} \gamma(v, P_i).$$

Then, $\gamma(v, P)$ represents the number of times node v appears in P. We again abuse the notation $v \in P$ to refer to the case $\gamma(v, P) \geq 1$. We define

$$\Gamma(P) = \sum_{v \in P, v \neq s, t} (\gamma(v, P) - 1)$$

and

$$\gamma(P) = \max_{v \in P, v \neq s, t} \{\gamma(v, P) - 1\}.$$

$\Gamma(P)$ is called the *total node shareability* of P, and $\gamma(P)$ is called the *maximum node shareability* of P. Clearly, $\Gamma(P) = 0$ if and only if $\gamma(P) = 0$, and either $\Gamma(P) = 0$ or $\gamma(P) = 0$ indicates that all paths in P are node-disjoint.

The total cost of k paths in P is defined as

$$l(P) = \sum_{P_i \in P} l(P_i)$$

where $l(P_i) = \sum_{e \in P_i} l(e)$.

The MCMPMS problem studied in this paper is to find a set of paths $P = \{P_1, P_2, \ldots, P_k\}$ in G with minimum $l(P)$, subject to various combinations of minimum $\Delta(P)$, minimum $\Gamma(P)$, minimum $\delta(P)$, and minimum $\gamma(P)$ as constraints.

Let $\langle C \rangle = \langle C_q, C_{q-1}, \ldots, C_1 \rangle$ be an ordered list of constraints. An optimization problem Π subject to ordered constraint list $\langle C \rangle$ is to find a solution $S(I)$ for an instance I of Π such that:

(1): $S(I)$ satisfies C_1;

(2): $S(I)$ satisfies C_2 subject to condition (1);
$\cdots$
$(q):\ S(I)$ satisfies C_q subject to condition $(q-1)$; and
$(q+1):\ S(I)$ has the optimal solution value among all solutions that satisfy C_q.

We call $\langle C\rangle$ an *ordered composite constraint*, and C_i's are the *component constraints* of $\langle C\rangle$. Constraints are defined recursively. An empty list is an ordered composite constraint; it is also called *empty constraint*. A single constraint is an ordered composite constraint. Then, an ordered composite constraint is an ordered pair of two ordered composite constraints. Having defined this recursive structure, we refer to ordered composite constraints simply as constraints.

Throughout this paper, we use the following notation:

$\langle\ \rangle$	empty constraint, i.e., no constraint on shareability.
$\langle\delta^*\rangle$	minimum δ constraint, which is also called the *minmax link shareability constraint*.
$\langle\gamma^*\rangle$	minimum γ constraint, which is also called the *minmax node shareability constraint*.
$\langle\Delta^*\rangle$	minimum Δ constraint, which is also called the *minsum link shareability constraint*.
$\langle\Gamma^*\rangle$	minimum Γ constraint, which is also called the *minsum node shareability constraint*.

These five constraints are called *basic shareability constraints*. $\langle\delta^*\rangle$ and $\langle\gamma^*\rangle$ are called *minmax constraints*, and $\langle\Delta^*\rangle$ and $\langle\Gamma^*\rangle$ are called *minsum constraints*. Let

$$C_{\text{minmax}}=\{\langle\delta^*\rangle,\langle\gamma^*\rangle\}$$

and

$$C_{\text{minsum}}=\{\langle\Delta^*\rangle,\langle\Gamma^*\rangle\}.$$

In general, we can have the following set of constraints: $Z = \{\langle X_i,\ldots,X_1\rangle|\langle X_1\rangle,\ldots,\langle X_i\rangle\in\{\langle\rangle,\ \langle\delta^*\rangle,\ \langle\gamma^*\rangle,\ \langle\Delta^*\rangle,\ \langle\Gamma^*\rangle\}\}$. Clearly, $\langle\langle\ \rangle,\langle\ \rangle\rangle=\langle\ \rangle$, $\langle\langle\ \rangle,\langle X\rangle\rangle=\langle X\rangle$, and $\langle\langle X\rangle,\langle\ \rangle\rangle=\langle X\rangle$. Unless $\langle C\rangle$ is itself an empty constraint, all component $\langle\ \rangle$ (empty) constraints in $\langle C\rangle$ are redundant and can therefore be eliminated. Similarly, if there is more than one occurrence of the same nonempty basic constraint in a given constraint $\langle C\rangle$, all but the rightmost are redundant and can therefore be eliminated from $\langle C\rangle$. For example, $\langle\langle\delta^*\rangle,\langle\Gamma^*\rangle,\langle\delta^*\rangle,\langle\gamma^*\rangle,\langle\Gamma^*\rangle\rangle$ can be simplified as $\langle\langle\delta^*\rangle,\langle\gamma^*\rangle,\langle\Gamma^*\rangle\rangle$. In the rest of this paper, we will not consider constraints with duplicated component constraints. For simplicity, we will remove delimiters "$\langle$" and "$\rangle$" of the component constraints whenever possible. For example, for convenience, we use $\langle\delta^*,\gamma^*,\Gamma^*\rangle$ to denote $\langle\langle\delta^*\rangle,\langle\gamma^*\rangle,\langle\Gamma^*\rangle\rangle$.

The MCMPMS problem we are considering is a problem with a prioritized hierarchy of optimization objectives. According to this feature of hierarchical optimization objective "constraints," we use ordered constraints to refer to ordered optimization objectives only for the sake of easy understanding. Readers should keep in mind that, strictly speaking, this is not a constrained optimization problem in the classical sense.

Given a nonempty constraint $\langle X\rangle=\langle X_i,\ldots,X_2,X_1\rangle$ $(1\le i\le 4)$, we say that component constraint $\langle X_b\rangle$ *succeeds* (*precedes*) component constraint $\langle X_a\rangle$ in $\langle X\rangle$ if $b>a$ $(b<a)$. We divide $\langle X\rangle$ into two ordered composite constraints $\langle X^{\text{minsum}}\rangle$ and $\langle X^{\text{minmax}}\rangle$, which consist of minsum (component) constraints and minmax (component) constraints of $\langle X\rangle$, respectively, in the order of their appearances in $\langle X\rangle$. In case of no minsum (resp. minmax) constraint in $\langle X\rangle$, $\langle X^{\text{minsum}}\rangle$ (resp. $\langle X^{\text{minmax}}\rangle$) is $\langle\ \rangle$, an empty constraint. Let

$$\langle X^{\text{normal}}\rangle=\langle\langle X^{\text{minsum}}\rangle,\langle X^{\text{minmax}}\rangle\rangle.$$

We call $\langle X^{\text{normal}}\rangle$ the *reference normal form* (RNF) of $\langle X\rangle$. For example, for $\langle X\rangle=\langle\gamma^*,\Delta^*,\delta^*,\Gamma^*\rangle$, $\langle X^{\text{normal}}\rangle=\langle\langle\Delta^*,\Gamma^*\rangle,\langle\gamma^*,\delta^*\rangle\rangle$. The notion of RNF is useful for simplifying our algorithms and proofs.

Let $S(n,r)$ denote the number of ordered r-tuples of distinct elements from an n-element set. Then, we have

$$|Z|=\sum_{i=0}^{4}S(4,i)=\sum_{i=0}^{4}\frac{4!}{i!}=65$$

different constraints.

III. Comparison of the Different Constraints

Given the 65 different versions of the problem, one may wonder which version to choose for a particular application. Roughly speaking, people can develop some general guidance to make the choice. For example, in the context of network survivability, if the probability of link failure is higher than that of node failure, either or both δ^* and Δ^* constraints should be satisfied first followed by either or both γ^* and Γ^* constraints. Otherwise, either or both γ^* and Γ^* constraints should be satisfied first. For the order of δ^* and Δ^*, if the probability of a single link failure is much higher than that of multiple (more than one) simultaneous link failures, δ^* should be satisfied before Δ^* since a single link failure may cause the infeasibility of multiple paths among those k paths. Otherwise, Δ^* should be satisfied first. Similarly, if the probability of a single node failure is much higher than that of multiple node failure, γ^* constraint should be satisfied before Γ^* constraint. Otherwise, Γ^* constraint should be satisfied first.

In the context of load balancing among links/nodes, we may put the δ^* and/or γ^* constraints in front of the Δ^* and/or Γ^* constraints if we care more about the extreme load of a single link/node, and put the Δ^* and/or Γ^* constraints in front of the δ^* and/or γ^* constraints if we care more about the average load of a single link/node.

In fact, our modeling framework provides an alternative and more flexible way to choose multiple paths than simply choosing a single version of the problem for a specific application. In a network, if the above general guidance cannot be applied due to the lack of clear network statistic information or an application is willing to make a tradeoff among cost, link shareability, and node shareability, then we can compare and choose among all possible solutions under the 65 cases. For such a purpose, we develop a unified algorithmic approach that can find optimal solutions for all 65 cases in polynomial time. We believe that this approach will be more appreciated than designing an algorithm for each individual case.

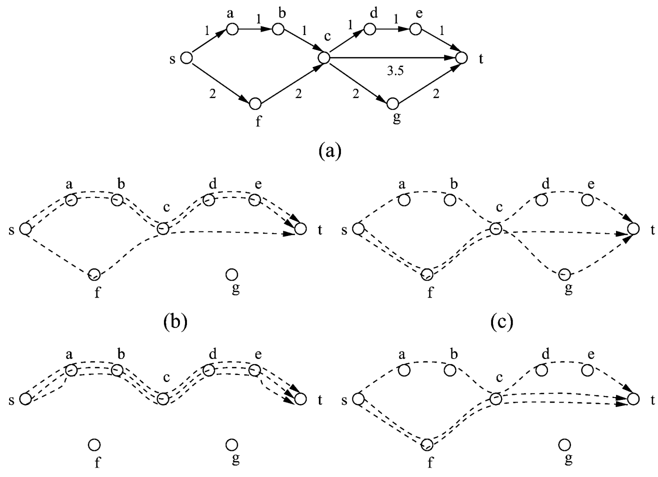


Fig. 1. Example 1 with $k = 3$. (a) The network. (b) Optimal solution under $\langle \delta^* \rangle$. (c) Optimal solution under $\langle \Delta^* \rangle$. (d) Optimal solution under $\langle \gamma^* \rangle$. (e) Optimal solution under $\langle \Gamma^* \rangle$.

We now use one example to shed some light on the motivation of investigating minsum, minmax, and mixed minmax and minsum constraints.

1) Example 1: Consider the network shown in Fig. 1, in which the value associated with a link is the length of the link. We want to find three s–t paths of minimum total cost (length).

Consider four different constraints and their corresponding optimal solutions, as shown in Table I.

- Observation 1: The four solutions are different.
- Observation 2: In terms of network reliability, assuming all links have identical failure probability and all nodes have identical failure probability:

  (2.1): The optimal solution [Fig. 1(c)] obtained under $\langle \Delta^* \rangle$ constraint is better than the optimal solution [Fig. 1(b)] obtained under $\langle \delta^* \rangle$ constraint, even though it has a larger cost (21.5 versus 19.5).

  (2.2): The optimal solution obtained [Fig. 1(e)] under minsum node shareability constraint $\langle \Gamma^* \rangle$ is better than the optimal solution [Fig. 1(d)] obtained under minmax node shareability constraint $\langle \gamma^* \rangle$, even though it has a larger cost (21 versus18).
- Observation 3: For this example, no solution with minmax node shareability smaller than 2 exists. Thus, in this case, the optimal solution under constraint $\langle \gamma^* \rangle$ is the same as the optimal solution under the empty constraint $\langle \rangle$. This is why the solution for $\langle \gamma^* \rangle$ has the minimum total cost.
- Observation 4: For this example, the optimal solution under constraint $\langle \delta^* \rangle$ is the same as the optimal solution under constraint $\langle \delta^*, \gamma^* \rangle$; the optimal solution under constraint $\langle \Delta^* \rangle$ is the same as the optimal solution under constraint $\langle \Delta^*, \gamma^* \rangle$; and the optimal solution under constraint $\langle \Gamma^* \rangle$ is the same as the optimal solution under constraint $\langle \Gamma^*, \gamma^* \rangle$.

Observation 2 of Example 1 is somewhat counterintuitive and shows that, given a problem instance consisting of a network G, nodes s and t in G, and a k value, it is not easy to find a k-path solution that is the best in terms of network reliability and survivability without comparing the solutions obtained under different shareability constraints. Observations 3 and 4 lead to a question: Are the optimal solutions under any two different ordered composite shareability constraints formed using five basic shareability constraints the same for all valid problem instances?

TABLE I
SOLUTION COMPARISON FOR EXAMPLE 1

Figures	Constraints	$\delta(P)$	$\Delta(P)$	$\gamma(P)$	$\Gamma(P)$	$l(P)$
Fig.1(b)	$\langle \delta^* \rangle$	1	6	2	6	19.5
Fig.1(c)	$\langle \Delta^* \rangle$	1	2	2	3	21.5
Fig.1(d)	$\langle \gamma^* \rangle$	2	12	2	10	18
Fig.1(e)	$\langle \Gamma^* \rangle$	1	3	2	3	21

For the MCMPMS problem, we say that two constraints $\langle X \rangle$ and $\langle Y \rangle$ are equivalent if and only if any optimal solution obtained under $\langle X \rangle$ is also an optimal solution obtained under $\langle Y \rangle$ for any network. In [25], we show that these 65 constraints are mutually (pair-wisely) inequivalent.

