

Multipath Routing Algorithms for Congestion Minimization

Ron Banner and Ariel Orda
Department of Electrical Engineering
Technion – Israel Institute of Technology
Haifa 32000, Israel
{ banner@tx, ariel@ee .technion.ac.il

Abstract

Unlike traditional routing schemes that route all traffic along a single path, multipath routing strategies split the traffic among several paths in order to ease congestion. It has been widely recognized that multipath routing can be fundamentally more efficient than the traditional approach of routing along single paths. Yet, in contrast to the single-path routing approach, most studies in the context of multipath routing focused on heuristic methods. We demonstrate the significant advantage of optimal solutions. Hence, we investigate multipath routing adopting a rigorous (theoretical) approach. We formalize problems that incorporate two major requirements of multipath routing. Then, we establish the intractability of these problems in terms of computational complexity. Accordingly, we establish efficient solutions with proven performance guarantees.

Keywords: Routing, Congestion, Algorithms, Optimization, Combinatorics.

1. Introduction

Current routing schemes typically focus on discovering a single "optimal" path for routing, according to some desired metric. Accordingly, traffic is always routed over a single path, which often results in substantial waste of network resources. *Multipath Routing* is an alternative approach that distributes the traffic among several "good" paths instead of routing all traffic along a single "best" path.

Multipath routing can be fundamentally more efficient than the currently used single-path routing protocols. It can significantly reduce congestion in "hot spots", by deviating traffic to unused network resources, thus improving network utilization and providing load balancing [1]. Moreover, congested links usually result in poor performance and high variance. For such circumstances, multipath routing can offer steady and smooth data streams [2].

Previous studies and proposals on multipath routing have focused on *heuristic methods*. In [14], a multipath routing scheme, termed Equal Cost MultiPath (ECMP), has been proposed for balancing the load along multiple shortest paths using a simple round-robin distribution. By limiting itself to shortest paths, ECMP considerably reduces the load-balancing capabilities of multipath routing; moreover the *equal* partition of flows along the (shortest) paths (resulting from the round robin distribution) further limits the ability to decrease congestion through load balancing. OSPF-OMP [6] allows splitting traffic among paths *unevenly*; however, the traffic distribution mechanism is based on a heuristic scheme that often results in an inefficient flow distribution. Both [15] and [31] considered multipath routing as an optimization problem with an objective function that minimizes the congestion of the most utilized link in the network; however, they focused on heuristics and did not consider the quality of the selected paths. In [7], a scheme is presented to proportionally split traffic among several "widest" paths that are disjoint with respect to the *bottleneck links*. However, here too, the scheme is heuristic and evaluated by way of simulations.

Simulation results clearly indicate that multipath solutions obtained by *optimal* congestion reduction schemes are *fundamentally* more efficient than the solutions obtained by heuristics. For example, in Section 5, we show that if the traffic distribution mechanism in the ECMP scheme had been optimal, the network congestion would have decreased by more than *three times*; moreover, if paths other than shortest had been allowed, the optimal partition would have decreased the network congestion by more than *ten times*. Hence, the full potential of multipath routing is far from having been exploited.

Accordingly, in this study we investigate multipath routing adopting a rigorous approach, and formulate it as an optimization problem of minimizing network congestion. Under this framework, we consider two fundamental requirements. First, each of the chosen paths should usually be of satisfactory "quality". Indeed, while better load balancing is achieved by allowing the employment of paths other than shortest, paths that are substantially inferior (i.e., "longer") may be prohibited. Therefore, we consider the problem of congestion minimization through multipath routing subject to a restriction on the "quality" (i.e., length) of the chosen paths.

Another practical restriction is on the *number of routing paths per destination*, which is due to several reasons [7]: first, establishing, maintaining and tearing down paths pose considerable overhead; second, the complexity of a scheme that distributes traffic among multiple paths considerably increases with the number of paths; third, often there is a limit on the number of explicitly routing paths (such as label-switched paths in MPLS [45]) that can be set up between a pair of nodes. Therefore, in practice, it is desirable to use as few paths as possible while at the same time minimize the network congestion.

Our Results

Consider first the problem of minimizing the congestion under the requirement to route traffic along paths of "satisfactory" quality. We first show that the considered problem is NP-hard, yet admits a pseudo-polynomial solution. Accordingly, we design two algorithms. The first is an optimal algorithm with a pseudo-polynomial running time, and the second approximates the optimal solution to any desired degree of precision at the (proportional) cost of increasing its running time (i.e., an ε -optimal approximation scheme). In addition, we show that these algorithms can be extended to offer solutions to reliability-related problems.

Consider now the requirement of limiting the number of paths per destination. We show that minimizing the congestion under this restriction is NP-hard as well. Accordingly, we establish a computationally efficient 2-approximation scheme¹. Then, we generalize the 2-approximation scheme into a bicriteria result and establish a $(1+1/r)$ -approximation scheme that, for any given $r \geq 1$, violates the constraint on the number of routing paths by a factor of at most r . Finally, we broaden the scope of this problem and establish an efficient approximation scheme for the dual problem, which restricts the level of congestion while minimizing the number of paths per destination.

Organization

The rest of this document is organized as follows. In section 2, we introduce some terminology and definitions, and formulate the main problems considered in this study. In section 3, we consider multipath routing schemes with additive QoS requirements and end-to-end reliability constraints. In section 4, we consider the problem of minimizing congestion subject to a restriction on the number of paths per destination. In section 5, we provide some simulation results. Finally, in section 6, we propose several directions for future work.

2. Model and Problems Formulation

This section formulates the general model and main problems that are addressed in this study. We begin with a definition of a general communication network.

¹ i.e., an algorithm that provides a solution that, in terms of congestion, is within a factor of at most 2 away from the optimum.

A *network* is represented by a directed graph $G(V, E)$, where V is the set of nodes and E is the set of links. Let $N = |V|$ and $M = |E|$. A *path* is a finite sequence of nodes $p = (v_0, v_1, \dots, v_h)$, such that, for $0 \leq n \leq h-1$, $(v_n, v_{n+1}) \in E$. A path is *simple* if all its nodes are distinct. A *cycle* is a path $p = (v_0, v_1, \dots, v_h)$ together with the link $(v_h, v_0) \in E$ i.e., $(v_0, v_1, \dots, v_h, v_0)$. Denote the set of all cycles in a network G by $T(G)$.

A *commodity* is a pair of nodes $(i, j) \in V \times V$ that is assigned with a non-negative demand $\gamma^{(i,j)}$. Let β be the set of all commodities with *positive* demand $\beta = \{(i, j) \mid (i, j) \in V \times V, \gamma^{(i,j)} > 0\}$. Given a commodity $(i, j) \in V \times V$, we say that node i is the *source node* of the given commodity and node j is the *target node*. If $|\beta| \leq 1$, we say that the network has a *single commodity* flow demand. Otherwise, we say that the network has a *multi-commodity* flow demand.

The set $P^{(i,j)}$ is the collection of all directed paths from the source i to the destination j in the network. In addition, let $P \triangleq \bigcup_{(i,j) \in V \times V} P^{(i,j)}$ and let $P_{simple}^{(i,j)} \subseteq P^{(i,j