Example 1 and many cases in the inequivalence proof in [25] illustrate that, for a given MCMPMS problem instance, the solutions obtained under minsum shareability constraints are not always worse than solutions obtained under minmax shareability constraints, and vice versa, for a given application. The feasible solution space for the MCMPMS problem is very large. It is not easy to find a k-path solution that is the best for a particular application without comparing some alternative solutions. The size of the space of alternative solutions is also too large. Our result leads to the following general framework for a given application instance: Based on application requirement, select a set of shareability constraints. Compute optimal solutions under these constraints. These optimal solutions form a space of candidate solutions. One can evaluate the candidate solutions in this space of manageable size by considering tradeoffs among cost, link shareability, and node shareability and select the best solution. This framework is consistent with the common approach of restricting the size of solution space for many complicated problems in network design, planning, and management.

IV. Algorithm Scheme I

In this section, we focus on 25 constraints listed in Table II, i.e., the composite constraints with minmax component constraints (if any) having priority higher than minsum component constraints (if any). We present a unified algorithm scheme, *Algorithm Scheme I*, for the versions of the MCMPMS problem corresponding to the constraints of Table II.

The version corresponding to constraint $\langle C_0 \rangle$ can be reduced to a problem of finding minimum-cost network flow (MCNF) of flow value k, which can be solved by a MCNF algorithm, such as the *successive shortest path algorithm* [1], in $O(k \cdot (|E| + |V| \log |V|))$ time. The minimum-cost flow algorithm also terminates in case that no s–t path exists in G, reporting this fact. In this and next section, we assume that there is at least one s–t path in G.

Each constraint $\langle C \rangle$ in the rest of Table II can be partitioned into two component constraints $\langle C' \rangle$ and $\langle C'' \rangle$ such that $\langle C' \rangle$ contains minmax constraints and $\langle C'' \rangle$ contains minsum constraints. Then, $\langle C \rangle$ can be considered as a constraint $\langle C'', C' \rangle$ formed by concatenating $\langle C'' \rangle$ and $\langle C' \rangle$. By the definition of RNF, we have

$$\langle C \rangle = \langle \langle C'' \rangle, \langle C' \rangle \rangle = \langle \langle C^{\text{minsum}} \rangle, \langle C^{\text{minmax}} \rangle \rangle = \langle C^{\text{normal}} \rangle.$$

TABLE II
25 DIFFERENT CONSTRAINTS FOR THE MINIMUM-COST k-PATH PROBLEM

	constraints		constraints
$\langle C_0\rangle$	$\langle\ \rangle$		
$\langle C_1\rangle$	$\langle\delta^*\rangle = \langle\langle\ \rangle, \langle\delta^*\rangle\rangle$	$\langle C_{13}\rangle$	$\langle\Gamma^*, \gamma^*, \delta^*\rangle = \langle\langle\Gamma^*\rangle, \langle\gamma^*, \delta^*\rangle\rangle$
$\langle C_2\rangle$	$\langle\gamma^*\rangle = \langle\langle\ \rangle, \langle\gamma^*\rangle\rangle$	$\langle C_{14}\rangle$	$\langle\langle\Gamma^*\rangle, \langle\delta^*, \gamma^*\rangle\rangle = \langle\langle\Gamma^*\rangle, \langle\delta^*, \gamma^*\rangle\rangle$
$\langle C_3\rangle$	$\langle\gamma^*, \delta^*\rangle\rangle = \langle\langle\ \rangle, \langle\gamma^*, \delta^*\rangle\rangle$	$\langle C_{15}\rangle$	$\langle\Gamma^*, \Delta^*\rangle\rangle = \langle\langle\Gamma^*, \Delta^*\rangle, \langle\ \rangle\rangle$
$\langle C_4\rangle$	$\langle\delta^*, \gamma^*\rangle = \langle\langle\ \rangle, \langle\delta^*, \gamma^*\rangle\rangle$	$\langle C_{16}\rangle$	$\langle\Gamma^*, \Delta^*, \delta^*\rangle = \langle\langle\Gamma^*, \Delta^*\rangle, \langle\delta^*\rangle\rangle$
$\langle C_5\rangle$	$\langle\Delta^*\rangle = \langle\langle\Delta^*\rangle, \langle\ \rangle\rangle$	$\langle C_{17}\rangle$	$\langle\Gamma^*, \Delta^*, \gamma^*\rangle = \langle\langle\Gamma^*, \Delta^*\rangle, \langle\gamma^*\rangle\rangle$
$\langle C_6\rangle$	$\langle\Delta^*, \delta^*\rangle = \langle\langle\Delta^*\rangle, \langle\delta^*\rangle\rangle$	$\langle C_{18}\rangle$	$\langle\Gamma^*, \Delta^*, \gamma^*, \delta^*\rangle = \langle\langle\Gamma^*, \Delta^*\rangle, \langle\gamma^*, \delta^*\rangle\rangle$
$\langle C_7\rangle$	$\langle\Delta^*, \gamma^*\rangle = \langle\langle\Delta^*\rangle, \langle\gamma^*\rangle\rangle$	$\langle C_{19}\rangle$	$\langle\Gamma^*, \Delta^*, \delta^*, \gamma^*\rangle = \langle\langle\Gamma^*, \Delta^*\rangle, \langle\delta^*, \gamma^*\rangle\rangle$
$\langle C_8\rangle$	$\langle\Delta^*, \gamma^*, \delta^*\rangle = \langle\langle\Delta^*\rangle, \langle\gamma^*, \delta^*\rangle\rangle$	$\langle C_{20}\rangle$	$\langle\Delta^*, \Gamma^*\rangle = \langle\langle\Delta^*, \Gamma^*\rangle, \langle\ \rangle\rangle$
$\langle C_9\rangle$	$\langle\Delta^*, \delta^*, \gamma^*\rangle = \langle\langle\Delta^*\rangle, \langle\delta^*, \gamma^*\rangle\rangle$	$\langle C_{21}\rangle$	$\langle\Delta^*, \Gamma^*, \delta^*\rangle = \langle\langle\Delta^*, \Gamma^*\rangle, \langle\delta^*\rangle\rangle$
$\langle C_{10}\rangle$	$\langle\Gamma^*\rangle = \langle\langle\Gamma^*\rangle, \langle\ \rangle\rangle$	$\langle C_{22}\rangle$	$\langle\Delta^*, \Gamma^*, \gamma^*\rangle = \langle\langle\Delta^*, \Gamma^*\rangle, \langle\gamma^*\rangle\rangle$
$\langle C_{11}\rangle$	$\langle\Gamma^*, \delta^*\rangle = \langle\langle\Gamma^*\rangle, \langle\delta^*\rangle\rangle$	$\langle C_{23}\rangle$	$\langle\Delta^*, \Gamma^*, \gamma^*, \delta^*\rangle = \langle\langle\Delta^*, \Gamma^*\rangle, \langle\gamma^*, \delta^*\rangle\rangle$
$\langle C_{12}\rangle$	$\langle\Gamma^*, \gamma^*\rangle = \langle\langle\Gamma^*\rangle, \langle\gamma^*\rangle\rangle$	$\langle C_{24}\rangle$	$\langle\Delta^*, \Gamma^*, \delta^*, \gamma^*\rangle = \langle\langle\Delta^*, \Gamma^*\rangle, \langle\delta^*, \gamma^*\rangle\rangle$

That is, $\langle C\rangle$ and $\langle C^{\text{normal}}\rangle$ are identical. Based on RNFs, we divide the constraints $\langle C_1\rangle$ to $\langle C_{24}\rangle$ in Table II into five classes as follows.

Class 1: $\langle C^{\text{minsum}}\rangle$ is empty, which contains $\langle C_1\rangle$ to $\langle C_4\rangle$.
Class 2: $\langle C^{\text{minsum}}\rangle$ is $\langle\Delta^*\rangle$, which contains $\langle C_5\rangle$ to $\langle C_9\rangle$.
Class 3: $\langle C^{\text{minsum}}\rangle$ is $\langle\Gamma^*\rangle$, which contains $\langle C_{10}\rangle$ to $\langle C_{14}\rangle$.
Class 4: $\langle C^{\text{minsum}}\rangle$ is $\langle\Gamma^*, \Delta^*\rangle$, which contains $\langle C_{15}\rangle$ to $\langle C_{20}\rangle$.
Class 5: $\langle C^{\text{minsum}}\rangle$ is $\langle\Delta^*, \Gamma^*\rangle$, which contains $\langle C_{20}\rangle$ to $\langle C_{24}\rangle$.

For example, $\langle C_{22}\rangle = \langle\Delta^*, \Gamma^*, \gamma^*\rangle = \langle\langle C_{20}\rangle, \langle C_2\rangle\rangle \quad \langle\langle\Delta^*, \Gamma^*\rangle, \langle\gamma^*\rangle\rangle$, and it is in Class 5. The version corresponding to constraint $\langle C_{22}\rangle$ is to find a set P^* of k paths from s to t such that:

(1) P^* has minmax node shareability;
(2) P^* has minsum node shareability subject to condition (1);
(3) P^* has minsum link shareability subject to condition (2);
(4) $l(P^*) = \min\{l(P') | P' \text{ is a solution that satisfies (3)}\}$.

This classification is useful in the analysis of our algorithm scheme.

Algorithm Scheme I, which is used to generate different algorithms by reducing the problem of finding a set P^* of minimum-cost k s–t paths in G to finding a minimum-cost s–t flow f^* of value k in G'' where the construction of G'' is introduced shortly. We use $|f|$ to denote the flow value of f. An s–t flow f of value k is called a *k-flow* if $|f| = k$. Algorithm Scheme I has three steps.

Step 1: Compute minmax link shareability k^L and/or minmax node shareability k^N if needed, and construct flow network $G'' = (V'', E'')$ from $G = (V, E)$ according to shareability requirement.
Step 2: Find a minimum-cost k-flow f^* from s to t in G''.
Step 3: Construct a set P^* of k s–t paths in G from the flow f^* in G''.

For Step 1, two transformations, *TRANSFORM-1* and *TRANSFORM-2*, are needed.

TRANSFORM-1: Obtain $G' = (V', E')$ from $G = (V, E)$ by node splitting (or node doubling) as follows: Replace each node v that is neither s nor t by two nodes v and v' such that all links ending at v in G end at v in G' and all links originating from v in G originate from v', and then add a link (v, v') (see Figs. 2 and 4).

TRANSFORM-2: Obtain graph $G'' = (V'', E'')$ from $G' = (V', E')$ by *link splitting* (or link doubling) as follows: Replace each link e in G' by two parallel links with the same direction of e. We denote the two links generated from a link (u', v) in G' corresponding to link (u, v) in G by (u', v) and $\overline{(u', v)}$, which are called the *primary link-generated u–v link* (or simply, *primary u–v link*) and the *secondary link-generated u–v link* (or simply, *secondary u–v link*), respectively (note: u' can be source node s). We denote the two links generated from a link (v, v') in G' corresponding to node v in G by (v, v') and $\overline{(v, v')}$, which are called the *primary node-generated v–v link* (or simply, *primary v link*) and the *secondary node-generated v–v link* (or simply, *secondary v link*), respectively (see Figs. 3 and 4).

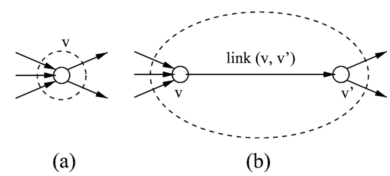


Fig. 2. Node splitting. (a) Node v in G. (b) v is replaced by two nodes and a link (v, v').

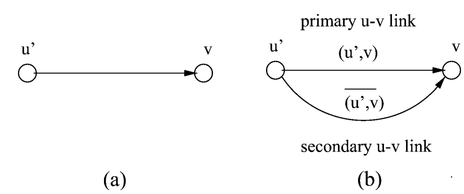


Fig. 3. Link splitting. (a) Original link in G'. (b) Two links obtained for the link of (a).

The purpose of node splitting is to reduce node sharing to link sharing, and the purpose of introducing link splitting is to exclude sharing as much as possible by assigning secondary u–v link and v link considerable large length.

Minmax link shareability k^L value and minmax node shareability k^N value, which are useful in defining the capacities of links in G'', are computed by applying the following procedure *MinMax-Search* to G' with $flag$ set to 0 and 1, respectively.

procedure *MinMax-Search* $(G', flag, k, k')$
begin
E'_1 := the set of all links in E' that are resulted from node splitting in the construction of G' by *TRANSFORM-1*;
$E'_2 := E' - E'_1$;
if flag $= 0$ **then** $E^* := E'_2$ **else** $E^* := E'_1$;
assign capacity k' to all links in $E' - E^*$;

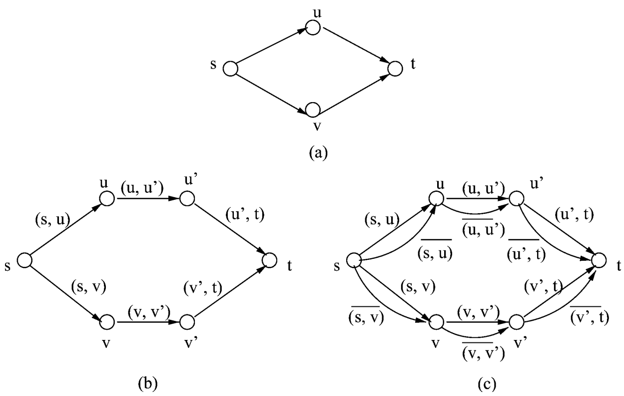


Fig. 4. An example. (a) Original graph. (b) Graph G' constructed from G by *TRANSFORM-1*. (c) Graph G'' constructed from G' by *TRANSFORM-2*. Capacities for links of (b) and (*cost, capacity*) pairs of (c) are not shown.

$low := 0; high := k; lim := \lfloor k/2 \rfloor;$
while $low \neq high$ **do**
  for each link $e \in E^*$ **do** assign e capacity lim
  end-for;
  find the maximum flow f in G' from s to t;
  if $|f| < k$ **then** $low := lim$ and $lim := low + \lceil (high - low)/2 \rceil$;
  else $high := lim$ and $lim := low + \lfloor (high - low)/2 \rfloor$;
end-while
return $(lim - 1)$;
end

For a nonempty $\langle C^{\text{minmax}} \rangle$, either $\langle C^{\text{minmax}} \rangle = \langle minmax_1 \rangle$ or $\langle C^{\text{minmax}} \rangle = \langle \langle minmax_2 \rangle, \langle minmax_1 \rangle \rangle$ depending on the number of basic minmax constraints in $\langle C^{\text{minmax}} \rangle$. $\langle C^{\text{minsum}} \rangle = \langle minsum_1 \rangle$ and $\langle C^{\text{minsum}} \rangle = \langle \langle minsum_2 \rangle, \langle minsum_1 \rangle \rangle$ are similarly defined.

For all constraints $\langle C^{\text{minsum}}, C^{\text{minmax}} \rangle$ in Table II with nonempty $\langle C^{\text{minmax}} \rangle$, the minmax shareability values with respect to $\langle C^{\text{minmax}} \rangle$ of $\langle C \rangle$ are computed by the following procedure *MinMax-Compute*.

procedure *MinMax-Compute* (G', k)
begin
  case
    $\langle C^{\text{minmax}} \rangle = \langle C_1 \rangle$:
      $k_1^L := MinMax - Search(G', 0, k, k);$
    $\langle C^{\text{minmax}} \rangle = \langle C_2 \rangle$:
      $k_1^N := MinMax - Search(G', 1, k, k);$
    $\langle C^{\text{minmax}} \rangle = \langle C_3 \rangle$:
      $k_1^L := MinMax - Search(G', 0, k, k);$
      $k_2^N := MinMax - Search(G', 1, k, k_1^L);$
    $\langle C^{\text{minmax}} \rangle = \langle C_4 \rangle$:
      $k_1^N := MinMax - Search(G', 1, k, k);$
      $k_2^L := MinMax - Search(G', 0, k, k_1^N);$
  end-case
end

It is important to note that procedure *MinMax-Search* is invoked twice for $\langle C_3 \rangle$ and $\langle C_4 \rangle$. For $\langle C_3 \rangle$ (resp. $\langle C_4 \rangle$), k_1^L (resp. k_1^N) is computed first, and the value of k_1^L (resp. k_1^N) affects the value of k_2^N (resp. k_2^L) that is computed by the second call to *MinMax-Search*. Thus, the k^L and k^N values computed for $\langle C_3 \rangle$ and $\langle C_4 \rangle$ for the same network G may be different. The subscript l of k_l^L and k_l^N is used to signify that the value is for $\langle minmax_l \rangle$. Similarly, *MinMax-Search* is invoked twice for $\langle C_8 \rangle$, $\langle C_9 \rangle$, $\langle C_{13} \rangle$, $\langle C_{14} \rangle$, $\langle C_{18} \rangle$, $\langle C_{19} \rangle$, $\langle C_{23} \rangle$, and $\langle C_{24} \rangle$.

In order for Step 2 to compute a minimum-cost k-flow f^* in G'', each link in G'' is assigned a $(cost, capacity)$ pair. More specifically, for each link $e = (u, v)$ in G, its corresponding primary u–v link (u', v) and secondary u–v link $\overline{(u', v)}$ in G'' are assigned pair $(cost(u', v), capacity(u', v))$ and $(cost(\overline{u', v}), capacity(\overline{u', v}))$, respectively. We call the ordered pair of $(cost, capacity)$ pairs

$$[(cost(u', v), capacity(u', v)), (cost(\overline{u', v}), capacity(\overline{u', v}))]$$

the *cost-capacity assignment for link shareability*. Similarly, for each node v in G, its corresponding primary v link (v, v') and secondary v link $\overline{(v, v')}$ are assigned pair $(cost(v, v'), capacity(v, v'))$ and $(cost(\overline{v, v'}), capacity(\overline{v, v'}))$, respectively. We have the *cost-capacity assignment for node shareability*

$$[(cost(v, v'), capacity(v, v')), (cost(\overline{v, v'}), capacity(\overline{v, v'}))].$$

For each constraint in Table II, we have a pair of cost-capacity assignments

$$[(cost(u', v), capacity(u', v)), (cost(\overline{u', v}), capacity(\overline{u', v}))],$$
$$[(cost(v, v'), capacity(v, v')), (cost(\overline{v, v'}), capacity(\overline{v, v'}))].$$

Since (u', v) and $\overline{(u', v)}$ are link-generated and (v, v') and $\overline{(v, v')}$ are node-generated, we simply use

$$[(cost_1^L(e), cap_1^L(e)), (cost_2^L(e), cap_2^L(e))],$$
$$[(cost_1^N(v), cap_1^N(v)), (cost_2^N(v), cap_2^N(v))]$$

to denote this pair of assignments.

Let $l_{\max} = \max_{e \in E}\{l(e)\}$. Assume that $l_{\max} > 1$. If this is not true, we simply choose a constant c to scale up the cost of each link e from $l(e)$ to $c \cdot l(e)$ so that $l_{\max} > 1$. Define $M_2 = k \cdot |V| \cdot l_{\max}$ and $M_1 = M_2^2 = k^2 \cdot |V|^2 \cdot l_{\max}^2$. Then, all versions of the MCMPMS problem considered are reduced to finding minimum-cost flows, and we present algorithms by only presenting the costs and capacities assigned to the links in G''. We provide cost-capacity assignments for all the 25 constraints of Table II in Table III.

The following general rules are used to assign cost and capacity of each link in G''. Any primary u–v link is assigned cost $l(u, v)$ and capacity 1, and any primary v link is assigned cost 1 and capacity 1. Any secondary u–v link is assigned cost $l(u, v)$ and capacity $k - 1$, and any secondary v link is assigned cost 1 and capacity $k - 1$. This cost/capacity assignment is for the empty constraint. In this case, the cost of a unit flow in the primary u–v link (resp. v link) and the cost of a unit flow in the secondary u–v link (resp. v link) in G'' corresponding to link (u', v) (resp. (v, v')) in G' are the same. The total capacity of the primary and secondary u–v links (resp. v links) corresponding to link (u', v) (resp. (v, v')) in G' is k. For the remaining 24 constraints, we make changes in the cost of secondary u–v links (resp. v links) to enforce minsum link (resp. node) shareability

TABLE III
$(cost, capacity)$ ASSIGNMENTS IN G'' FOR THE 25 VERSIONS OF THE k-PATH PROBLEM OF TABLE II

	$[(cost_1^L(e), cap_1^L(e)), (cost_2^L(e), cap_2^L(e))]$, $[(cost_1^N(v), cap_1^N(v)), (cost_2^N(v), cap_2^N(v))]$		$[(cost_1^L(e), cap_1^L(e)), (cost_2^L(e), cap_2^L(e))]$, $[(cost_1^N(v), cap_1^N(v)), (cost_2^N(v), cap_2^N(v))]$
$\langle C_0 \rangle$	$[(l(e),1),(l(e),k-1)],[(0,1),(0,k-1)]$		
$\langle C_1 \rangle$	$[(l(e),1),(l(e),k_1^L)],[(0,1),(0,k-1)]$	$\langle C_{13} \rangle$	$[(l(e),1),(l(e),k_1^L)],[(0,1),(M_1,k_2^N)]$
$\langle C_2 \rangle$	$[(l(e),1),(l(e),k-1)],[(0,1),(0,k_1^N)]$	$\langle C_{14} \rangle$	$[(l(e),1),(l(e),k_2^L)],[(0,1),(M_1,k_1^N)]$
$\langle C_3 \rangle$	$[(l(e),1),(l(e),k_1^L)],[(0,1),(0,k_2^N)]$	$\langle C_{15} \rangle$	$[(l(e),1),(l(e)+M_1,k-1)],[(0,1),(M_2,k-1)]$
$\langle C_4 \rangle$	$[(l(e),1),(l(e),k_2^L)],[(0,1),(0,k_1^N)]$	$\langle C_{16} \rangle$	$[(l(e),1),(l(e)+M_1,k_1^L)],[(0,1),(M_2,k-1)]$
$\langle C_5 \rangle$	$[(l(e),1),(l(e)+M_1,k-1)],[(0,1),(0,k-1)]$	$\langle C_{17} \rangle$	$[(l(e),1),(l(e)+M_1,k-1)],[(0,1),(M_2,k_1^N)]$
$\langle C_6 \rangle$	$[(l(e),1),(l(e)+M_1,k_1^L)],[(0,1),(0,k-1)]$	$\langle C_{18} \rangle$	$[(l(e),1),(l(e)+M_1,k_1^L)],[(0,1),(M_2,k_2^N)]$
$\langle C_7 \rangle$	$[(l(e),1),(l(e)+M_1,k-1)],[(0,1),(0,k_1^N)]$	$\langle C_{19} \rangle$	$[(l(e),1),(l(e)+M_1,k_2^L)],[(0,1),(M_2,k_1^N)]$
$\langle C_8 \rangle$	$[(l(e),1),(l(e)+M_1,k_1^L)],[(0,1),(0,k_2^N)]$	$\langle C_{20} \rangle$	$[(l(e),1),(l(e)+M_2,k-1)],[(0,1),(M_1,k-1)]$
$\langle C_9 \rangle$	$[(l(e),1),(l(e)+M_1,k_2^L)],[(0,1),(0,k_1^N)]$	$\langle C_{21} \rangle$	$[(l(e),1),(l(e)+M_2,k_1^L)],[(0,1),(M_1,k-1)]$
$\langle C_{10} \rangle$	$[(l(e),1),(l(e),k-1)],[(0,1),(M_1,k-1)]$	$\langle C_{22} \rangle$	$[(l(e),1),(l(e)+M_2,k-1)],[(0,1),(M_1,k_1^N)]$
$\langle C_{11} \rangle$	$[(l(e),1),(l(e),k_1^L)],[(0,1),(M_1,k-1)]$	$\langle C_{23} \rangle$	$[(l(e),1),(l(e)+M_2,k_1^L)],[(0,1),(M_1,k_2^N)]$
$\langle C_{12} \rangle$	$[(l(e),1),(l(e),k-1)],[(0,1),(M_1,k_1^N)]$	$\langle C_{24} \rangle$	$[(l(e),1),(l(e)+M_2,k_2^L)],[(0,1),(M_1,k_1^N)]$

by introducing additional cost M, where $M = M_1$ or M_2, and/or make changes in the capacity of secondary u–v links (resp. v links) to enforce minmax link (resp. node) shareability by replacing $k-1$ by k^L (resp. k^N). Specifically, for a constraint $\langle C \rangle$:

- if $\langle C^{\text{minsum}} \rangle = \langle\ \rangle$, then any secondary u–v link is assigned cost $l(u,v)$;
- if $\langle C^{\text{minsum}} \rangle = \langle \Delta^* \rangle$, then any secondary u–v link is assigned cost $l(u,v) + M_1$;
- if $\langle C^{\text{minsum}} \rangle = \langle \Gamma^* \rangle$, then any secondary v link is assigned cost M_1;
- if $\langle C^{\text{minsum}} \rangle = \langle \Gamma^*, \Delta^* \rangle$, then any secondary u–v link is assigned cost $l(u,v) + M_1$ and any secondary v link is assigned cost M_2;
- if $\langle C^{\text{minsum}} \rangle = \langle \Delta^*, \Gamma^* \rangle$, then any secondary u–v link is assigned cost $l(u,v) + M_2$ and any secondary v link is assigned cost M_1.
- if $\langle C^{\text{minmax}} \rangle = \langle\ \rangle$, then any secondary u–v link is assigned capacity $k-1$;
- if $\langle C^{\text{minmax}} \rangle = \langle \delta^* \rangle$, then any secondary u–v link is assigned capacity k_1^L;
- if $\langle C^{\text{minmax}} \rangle = \langle \gamma^* \rangle$, then any secondary v link is assigned capacity k_1^N;
- if $\langle C^{\text{minmax}} \rangle = \langle \gamma^*, \delta^* \rangle$, then any secondary u–v link is assigned capacity k_1^L and any secondary v link is assigned capacity k_2^N;
- if $\langle C^{\text{minmax}} \rangle = \langle \delta^*, \gamma^* \rangle$, then any secondary u–v link is assigned capacity k_2^L and any secondary v link is assigned capacity k_1^N.

M_2 is selected to be larger than the total cost (length) of any k s–t (loop-free) paths, and M_1 is selected to be larger than $k \cdot |V|$ times of M_2. The cost of each secondary link, with M_1 and M_2 possibly included, determines the "power" of the link to "push away" or "attract" flow (paths), and the capacity of the secondary link restricts the maximum amount (number) of flow (paths) that can use the link. Smaller M_1 and M_2 values can be used. Our selection of M_1 and M_2 values also takes easy correctness proof into consideration.

In Step 2, a MCNF algorithm is applied to the flow network G'' constructed in Step 1 to find minimum-cost s–t k-flow f^*. In Step 3, the integral flow of value k in G'' guaranteed by integrality of the link capacities is decomposed into k s–t paths, and this decomposition is a simple procedure that utilizes the flow conservation property.

Theorem 1: For any graph $G = (V, E)$ with nonnegative link cost, source s and destination t in V, if there exists an s–t path in G, then for each of the shareability constraints $\langle C_0 \rangle$ to $\langle C_{24} \rangle$ in Table II, an algorithm corresponding to the constraint can be generated from Algorithm Scheme I, and this algorithm computes a set of k s–t paths $P^* = \{P_1^*, P_2^*, \ldots, P_k^*\}$ such that $l(P^*)$ is minimum subject to the shareability constraint.

Proof: For constraint $\langle C_0 \rangle$, the theorem obviously holds. We prove the theorem by considering all five classes. For constraints in Class 1, our Algorithm Scheme I first finds minmax link shareability k^L and/or minmax node shareability k^N, which are used to restrict the feasible solution space. The MCNF algorithm finds an optimal solution within this restricted space, and the solution found is certainly optimal.

For Class 2, the solutions have to satisfy minmax constraints, which are enforced by k^L and/or k^N. Suppose for the sake of contradiction the theorem is not true, then there exists a different set of k paths $P' = \{P'_1, P'_2, \ldots, P'_k\}$ in G, such that one of the following conditions holds:

(1) $\Delta(P') < \Delta(P^*)$;

(2) $\Delta(P') = \Delta(P^*)$, and $l(P') < l(P^*)$.

We create the unique network flow f' in G'' according to P' as follows. Initially let $f'(e) = 0$ for every link in G''. Then, for each link $e = (u,v) \in P'$, let $f'(u',v) = 1$ and $f'\overline{(u',v)} = \delta(e, P') - 1$, where $\delta(e, P')$ is the number of times link e is used in paths of P' (see Section II for its formal definition).

Let f^* denote the flow corresponding to P^* in G''. Note that $\delta(e, P^*) - 1$ is exactly the value of f^* on $\overline{e} = \overline{(u,v)}$. Let $c(f^*)$ denote the cost of flow f^* in G''. We have

$$\begin{aligned} c(f^*) &= \sum_{e \in P^*} l(e) + \sum_{e \in P^*} (\delta(e, P^*) - 1) \cdot (M_1 + l(e)) \\ &= \sum_{e \in P^*} \delta(e, P^*) l(e) + M_1 \cdot \sum_{e \in P^*} (\delta(e, P^*) - 1) \\ &= \sum_{e \in P^*} \delta(e, P^*) l(e) + M_1 \cdot \Delta(P^*). \end{aligned}$$

Similarly, we have $\Delta(P') = \sum_{e \in P'} (\delta(e, P') - 1)$, and $c(f') = \sum_{e \in P'} \delta(e, P') l(e) + M_1 \cdot \Delta(P')$. Then

$$\begin{aligned} c(f^*) - c(f') = &\left(\sum_{e \in P^*} \delta(e, P^*) l(e) - \sum_{e \in P'} \delta(e, P') l(e) \right) \\ &+ M_1 \cdot (\Delta(P^*) - \Delta(P')). \end{aligned}$$

Since $0 \leq \sum_{e\in P^*}\delta(e,P^*)l(e) < M_1$ and $0 \leq \sum_{e\in P'}\delta(e,P')l(e) < M_1$, $\sum_{e\in P^*}\delta(e,P^*)l(e) - \sum_{e\in P'}\delta(e,P')l(e) > -M_1$. We have two possibilities.

- Case (1): $\Delta(P') < \Delta(P^*)$. Then, $\Delta(P^*) - \Delta(P') \geq 1$. Thus, $c(f^*) - c(f') = (\sum_{e\in P^*}\delta(e,P^*)l(e) - \sum_{e\in P'}\delta(e,P')l(e)) + M_1\cdot(\Delta(P^*)-\Delta(P')) > 0$, which contradicts the assumption that f^* is a minimum-cost flow.
- Case (2): $\Delta(P') = \Delta(P^*)$ and $l(P') < l(P^*)$. Then, we have $c(f^*)-c(f') = l(P^*)-l(P') > 0$, which contradicts the assumption that f^* is a minimum-cost flow.

This completes the proof for Class 2.

The proof for Class 3 is about the same as that for Class 2, except that $c(f^*) = \sum_{e\in P^*} l(e) + M_1\cdot\Gamma(P^*)$ and $c(f') = \sum_{e\in P^*} l(e) + M_1\cdot\Gamma(P')$.

For Class 4, the solutions have to satisfy minmax constraints, which are enforced by k^L and/or k^N. Suppose for the sake of contradiction the theorem is not true, then there exists a different set of k paths from s to t, $P = \{P'_1, P'_2, \ldots, P'_k\}$ such that one of the following conditions must hold:

(1) $\Delta(P') < \Delta(P^*)$;
(2) $\Delta(P') = \Delta(P^*)$, and $\Gamma(P') < \Gamma(P^*)$;
(3) $\Delta(P') = \Delta(P^*), \Gamma(P') = \Gamma(P^*)$, and $l(P') < l(P^*)$.

We have

$$\begin{aligned}c(f^*) =& \sum_{v\in P^*, v\neq s,t}(\gamma(v,P^*)-1)\cdot M_2 + \sum_{e\in P^*} l(e)\\ &+ \sum_{e\in P^*}(\delta(e,P^*)-1)\cdot(M_1+l(e))\\ =&\,\Gamma(P^*)\cdot M_2 + \sum_{e\in P^*}\delta(e,P^*)l(e)\\ &+ \sum_{e\in P^*}(\delta(e,P^*)-1)\cdot M_1\\ =&\, M_2\cdot\Gamma(P^*) + M_1\cdot\Delta(P^*) + \sum_{e\in P^*}\delta(e,P^*)l(e).\end{aligned}$$

We create the unique network flow f' in G'' according to P' as follows. Initially, let $f'(e) = 0$ for every link in G''. Then, for each link $e = (u,v) \in P'$, let $f'(u',v) = 1$ and $f'\overline{(u',v)} = \delta(e,P') - 1$, where $\delta(e,P')$ is the number of times link e is used in paths of P'. For each node $v \in P'$ such that v is neither s nor t, let $f'(v,v') = 1$ and $f'\overline{(v,v')} = \gamma(v,P') - 1$, where $\gamma(e,P')$ is the number of times node v is used in paths of P' (see Section II for its formal definition). We have

$$c(f') = M_2\cdot\Gamma(P') + M_1\cdot\Delta(P') + \sum_{e\in P'}\delta(e,P')l(e).$$

Then

$$\begin{aligned}c(f^*) - c(f') =& M_2\cdot(\Gamma(P^*)-\Gamma(P'))\\ &+ M_1\cdot(\Delta(P^*)-\Delta(P'))\\ &+\left(\sum_{e\in P^*}\delta(e,P^*)l(e) - \sum_{e\in P'}\delta(e,P')l(e)\right).\end{aligned}$$

For any set of k s–t paths with minimum total cost, there is no loop of positive cost in any path. Then, $0 \leq \Gamma(P^*) \leq k\cdot(|V|-2)$ and $0 \leq \Gamma(P') \leq k\cdot(|V|-2)$, and $\Gamma(P^*) - \Gamma(P') > -k\cdot(|V|-2)$ and $M_2\cdot(\Gamma(P^*)-\Gamma(P')) > -M_1 + 2kM_2$. Since $\sum_{e\in P^*}\delta(e,P^*)l(e) \leq k\cdot(|V|-1)l_{\max} < M_2$ and $\sum_{e\in P'}\delta(e,P')l(e) \leq k\cdot(|V|-1)l_{\max} < M_2$, we have $\sum_{e\in P^*}\delta(e,P^*)l(e) - \sum_{e\in P'}\delta(e,P')l(e) > -M_2$. We have three possibilities.

Case (1): $\Delta(P') < \Delta(P^*)$. Then, $\Delta(P^*) - \Delta(P') \geq 1$, and

$$\begin{aligned}c(f^*) - c(f') =& M_2\cdot(\Gamma(P^*)-\Gamma(P'))\\ &+ M_1\cdot(\Delta(P^*)-\Delta(P'))\\ &+\left(\sum_{e\in P^*}\delta(e,P^*)l(e) - \sum_{e\in P'}\delta(e,P')l(e)\right)\\ \geq& M_2\cdot(\Gamma(P^*)-\Gamma(P')) + M_1\\ &+\left(\sum_{e\in P^*}\delta(e,P^*)l(e) - \sum_{e\in P'}\delta(e,P')l(e)\right)\\ >&(-M_1+2kM_2) + M_1 - M_2\\ =&(2k-1)M_2 > 0\end{aligned}$$

which contradicts the assumption that f^* is a minimum-cost flow.

Case (2): $\Delta(P') = \Delta(P^*)$, and $\Gamma(P') < \Gamma(P^*)$. Then

$$\begin{aligned}c(f^*) - c(f') =& M_2\cdot(\Gamma(P^*)-\Gamma(P'))\\ &+\left(\sum_{e\in P^*}\delta(e,P^*)l(e) - \sum_{e\in P'}\delta(e,P')l(e)\right)\\ >&0\end{aligned}$$

which contradicts the assumption that f^* is a minimum-cost flow.

Case (3): $\Delta(P') = \Delta(P^*), \Gamma(P') = \Gamma(P^*)$, and $l(P') < l(P^*)$. Then

$$c(f^*) - c(f') = \sum_{e\in P^*}\delta(e,P^*)l(e) - \sum_{e\in P'}\delta(e,P')l(e) > 0$$

which contradicts the assumption that f^* is a minimum-cost flow. This completes the proof for Class 4.

The proof for constraints in Class 5 is about the same as that for Class 4, except that $c(f^*) = M_1\cdot\Gamma(P^*) + M_2\cdot\Delta(P^*) + \sum_{e\in P^*}\delta(e,P^*)l(e)$ and $c(f') = M_1\cdot\Gamma(P') + M_2\cdot\Delta(P') + \sum_{e\in P'}\delta(e,P')l(e)$ are used. The proof of the theorem is complete. ■

Both G' and G'' have $O(|V|)$ nodes and $O(|E|)$ links, where V and E are the set of nodes and links of G, respectively. Constructing G' and G'' takes $O(|V|+|E|)$ time. Computing maximum flow on G' can be done in $O(k\cdot|E|)$ time by applying Breadth First Search on residual graphs. Thus, procedure *MinMax-Search* for computing k^L and/or k^N has time complexity $O(k\cdot\log k\cdot|E|)$. Finding a minimum-cost flow on G'' takes $O(k\cdot(|E|+|V|\log|V|))$ time. Therefore, the total time for computing minimum-cost k s–t paths under any of the 25 constraints listed in Table II is $O(k\cdot(|E|\log k + |V|\log|V|))$. Summarizing above discussions, we have the following result.

Theorem 2: Any algorithm generated by Algorithm Scheme I takes $O(k\cdot(|E|\log k + |V|\log|V|))$ time for computing minimum-cost k s–t paths in G subject to its corresponding shareability constraint of Table II.

In network applications, $k < |V|$, and the complexity of our algorithm scheme is actually $O(k|E|\log|V|)$.

V. Algorithm Scheme II

Algorithm Scheme I presented in the previous section solves the MCMPMS problem with shareability constraints satisfying a specific hierarchy, i.e., minmax constraints are of higher priority than minsum constraints. In this section, we provide a generalized algorithm scheme, Algorithm Scheme II, that can be used to solve the MCMPMS problem with all 65 possible shareability constraints, including the 25 constraints considered in the previous section.

Given a nonempty constraint $\langle X\rangle = \langle X_i, \ldots, X_2, X_1\rangle$ $(1 \le i \le 4)$, we obtain i constraints $\langle X_{j\sim 1}\rangle = \langle X_j, \ldots, X_2, X_1\rangle$ for $1 \le j \le i$. For each $\langle X_{j\sim 1}\rangle$, we obtain the *RNF with respect to* $\langle X_{j\sim 1}\rangle$ as

$$\langle X_{j\sim 1}^{\text{normal}}\rangle = \langle\langle X_{j\sim 1}^{\text{minsum}}\rangle,\ \langle X_{j\sim 1}^{\text{minmax}}\rangle\rangle.$$

Clearly, $\langle X_{j\sim 1}^{\text{normal}}\rangle$ is one of the constraints in Table II. For example, consider $\langle X\rangle = \langle \delta^*,\ \Gamma^*,\ \gamma^*,\ \Delta^*\rangle$. We have $\langle X_{1\sim 1}\rangle = \langle \Delta^*\rangle$, $\langle X_{2\sim 1}\rangle = \langle \gamma^*,\ \Delta^*\rangle$, $\langle X_{3\sim 1}\rangle = \langle \Gamma^*,\ \gamma^*,\ \Delta^*\rangle$, $\langle X_{4\sim 1}\rangle = \langle \delta^*,\ \Gamma^*,\ \gamma^*,\ \Delta^*\rangle$, $\langle X_{1\sim 1}^{\text{normal}}\rangle = \langle\langle \Delta^*\rangle,\ \langle\ \rangle\rangle = \langle \Delta^*\rangle$, $\langle X_{2\sim 1}^{\text{normal}}\rangle = \langle\langle \Delta^*\rangle,\ \langle \gamma^*\rangle\rangle = \langle \Delta^*,\ \gamma^*\rangle$, $\langle X_{3\sim 1}^{\text{normal}}\rangle = \langle\langle \Gamma^*,\ \Delta^*\rangle,\ \langle \gamma^*\rangle\rangle = \langle \Gamma^*,\ \Delta^*,\ \gamma^*\rangle$, $\langle X_{4\sim 1}^{\text{normal}}\rangle = \langle\langle \Gamma^*,\ \Delta^*\rangle,\ \langle \delta^*,\ \gamma^*\rangle\rangle == \langle \Gamma^*,\ \Delta^*,\ \delta^*,\ \gamma^*\rangle$.

Similar to Algorithm Scheme I, Algorithm Scheme II reduces the MCMPMS problem with shareability constraint $\langle X\rangle = \langle X_i, \ldots, X_2, X_1\rangle$ to finding a minimum-cost flow f^* in G'' such that $|f^*| = k$. For links in G'', the $(cost, capacity)$ parameters are assigned according to the assignment for $\langle X_{j\sim 1}^{\text{normal}}\rangle$ in Table III. The difference is that the basic component constraints in $\langle X\rangle$ are successively satisfied in multiple phases using different cost-capacity assignments in such a way that enforcing the current basic component constraint $\langle X_j\rangle$ does not violate the previously satisfied basic component constraints.

Same as before, we use $\delta(P)$, $\Delta(P)$, $\gamma(P)$, and $\Gamma(P)$ to denote the shareability values of P, a set of k s–t paths in G, with respect to constraint $\langle \delta^*\rangle$, $\langle \Delta^*\rangle$, $\langle \gamma^*\rangle$, and $\langle \Gamma^*\rangle$, respectively. For any problem instance of the MCMPMS with nonempty constraint $\langle X\rangle = \langle X_i, \ldots, X_2, X_1\rangle$ $(1 \le i \le 4)$, let s_j be the minimum shareability value of any solution P with respect to $\langle X_j\rangle$ subject to shareability value $s_0, \ldots s_{j-1}$, where s_0 is the *null* (i.e., immaterial) shareability value. Let f_j^* denote a minimum-cost k-flow whose corresponding paths satisfy minimum shareability values $s_0, \ldots, s_j$, and let f_{j-1}^* denote a k-flow satisfying minimum shareability values $s_0, \ldots, s_{j-1}$. Note that $s_j = k^L$ (resp. $s_j = k^N$) if $\langle X_j\rangle = \langle \delta^*\rangle$ (resp. $\langle X_j\rangle = \langle \gamma^*\rangle$). If the current constraint $\langle X_j\rangle$ is a minmax constraint (i.e., $\langle \delta^*\rangle$ or $\langle \gamma^*\rangle$) such that none of its preceding constraints (if any) is a minsum constraint, then the minmax constraint value s_j (k^L or k^N) is computed by binary search on the maximum flow of G'. If the current constraint $\langle X_j\rangle$ is a minmax constraint that succeeds any minsum constraint (i.e., $\langle \Delta^*\rangle$ and/or $\langle \Gamma^*\rangle$), then s_j (k^L or k^N) is computed according to the minimum shareability values $s_0, \ldots, s_{j-1}$ computed under its preceding constraints, including minsum constraint(s), by a *generalized binary search* against known minimum shareability values $\Delta(P^*)$ and/or $\Gamma(P^*)$ values. If the current constraint $\langle X_j\rangle$ is a minsum constraint, then the cost-capacity assignment of $\langle X_{j\sim 1}^{\text{normal}}\rangle$ is used to compute a minimum-cost k-flow f_j^* in G'' such that its corresponding P^* satisfies $s_0, \ldots, s_{j-1}$. More specifically, our Algorithm Scheme II is described as follows.

Algorithm Scheme II

---

begin
  construct G'' as in Algorithm Scheme I;
  find the minimum-cost flow f^* in G'' using cost-capacity assignment for the empty constraint $\langle C_0\rangle$ in **Table III**;
  if $\langle X\rangle = \langle\rangle$ **then** compute P^* from f^*, output P^* and stop;
  let s_0 have an arbitrary value;
  for $j = 1$ **to** i **do**
    case
      $\langle X_j\rangle = \langle \delta^*\rangle$ and $\langle X_j\rangle$ is $\langle minmax_1\rangle$ in $\langle X_{j\sim 1}\rangle$:
        find minimum-cost flow f_j^* in G'' using cost-capacity assignment of $\langle X_{j\sim 1}^{\text{normal}}\rangle$ in **Table III** with minimum k_1^L, while maintaining $s_0, \ldots, s_{j-1}$;
        $s_j := k_1^L$;
      $\langle X_j\rangle = \langle \delta^*\rangle$ and $\langle X_j\rangle$ is $\langle minmax_2\rangle$ in $\langle X_{j\sim 1}\rangle$:
        find minimum-cost flow f_j^* in G'' using cost-capacity assignment of $\langle X_{j\sim 1}^{\text{normal}}\rangle$ in **Table III** with minimum k_2^L, while maintaining $s_0, \ldots, s_{j-1}$;
        $s_j := k_2^L$;
      $\langle X_j\rangle = \langle \gamma^*\rangle$ and $\langle X_j\rangle$ is $\langle minmax_1\rangle$ in $\langle X_{j\sim 1}\rangle$:
        find minimum-cost flow f_j^* in G'' using cost-capacity assignment of $\langle X_{j\sim 1}^{\text{normal}}\rangle$ in **Table III** with minimum k_1^N, while maintaining $s_0, \ldots, s_{j-1}$;
        $s_j := k_1^N$;
      $\langle X_j\rangle = \langle \gamma^*\rangle$ and $\langle X_j\rangle$ is $\langle minmax_2\rangle$ in $\langle X_{j\sim 1}\rangle$:
        find minimum-cost flow f_j^* in G'' using cost-capacity assignment of $\langle X_{j\sim 1}^{\text{normal}}\rangle$ in **Table III** with minimum k_2^N, while maintaining $s_0, \ldots, s_{j-1}$;
        $s_j := k_2^N$;
      $\langle X_j\rangle$ is a minsum constraint:
        find minimum-cost flow f_j^* in G'' using cost-capacity assignment of $\langle X_{j\sim 1}^{\text{normal}}\rangle$ in **Table III** while maintaining $s_0, \ldots, s_{j-1}$;
        $x_j := \lfloor \frac{c(f^*)}{M_1} \rfloor$;
        if $\langle X_j\rangle$ is $minsum_1$ in $\langle X_{j\sim 1}\rangle$ **then** $s_j := x_j$
        else $s_j := \lfloor \frac{c(f^*) - x_j \cdot M_1}{M_2} \rfloor$
    end-case
  end-for
  compute P^* from flow f_i^*, and output P^*;
end

We will show how to compute k_1^L, k_1^N, k_2^L, and k_2^N shortly.

Lemma 1: For any graph $G = (V, E)$ with nonnegative link cost, source s, and destination t in V, if there exists an s–t path in G, then for each of the 65 shareability constraints $\langle X\rangle = \langle X_i, \ldots, X_2, X_1\rangle$, Algorithm Scheme II computes a set of k s–t paths $P^* = \{P_1^*, P_2^*, \ldots, P_k^*\}$ such that $l(P^*)$ is minimum subject to shareability constraint $\langle X\rangle$, assuming that k^L and/or k^N are known.

TABLE IV
CAPACITY CHANGE FROM CAPACITY ASSIGNMENT FOR $\langle X^{\text{normal}}_{(j-1)\sim 1}\rangle$ TO CAPACITY ASSIGNMENT FOR $\langle X^{\text{normal}}_{j\sim 1}\rangle$ WITH RESPECT TO $\langle X_j\rangle$. NC STANDS FOR "NO CHANGE". COST ASSIGNMENTS OF $\langle X^{\text{normal}}_{(j-1)\sim 1}\rangle$ AND $\langle X^{\text{normal}}_{j\sim 1}\rangle$ ARE THE SAME IF $\langle X_j\rangle$ IS A $minmax$ CONSTRAINT

$\langle X_j\rangle$	$cap^L_1(e)$	$cap^L_2(e)$	$cap^N_1(v)$	$cap^N_2(v)$
$\langle\delta^*\rangle$ and $\langle minmax_1\rangle$	NC	$k-1 \to k^L_1$	NC	NC
$\langle\delta^*\rangle$ and $\langle minmax_2\rangle$	NC	$k-1 \to k^L_2$	NC	NC
$\langle\gamma^*\rangle$ and $\langle minmax_1\rangle$	NC	NC	NC	$k-1 \to k^N_1$
$\langle minmax_2\rangle$	NC	NC	NC	$k-1 \to k^N_2$
$\langle minsum_1\rangle$ or $\langle minsum_2\rangle$	NC	NC	NC	NC

TABLE V
COST CHANGE FROM COST ASSIGNMENT FOR $\langle X^{\text{normal}}_{(j-1)\sim 1}\rangle$ TO COST ASSIGNMENT FOR $\langle X^{\text{normal}}_{j\sim 1}\rangle$ WITH RESPECT TO $\langle X_j\rangle$. NC STANDS FOR "NO CHANGE". CAPACITY ASSIGNMENTS FOR $\langle X^{\text{normal}}_{(j-1)\sim 1}\rangle$ AND $\langle X^{\text{normal}}_{j\sim 1}\rangle$ ARE THE SAME IF $\langle X_j\rangle$ IS A $minsum$ CONSTRAINT

$\langle X_j\rangle$	$cost^L_1(e)$	$cost^L_2(e)$	$cost^N_1(v)$	$cost^N_2(v)$
$\langle\Delta^*\rangle$ and $\langle minsum_1\rangle$	NC	$l(e) \to l(e)+M_1$	NC	NC
$\langle\Delta^*\rangle$ and $\langle minsum_2\rangle$	NC	$l(e) \to l(e)+M_2$	NC	NC
$\langle\Gamma^*\rangle$ and $\langle minsum_1\rangle$	NC	NC	NC	$0 \to M_1$
$\langle\Gamma^*\rangle$ and $\langle minsum_2\rangle$	NC	NC	NC	$0 \to M_2$
$\langle minmax_1\rangle$ or $\langle minmax_2\rangle$	NC	NC	NC	NC

Proof: If $\langle X\rangle = \langle\rangle$, the lemma obviously holds.

Consider $\langle X\rangle = \langle X_i, X_{i-1}, \dots, X_1\rangle \neq \langle\rangle$. We say that a k-flow f in G'' is feasible with respect to $\langle X\rangle$ if its corresponding set P of paths in G satisfy $\langle X\rangle$. We say that a k-flow f^* in G'' is optimal with respect to $\langle X\rangle$ if it is feasible with respect to $\langle X\rangle$ and $l(P^*)$ is minimum. We prove the lemma by inductively proving the claim that f^*_j is optimal with respect to $\langle X_{j\sim 1}\rangle$. Obviously, f^*_1 is optimal with respect to $\langle X_{1\sim 1}\rangle = \langle X_1\rangle$. Suppose the claim is true for $j-1$, where $j > 1$. We want to show that the claim is true for the k-flow f^*_j computed in the jth iteration of Algorithm Scheme II with the cost-capacity assignment of $\langle X^{\text{normal}}_{j\sim 1}\rangle$ respecting $s_0, \dots s_{j-1}$ is optimal with respect to $\langle X_{j\sim 1}\rangle$. We have two cases.

Case 1: $\langle X_j\rangle$ is a minmax constraint. For this case, the differences between the cost-capacity assignments corresponding to $\langle X^{\text{normal}}_{(j-1)\sim 1}\rangle$ and the cost-capacity assignments corresponding to $\langle X^{\text{normal}}_{j\sim 1}\rangle = \langle X^{\text{minsum}}_{(j-1)\sim 1}, \langle X_j\rangle, \langle X^{\text{minmax}}_{(j-1)\sim 1}\rangle\rangle$ in Table III are shown in Table IV. We have four subcases: $\langle X_j\rangle = \langle\delta^*\rangle$ and $\langle X_j\rangle$ is $\langle minmax_1\rangle$, $\langle X_j\rangle = \langle\delta^*\rangle$ and $\langle X_j\rangle$ is $\langle minmax_2\rangle$, $\langle X_j\rangle = \langle\gamma^*\rangle$ and $\langle X_j\rangle$ is $\langle minmax_1\rangle$, and $\langle X_j\rangle = \langle\gamma^*\rangle$ and $\langle X_j\rangle$ is $\langle minmax_2\rangle$.

For the subcase of $\langle X_j\rangle = \langle\delta^*\rangle$ and $\langle X_j\rangle$ being $\langle minmax_1\rangle$, cap^L_2 is reduced from $k-1$ to k^L_1, which is the minimum shareability δ value subjected to $\langle X_{(j-1)\sim 1}\rangle$. Hence, the minimum-cost flow f^*_j computed using the cost-capacity assignment for $\langle X^{\text{normal}}_j\rangle$ respects $s_0, \dots, s_{j-1}$ and, in addition, has $s_j = k^L_1$, and it is an optimal flow with respect to $\langle X_{j\sim 1}\rangle$. The proofs for other three subcases are almost the same, except that k^L_2, k^N_1 and k^N_2 are used, respectively.

Case 2: $\langle X_j\rangle$ is a minsum constraint. The differences between the cost-capacity assignments corresponding to $\langle X^{\text{normal}}_{j\sim 1}\rangle = \langle\langle X_j\rangle, \langle X^{\text{normal}}_{(j-1)\sim 1}\rangle\rangle$ and the cost-capacity assignments corresponding to $\langle X^{\text{normal}}_{(j-1)\sim 1}\rangle$ are shown in Table V. We have four subcases.

(i) $\langle X_j\rangle = \langle\Delta^*\rangle$ and $\langle X_j\rangle$ is $\langle minsum_1\rangle$ in $\langle X_{j\sim 1}\rangle$.
(ii) $\langle X_j\rangle = \langle\Gamma^*\rangle$ and $\langle X_j\rangle$ is $\langle minsum_1\rangle$ in $\langle X_{j\sim 1}\rangle$.
(iii) $\langle X_j\rangle = \langle\Gamma^*\rangle$, $\langle X_j\rangle$ is $\langle minsum_2\rangle$ in $\langle X_{j\sim 1}\rangle$ and $\langle X_l\rangle = \langle\Delta^*\rangle$ is $\langle minsum_1\rangle$ in $\langle X_{j\sim 1}\rangle$ for some $l < j$.
(iv) $\langle X_j\rangle = \langle\Delta^*\rangle$, $\langle X_j\rangle$ is $\langle minsum_2\rangle$ in $\langle X_{j\sim 1}\rangle$ and $\langle X_l\rangle = \langle\Gamma^*\rangle$ is $\langle minsum_1\rangle$ in $\langle X_{j\sim 1}\rangle$ for some $l < j$.

From the proof of Theorem 1, we know the following.
(a) $c(f^*_j) = l(P^*_j) + M_1 \cdot \Delta(P^*_j)$ for (i).
(b) $c(f^*_j) = l(P^*_j) + M_1 \cdot \Gamma(P^*_j)$ for (ii).
(c) $c(f^*_j) = l(P^*_j) + M_1 \cdot \Delta(P^*_j) + M_2 \cdot \Gamma(P^*_j)$ for (iii).
(d) $c(f^*_j) = l(P^*_j) + M_2 \cdot \Delta(P^*_j) + M_1 \cdot \Gamma(P^*_j)$ for (iv).
Since $\Delta(P^*) \le (k-1)\cdot(|V|-1)\cdot l_{\max} < k\cdot|V|\cdot l_{\max} = M_2$, $\Gamma(P^*) \le (k-1)\cdot(|V|-2) < k\cdot|V|\cdot l_{\max} = M_2$, $l(P^*) \le k\cdot(|V|-1)\cdot l_{\max} < k\cdot|V|\cdot l_{\max}$, and $M_1 = M^2_2$, we have the following.

(a′) $s_j = \Delta(P^*_j) = \lfloor \frac{c(f^*_j)}{M_1} \rfloor$ for (i).
(b′) $s_j = \Gamma(P^*_j) = \lfloor \frac{c(f^*_j)}{M_1} \rfloor$ for (ii).
(c′) $\Delta(P^*_j) = \lfloor \frac{c(f^*_j)}{M_1} \rfloor$ and $s_j = \Gamma(P^*_j) = \lfloor \frac{c(f^*_j) - \Delta(P^*)\cdot M_1}{M_2} \rfloor$ for (iii).
(d′) $\Gamma(P^*_j) = \lfloor \frac{c(f^*_j)}{M_1} \rfloor$ and $s_j = \Delta(P^*_j) = \lfloor \frac{c(f^*_j) - \Gamma(P^*)\cdot M_1}{M_2} \rfloor$ for (iv).

For (i) and (ii), only minmax shareability constraint(s) can precede $\langle X_j\rangle$ in $\langle X_{j\sim 1}\rangle$, and the computation of f^*_j preserves k^L and/or k^N. Hence, f^*_j is optimal with respect to $\langle X_{j\sim 1}\rangle$.

For (iii), compare $c(f^*_{j-1}) = l(P^*_{j-1}) + M_1 \cdot \Delta(P^*_{j-1})$ according to (a) and $c(f^*_j) = l(P^*_j) + M_1 \cdot \Delta(P^*_j) + M_2 \cdot \Gamma(P^*_j)$ according to (c). Clearly, $c(f^*_j) \le c(f^*_{j-1}) + M_2 \cdot \Gamma(P^*_{j-1}) < c(f^*_{j-1}) + M_1$, which leads to $\lfloor \frac{c(f^*_j) - c(f^*_{j-1})}{M_1} \rfloor < 1$. By (a′) and (c′), $\Delta(P^*_j) = \Delta(P^*_{j-1}) = s_{j-1}$. Furthermore, $\Gamma(P^*_j)$ is minimized while maintaining $s_0, \dots, s_{j-1}$. Hence, f^*_j is optimal with respect to $\langle X_{j\sim 1}\rangle$.

Since the induction for (iv) is similar to that of (iii), we omit it for brevity. This completes the induction and the proof of the theorem. ■

In Algorithm Scheme II and Lemma 1, minmax shareability values k^L_l and k^N_l, $l \in \{1, 2\}$, are assumed to be known. Now we show how to compute them. Assume that $\langle X_j\rangle$ is a minmax constraint. If there is no minsum constraint preceding $\langle X_j\rangle$ in $\langle X\rangle$, then the procedure *MinMax-Search* given in the previous section can be used to find the minimum $\langle X_j\rangle$ value. Complication arises if there exists a minsum constraint that precedes $\langle X_j\rangle$ in $\langle X\rangle$. We need to find k^L (resp. k^N) value if $\langle X_j\rangle = \langle\delta^*\rangle$ (resp. $\langle X_j\rangle = \langle\gamma^*\rangle$) subject to satisfying $s_1, \dots s_{j-1}$. The following two lemmas provide a basis for finding k^L_l and k^N_l, $l \in \{1, 2\}$ using a generalized binary search.

Lemma 2: Assume that $\langle X_j\rangle = \langle\delta^*\rangle$ (resp. $\langle X_j\rangle = \langle\gamma^*\rangle$) and $\langle X_m\rangle = \langle\Delta^*\rangle$ (resp. $\langle X_m\rangle = \langle\Gamma^*\rangle$), $m < j$, is the only minsum constraint preceding $\langle X_j\rangle$ in $\langle X\rangle$, let $s_1, \dots s_{j-1}$ be the shareability values computed for $\langle X \rangle$ in the first $j-1$ iterations of Algorithm Scheme II, let f' be any k-flow obtained from G'' using the cost-capacity assignment for $\langle X_{j\sim 1}^{\text{normal}} \rangle$ with an arbitrary nonnegative integer value k' between 0 and $k-2$ for k^L (resp. k^N), and let P' be the set of k paths in G that correspond to f'. The following statements hold.

(a) $\Delta(P') = \lfloor \frac{c(f')}{M_1} \rfloor$ (resp. $\Gamma(P') = \lfloor \frac{c(f')}{M_1} \rfloor$), where P' is the set of k paths in G corresponding to f'.

(b) If $\Delta(P') \neq s_m$ (resp. $\Gamma(P') \neq s_m$), then the minimum k^L (resp. k^N) with respect to $\langle X_{j-1} \rangle$ is greater than k'.

(c) If $\Delta(P') = s_m$ (resp. $\Gamma(P') = s_m$), then the minimum k^L (resp. k^N) with respect to $\langle X_{j-1} \rangle$ is smaller than or equal to k'.

Proof: (a) directly follows from the proofs of Theorem 1 and Lemma 1. By Algorithm Scheme II, s_m is $\Delta(P_m^*)$, which is the minimum Δ value computed in the mth iteration with $k^L = k-1$ (resp. $k^N = k-1$). Then, $\Delta(P') \geq s_m$ for $k^L < k-1$ (resp. $k^N < k-1$), which implies (b) and (c). Note that $\Delta(P') = s_m$ (resp. $\Gamma(P') = s_m$) for the minimum k^L (resp. k^N). ■

Lemma 3: Assume that there are two minsum constraints $\langle X_{m_1} \rangle$ and $\langle X_{m_2} \rangle$ preceding $\langle X_j \rangle$ in $\langle X \rangle$ and $m_1 < m_2 < j$, let $s_1, \ldots, s_{j-1}$ be the shareability values computed for $\langle X \rangle$ in the first $j-1$ iterations of Algorithm Scheme II, and let f' be any k-flow obtained from G'' using the cost-capacity assignment for $\langle X_{j\sim 1}^{\text{normal}} \rangle$ with an arbitrary nonnegative integer value k' between 0 and $k-2$ for k^L (resp. k^N), and let P' be the k paths in G that correspond to f'. The following statements hold.

(a) If $\langle X_{m_1} \rangle = \langle \Delta^* \rangle$ and $\langle X_{m_2} \rangle = \langle \Gamma^* \rangle$ (resp. $\langle X_{m_1} \rangle = \langle \Gamma^* \rangle$ and $\langle X_{m_2} \rangle = \langle \Delta^* \rangle$), then $\Delta(P') = \lfloor \frac{c(f')}{M_1} \rfloor$ and $\Gamma(P') = \lfloor \frac{c(f') - M_1 \cdot \Delta(P')}{M_2} \rfloor$ (resp. $\Gamma(P') = \lfloor \frac{c(f')}{M_1} \rfloor$ and $\Delta(P') = \lfloor \frac{c(f') - M_1 \cdot \Gamma(P')}{M_2} \rfloor$).

(b) If $\langle X_{m_1} \rangle = \langle \Delta^* \rangle$ and $\langle X_{m_2} \rangle = \langle \Gamma^* \rangle$ (resp. $\langle X_{m_1} \rangle = \langle \Gamma^* \rangle$ and $\langle X_{m_2} \rangle = \langle \Delta^* \rangle$), and $\Delta(P') > s_{m_1}$ or $\Gamma(P') > s_{m_2}$ (resp. $\Gamma(P') > s_{m_1}$ or $\Delta(P') > s_{m_2}$), then the minimum k^L (resp. k^N) with respect to $\langle X_j \rangle$ is greater than k'.

(c) If $\langle X_{m_1} \rangle = \langle \Delta^* \rangle$ and $\langle X_{m_2} \rangle = \langle \Gamma^* \rangle$ (resp. $\langle X_{m_1} \rangle = \langle \Gamma^* \rangle$ and $\langle X_{m_2} \rangle = \langle \Delta^* \rangle$), and $\Delta(P') = s_{m_1}$ and $\Gamma(P') = s_{m_2}$ (resp. $\Gamma(P') = s_{m_1}$ and $\Delta(P') = s_{m_2}$), then the minimum k^L (resp. k^N) with respect to $\langle X_j \rangle$ is smaller than or equal to k'.

Proof: (a) is implied by the proofs of Theorem 1 and Lemma 1. For $1 < m < j$, s_m is obtained by satisfying s_n, $n < m$. Since P' satisfies $\langle X_{(j-1)\sim 1} \rangle$, $\Delta(P') \geq s_{m_1}$ and $\Gamma(P') \geq s_{m_2}$ (resp. $\Delta(P') \geq s_{m_1}$ and $\Gamma(P') \geq s_{m_2}$); otherwise we reach a contradiction. This implies (b) and (c). ■

Remark: Lemmas 2 and 3 imply that if a minmax constraint succeeds one or two minsum constraints, minimum k^L or k^N value can always be found while maintaining previously computed cost and shareability values.

We now describe a specific algorithm, which is generated from Algorithm Scheme II, for finding minimum-cost k s–t paths with shareability constraint $\langle X \rangle = \langle \delta^*, \Gamma^*, \gamma^*, \Delta^* \rangle$.

- Iteration $j = 1$: Find minimum-cost k-flow f_1^* in G'' under constraint $\langle \Delta^* \rangle$ using cost-capacity assignments $[(l(e), 1), (l(e) + M_1, k-1)]$, $[(0,1), (0, k-1)]$ for RNF $\langle X_{1\sim 1}^{\text{normal}} \rangle = \langle C_5 \rangle = \langle \Delta^* \rangle$ in Table III. Clearly, $s_1 = \Delta(P_1^*) = \lfloor \frac{c(f_1^*)}{M_1} \rfloor$.
- Iteration $j = 2$: Compute k_1^N under constraint $\langle \Delta^* \rangle$ using generalized binary search as follows:

$low := 0$; $high := k-1$; $k_1^N := \lfloor \frac{k-1}{2} \rfloor$;
while $low \neq high$ **do**
  apply MCNF algorithm to find minimum-cost k-flow f_2^* in G'' using cost-capacity assignments $[(l(e), 1), (l(e) + M_1, k-1)]$, $[(0,1), (0, k_1^N)]$ for **RNF** $\langle X_{2\sim 1}^{\text{normal}} \rangle = \langle C_7 \rangle = \langle \Delta^*, \gamma^* \rangle$;
  if $\lfloor \frac{c(f_2^*)}{M_1} \rfloor \neq \Delta(P_1^*)$
  then $low := k_1^N$ and $k_1^N := low + \lfloor (high - low)/2 \rfloor$
  else $high := k_1^N$ and $k_1^N := low + \lfloor (high - low)/2 \rfloor$
end-while
return k_1^N as $\gamma(P_2^*)$;

  Clearly, $s_2 = \gamma(P_2^*)$ and $\Delta(P_2^*) = \lfloor \frac{c(f_2^*)}{M_1} \rfloor = \lfloor \frac{c(f_1^*)}{M_1} \rfloor = \Delta(P_1^*) = s_1$.
- Iteration $j = 3$: Find minimum-cost k-flow f_3^* in G'' under constraint $\langle \Gamma^*, \gamma^*, \Delta^* \rangle$ using cost-capacity assignments $[(l(e), 1), (l(e) + M_1, k-1)]$, $[(0,1), (M_2, k_1^N)]$ for RNF $\langle X_{3\sim 1}^{\text{normal}} \rangle = \langle C_{17} \rangle = \langle \Gamma^*, \Delta^*, \gamma^* \rangle$ of Table III. Clearly, $s_3 = \Gamma(P_3^*) = \lfloor \frac{c(f_3^*) - M_1 \cdot \Delta(P_2^*)}{M_2} \rfloor$, $\gamma(P_3^*) = \gamma(P_2^*) = s_2$, and $\Delta(P_3^*) = \lfloor \frac{c(f_3^*)}{M_1} \rfloor = \lfloor \frac{c(f_2^*)}{M_1} \rfloor = \Delta(P_2^*) = \Delta(P_1^*) = s_1$.
- Iteration $j = 4$: Find minimum-cost k-flow f_4^* in G'' under constraint $\langle \delta^*, \Gamma^*, \gamma^*, \Delta^* \rangle$ using cost-capacity assignments $[(l(e), 1), (M_1 + l(e), k_2^L)]$, $[(0,1), (M_2, k_1^N)]$ for RNF $\langle X_{4\sim 1}^{\text{normal}} \rangle = \langle C_{19} \rangle = \langle \Gamma^*, \Delta^*, \delta^*, \gamma^* \rangle$ and generalized binary search as follows:

$low := 0$; $high := k-1$; $k_2^L := \lfloor \frac{k-1}{2} \rfloor$;
while $low \neq high$ **do**
  apply MCNF algorithm to find minimum-cost k-flow f_4^* in G'' using cost-capacity assignments $[(l(e), 1), (M_1 + l(e), k_2^L)]$, $[(0,1), (M_2, k_1^N)]$ for $\langle C_{19} \rangle$;
  $\Delta' := \lfloor \frac{c(f_4^*)}{M_1} \rfloor$; $\Gamma' := \lfloor \frac{c(f_4^*) - M_1 \cdot \Delta'}{M_2} \rfloor$;
  if $\Delta' \neq \Delta(P_3^*)$ or $\Gamma' \neq \Gamma(P_3^*)$
  then $low := k_2^L$ and $k_2^L := low + \lfloor (high - low)/2 \rfloor$
  else $high := k_2^L$ and $k_2^L := low + \lfloor (high - low)/2 \rfloor$
end-while
return k_2^L as $\delta(P_4^*)$;

  Clearly, $s_4 = \delta(P_4^*)$, $\Gamma(P_4^*) = \lfloor \frac{c(f_4^*) - M_1 \cdot \Delta(P_4^*)}{M_2} \rfloor = \lfloor \frac{c(f_3^*) - M_1 \cdot \Delta(P_3^*)}{M_2} \rfloor = s_3$, $\gamma(P_4^*) = \gamma(P_3^*) = \gamma(P_2^*) = s_2$, and $\Delta(P_4^*) = \lfloor \frac{c(f_4^*)}{M_1} \rfloor = \lfloor \frac{c(f_3^*)}{M_1} \rfloor = \lfloor \frac{c(f_2^*)}{M_1} \rfloor = \lfloor \frac{c(f_1^*)}{M_1} \rfloor = s_1$.
- Finally, the set P_4^* of k s–t paths is computed according to flow f_4^* as the final solution of minimum-cost paths satisfying $\langle \delta^*, \Gamma^*, \gamma^*, \Delta^* \rangle$.

Theorem 3: For any graph $G = (V, E)$ with nonnegative link cost, source s, and destination t in V, if there exists an s–t path in G, then for any of the 65 shareability constraints the algorithm generated from Algorithm Scheme II computes a set of k s–t paths $P^* = \{P_1^*, P_2^*, \ldots, P_k^*\}$ such that $l(P^*)$ is minimum subject to the shareability constraint in $O(k \cdot (|E| \log k + |V| \log |V|))$ time.

Proof: By Lemmas 1 to 3, we conclude such an algorithm is correct. Constructing G' and G'' takes

$O(|V| + |E|)$ time. Finding a minimum-cost k-flow in G'' takes $O(k \cdot (|E| + |V| \log |V|))$ time. Generalized binary search for k_l^L or k_l^N, $l \in \{1,2\}$, invokes $O(\log k)$ executions of the minimum-cost k-flow algorithm. Hence, the total time for any algorithm generated from Algorithm Scheme II is $O(k \cdot \log k \cdot (|E| + |V| \log |V|))$ ■

In network applications, $k < |V|$, and the complexity of our algorithm scheme is actually $O(|E| k \log k \log |V|)$.

VI. Generalizations

Generalizations of the results of the previous sections are possible. In this section, we discuss two such generalizations.

A. *Nonuniform Maximum Allowable Shareabilities*

So far, we have assumed that all links and nodes (except s and t) have the same maximum allowable shareability $k-1$. For many applications, we may want to assign different maximum allowable shareabilities to individual links and/or nodes. For example, in WDM optical networks, the number of available wavelengths on links may be different, and consequently we may assume that the maximum allowable shareability of a link to be the number of its available wavelengths less 1. In reliable network communication, if we know a link (resp. node) has smaller (larger) failure probability, we may assign a larger (smaller) maximum allowable shareability value to the link (resp. node), thereby restricting the number of paths that use the link (resp. node).

We generalize the minimum-cost k path problem with shareability constraints by adding two more constraints:

$$\delta_{\max} : E \longrightarrow \{0, 1, \ldots, k-1\}$$

and

$$\gamma_{\max} : V \longrightarrow \{0, 1, \ldots, k-1\}$$

where E and V are the link set and node set of a given network $G = (V, E)$. Each link e (resp. node $v \neq s, t$) is allowed to be shared by at most $\delta_{\max}(e) + 1$ (resp. $\gamma_{\max}(v) + 1$) paths, where $\delta_{\max}(e)$ (resp. $\gamma_{\max}(v)$) is the *maximum allowable link* (resp. *node*) *shareability* of e (resp. v). Clearly, the problems (with uniform maximum allowable shareability constraints) considered in the previous sections are special cases of the problem (with nonuniform maximum allowable shareability constraints) we are discussing.

Our Algorithm Schemes I and II can be easily modified to solve the problem with extra nonuniform maximum allowable shareability constraints.

Let us consider Algorithm Scheme I. It is possible that k paths from s to t in G satisfying maximum allowable shareability constraints do not exist even t is reachable from s. Thus, before trying to find k desired paths, it is necessary to check whether or not there exist k s–t paths in G by running a maximum flow algorithm on G' with the following capacity assignment: Assign capacity $\delta_{\max}(e) + 1$ to link (u', v) in G' corresponding to link (u, v) in G, and assign capacity $\gamma_{\max}(v) + 1$ to link (v, v') in G' corresponding to node v in G. If a k-flow exists in G', then there is a feasible solution. Otherwise, k s–t paths in G do not exist.

For finding the minimum k^L (resp. k^N), binary search of procedure *MinMax-Search-Modified*, a modified version of *MinMax-Search*, can be used to find minimum k^L (or k^N) such that a k-flow f exists without violating the capacity limit $\delta_{\max}(e)$ (resp. $\gamma_{\max}(v)$) of secondary $u-v$ (resp. v) link in G''. For the aim of simplicity, we only list the modifications that are needed in *MinMax-Search-Modified*.

- Before the **while** loop, *MinMax-Search* initializes all links in $E' - E^*$ with capacity k'. In *MinMax-Search-Modified*, such an initialization should be changed to $\min\{k', \gamma_{\max}(v)+1\}$ if $\text{flag} = 0$, or $\min\{k', \delta_{\max}(e)+1\}$ if $\text{flag} = 1$.
- In the **while** loop, *MinMax-Search* assigns each link in E^* the capacity $\lim$. In *MinMax-Search-Modified*, the capacity assignment should be changed to $\min\{\lim, \delta_{\max}(e) + 1\}$ if $\text{flag} = 0$, or $\min\{\lim, \gamma_{\max}(v) + 1\}$ if $\text{flag} = 1$.

Then, for the 25 constraints given in Table II, k^L and k^N are computed using procedure *MinMax-Compute* by calling *MinMax-Search-Modified* instead of *MinMax-Search*.

With respect to Table III, the $capacity$ of a $(cost, capacity)$ pair for a link in G'' for finding optimal k paths under a specific composite constraint is modified as follows: Value $k-1$ in $cap_2^L(e)$ (resp. $cap_2^N(v)$) is replaced by $\delta_{\max}(e)$ (resp. $\gamma_{\max}(v)$), and value k_l^L (resp. k_l^N), $l \in \{1,2\}$, in $cap_2^L(e)$ (resp. $cap_2^N(v)$) is replaced by $\min\{\delta_{\max}(e), k_l^L\}$ (resp. $\min\{\gamma_{\max}(v), k_l^N\}$). It is easy to see that minimum-cost k s–t paths that satisfy any (composite) constraint of Table II and nonuniform maximum allowable individual link/node shareabilities can be computed by finding a minimum-cost k-flow f^* in G''. Hence, using this generalized Algorithm Scheme I, the corresponding 25 versions of the MCMPMS problem with nonuniform maximum allowable shareabilities can be solved in the same amount of time as solving their counterparts with uniform maximum allowable shareabilities.

Algorithm Scheme II can also be generalized to cope with nonuniform maximum allowable shareability. The key to this generalization is to compute a minimum-cost k-flow f_j^* in G'' with constraint $\langle X_{j\sim 1}\rangle$ using the above modified cost-capacity assignment that satisfies $s_1, \ldots, s_{j-1}$. If $\langle X_j\rangle$ is a minsum constraint, this computation is trivial. However, if $\langle X_j\rangle$ is a minmax constraint, a generalized binary search is needed. Procedure *MinMax-Search-Modified* provides sufficient details for deriving a generalized binary search procedure for finding minimum minmax shareability values; for brevity, we omit further discussions. In summary, using this generalized Algorithm Scheme II, all 65 versions of the MCMPMS problem with nonuniform maximum allowable shareabilities can be solved in the same amount of time as solving their counterparts with uniform maximum allowable shareabilities.

B. *Minimum-Cost One-to-Many and Many-to-One Paths Subject to Minimum Shareability*

Consider the following one-to-many communication problem. Given a weighted directed graph $G = (V, E)$, a node s in G, and a subset $T = \{t_1, t_2, \ldots, t_k\}$ of $V - \{s\}$. Our objective is to find minimum-cost k paths $P = \{P_{t_1}, P_{t_2}, \ldots, P_{t_k}\}$, where P_{t_i} is a path from s to t_i, subject to minimum shareability constraint $\langle C\rangle$. Finding many-to-one paths can be carried out

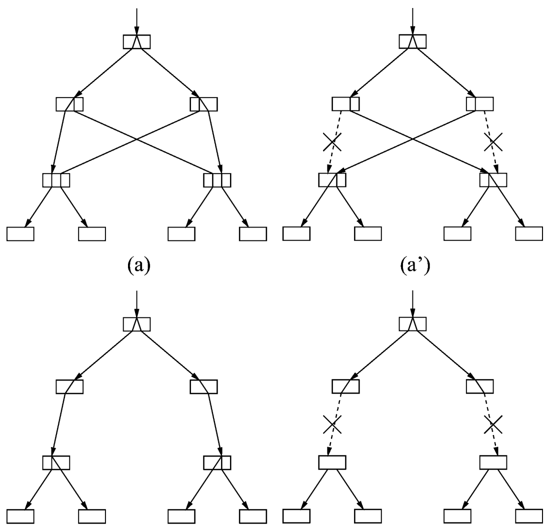


Fig. 5. Multiple paths for multicasting. Multiple paths of (a) have minimum vulnerability as shown in (a′). Multiple paths of (b), which form a tree, have larger vulnerability as shown in (b′).

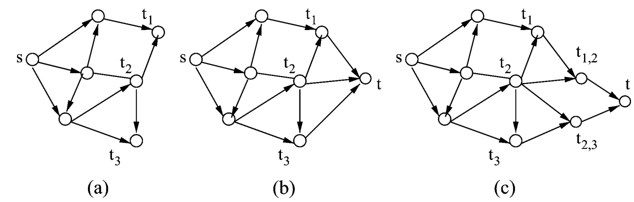


Fig. 6. Transformation used to solve related problems. (a) A given graph G. (b) Graph G^* for finding optimal paths from s to destinations in $T = \{t_1, t_2, t_3\}$. (c) Graph G^* for finding optimal paths from s to destination-pairs (t_1, t_2) and (t_2, t_3).

by reversing the directions of all links in G, and then finding k paths to nodes in T.

This problem has applications in reliable multicasting, with s being the source node and T being the set of destination nodes. Node s can multicast information to all nodes in T using paths of P without each destination receiving the same information (such as a packet) twice. With such set P of paths, the number of affected nodes in T can be expected minimum when a link/node fails. Consider the example shown in Fig. 5. It is easy to see that the structure of Fig. 5(a) and the structure of Fig. 5(b) have the same minmax link and node shareability values, but the minsum link and node shareability values of Fig. 5(a) are smaller than that of Fig. 5(b). In case of the two indicated link failures, the structure of Fig. 5(a) guarantees that all destinations to be still connected to the source by reconfiguring intermediate switches (routers) as shown in Fig. 5(a′), while all destinations are disconnected from the source in the structure of Fig. 5(b), as shown in Fig. 5(b′).

This problem also has applications in reliable client–server communication. The source node can be considered as a server, and all destinations are clients. In this case, one-to-k paths subject to minimum shareability can be used to ensure the least number of clients are affected in case of link/node failure or hotspot congestion. Another possibility is that the source is a client that wishes to receive reliable service from one of k geographically distributed server nodes that provide the same service. When one path is not usable, due to either link/node failure or hotspot congestion, the client can quickly switch to another path. If the servers provide different services, such one-to-k paths can be used to maximize the service functionality in case of link/node failure.

We can reduce this problem of finding one-to-k paths to finding k s–t paths as follows: We construct a graph $G^* = (V^*, E^*)$ from G by introducing a new node t, and introducing a link from each node t_i to the new node t (see Fig. 6(b) for an example). We assign cost 0 to the link from t_i to t, and a maximum allowable link shareability 0 to its secondary link in G''. Then, we apply our algorithm schemes to find minimum-cost k s–t paths from s to t in G^* satisfying $\langle C \rangle$.

Another related problem is finding minimum-cost protection of dual homing architecture considered in [17], [18], and [24]. Given a weighted graph $G = (V, E)$ with $|V| = n$, $|E| = m$, a source node s, a set T of k pairs (t_i, t_j) with $t_i, t_j \in V - \{s\}$, and shareability constraint $\langle C \rangle$, find two paths from s to every pair (t_i, t_j) of nodes in T such that $\langle C \rangle$ is satisfied and the total cost of the paths is minimum. This problem can be reduced to finding k s–t paths as follows: We construct a graph $G^* = (V^*, E^*)$ from G by introducing a new node $t_{i,j}$ for each pair (t_i, t_j), two links from nodes t_i and t_j to the new node $t_{i,j}$ with cost 0 and maximum allowable link shareability 0, a new node t, and a link from each $t_{i,j}$ to t with cost 0 and maximum allowable link shareability 1 (see Fig. 6(c) for an example). Then, all we need to do is to find minimum-cost $2k$ paths from s to t satisfying $\langle C \rangle$ in G^*. Using the generalized Algorithm Scheme I (resp. Algorithm Scheme II), all 25 versions (resp. 65) of this problem of finding minimum-cost one-to-$2k$ paths with uniform or nonuniform maximum allowable shareabilities can be solved in the same amount of time as solving their counterparts of finding k s–t paths.

VII. Conclusion

We characterized the degree of link sharing and node sharing by the notion of link shareability and node shareability. We defined a collection of minimum-cost multiple paths problems with prioritized minimization objectives. All shareability minimization objectives are treated as constraints for finding minimum-cost paths. We identified 65 mutually inequivalent shareability constraints based on selections and permutations of minmax link and node shareabilities and minsum link and node shareabilities. In addition, we also considered uniform allowable link and node shareability constraint, and nonuniform allowable link and node shareability constraints.

We presented two algorithm schemes, Scheme I and Scheme II, each of which is used to generate a set of efficient polynomial-time algorithms according to the constraints. These algorithms can be used to find link-disjoint and node-disjoint paths if they exist by checking the minmax link shareability and minmax node shareability in the resulting solution. Our results constitute a general framework for finding multiple paths with minimum link and/or node sharing.

An outstanding open problem is to design algorithms for finding k paths between k source–destination pairs, one path per pair, so that the cost of these paths are minimum subject to shareability constraints. This problem is NP-complete, since the known NP-complete 2DP problem [6] can be reduced to this multiple-source, multiple-destination problem. We refer to [4] for a latest review on this topic. It is also important to generalize the MCMPMS modeling framework to such multicommodity problems.

ACKNOWLEDGMENT

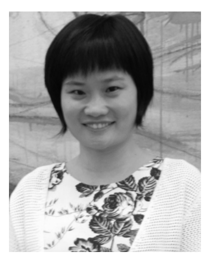
The authors would like to thank anonymous reviewers for their valuable comments that made the improvement of this paper possible.

REFERENCES

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows*. Englewood Cliffs, NJ: Prentice-Hall, 1993.
[2] B. Bank and M. E. Zarki, "Dynamic multi-path routing and how it compares with other dynamic routing algorithms for high speed wide area networks," in *Proc. ACM SIGCOMM*, 1992, pp. 53–64.
[3] D. Castanon, "Efficient algorithms for finding the K best paths through a trellis," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 26, no. 2, pp. 405–410, Mar. 1990.
[4] C. Chekuri and S. Khanna, "Edge-disjoint paths revisited," *ACM Trans. Algor.*, vol. 3, no. 4, 2007, Article no. 46.
[5] L. R. Ford and D. R. Fulkerson, *Flows in Networks*. Princeton: Princeton Univ. Press, 1962.
[6] S. Fortune, J. Hopcroft, and J. Wyllie, "The directed subgraph homeomorphism problem," *Theor. Comput. Sci.*, vol. 10, pp. 111–121, 1980.
[7] P. Georgatsos and D. Griffin, "A management system for load balancing through adaptive routing in multiservice ATM networks," in *Proc. IEEE INFOCOM*, 1996, pp. 863–870.
[8] K. Ishida, Y. Kakuda, and T. Kikuno, "A routing protocol for finding two node-disjoint paths in computer networks," in *Proc. Int. Conf. Netw. Protocols*, 1992, pp. 340–347.
[9] R. Krishnan and J. Silvester, "Choice of allocation granularity in multipath source routing schemes," in *Proc. IEEE INFOCOM*, 1993, vol. 1, pp. 322–329.
[10] S.-W. Lee and C.-S. Wu, "A k-best paths algorithm for highly reliable communication networks," *IEICE Trans. Commun.*, vol. E82-B.4, pp. 586–590, 1999.
[11] C. L. Li, S. T. McCormick, and D. Simchi-Levi, "The complexity of finding two disjoint paths with min-max objective function," *Discrete Appl. Math.*, vol. 26, no. 1, pp. 105–115, 1990.
[12] C. L. Li, S. T. McCormick, and D. Simchi-Levi, "Finding disjoint paths with different path-costs: Complexity and algorithms," *Networks*, vol. 22, pp. 653–667, 1992.
[13] S. D. Nikolopoulos, A. Pitsillides, and D. Tipper, "Addressing network survivability issues by finding the K-best paths through a trellis graph," in *Proc. IEEE INFOCOM*, 1997, pp. 370–377.
[14] J. W. Suurballe, "Disjoint paths in a network," *Networks*, vol. 4, pp. 125–145, 1974.
[15] J. W. Suurballe and R. E. Tarjan, "A quick method for finding shortest pairs of disjoint paths," *Networks*, vol. 14, pp. 325–336, 1984.
[16] H. Suzuki and F. A. Tobagi, "Fast bandwidth reservation scheme with multi-link and multi-path routing in ATM networks," in *Proc. IEEE INFOCOM*, 1992, vol. 3, pp. 2233–2240.
[17] J. Wang, M. Yang, X. Qi, and R. Cook, "Dual-homing multicast protection," in *Proc. IEEE Globecom*, 2004, pp. 1123–1127.
[18] J. Wang, M. Yang, B. Yang, and S. Q. Zheng, "Dual-homing based scalable partial multicast protection," *IEEE Trans. Comput.*, vol. 55, no. 9, pp. 1130–1141, Sep. 2006.
[19] D. Xu, Y. Chen, Y. Xiong, C. Qiao, and X. He, "On the complexity and algorithms for finding the shortest path with a disjoint counterpart," *IEEE/ACM Trans. Netw.*, vol. 14, no. 1, pp. 147–158, Feb. 2006.
[20] B. Yang, S. Q. Zheng, and S. Katukam, "Finding two disjoint paths in a network with min-min objective function," in *Proc. 15th Int. Conf. Parallel Distrib. Comput. Syst.*, 2003, pp. 75–80.
[21] B. Yang, S. Q. Zheng, and E. Lu, "Finding two disjoint paths in a network with normalized α^+-min-sum objective function," in *Proc. 16th Int. Symp. Algor. Comput.*, 2005, Lecture Notes in Computer Science, 3827, pp. 954–063.
[22] B. Yang, S. Q. Zheng, and E. Lu, "Finding two disjoint paths in a network with normalized α^--min-sum objective function," in *Proc. 17th Int. Conf. Parallel Distrib. Comput. Syst.*, 2005, pp. 342–348.
[23] B. Yang, S. Q. Zheng, and E. Lu, "Finding two disjoint paths in a network with MinSum-MinMin objective function," in *Proc. Int. Conf. Found. Comput. Sci.*, 2007, pp. 355–361.
[24] M. Yang, J. Wang, X. Qi, and Y. Jiang, "On finding the best partial multicast protection tree under dual-homing architecture," in *Proc. IEEE HPSR*, 2005, pp. 128–132.
[25] S. Q. Zheng, J. Wang, B. Yang, and M. Yang, "Minimum cost multiple paths subject to minimum link and node sharing in a network: The inequivalence of 65 cases," Tech. Rep. UTDCS-04-09, 2009.

S. Q. Zheng (SM'98) received the Ph.D. degree from the University of California, Santa Barbara, in 1987.

He is currently a Professor of computer science, computer engineering, and telecommunications engineering with the University of Texas at Dallas, Richardson. His research interests include algorithms, computer architectures, networks, parallel and distributed processing, telecommunications, and VLSI design. He has published extensively in these areas.



Jianping Wang (M'03) received the B.Sc. and the M.Sc. degrees in computer science from Nankai University, Tianjin, China, in 1996 and 1999, respectively, and the Ph.D. degree in computer science from the University of Texas at Dallas, Richardson, in 2003.

She is currently an Assistant Professor with the Department of Computer Science, City University of Hong Kong. Her research interests include optical networks and wireless networks.

Bing Yang received the B.S. degree in computer science from Nankai University, Tianjin, China, in 1993; the M.S. degree in mathematics from Southern Illinois University, Edwardsville, in 1996; the M.S. degree in computer science from Texas A&M University, College Station, in 1998; and the Ph.D. degree from the University of Texas at Dallas, Richardson, in 2005.

Currently, he is a Software Engineer with the Optical Network Group at Cisco Systems, Richardson, TX. His primary research interest is on networking algorithms.

Mei Yang (M'03) received the Ph.D. degree in computer science from the University of Texas at Dallas, Richardson, in August 2003.

She is currently an Assistant Professor with the Department of Electrical and Computer Engineering, University of Nevada, Las Vegas (UNLV). Her research interests include computer networks, wireless sensor networks, computer architecture, and embedded systems.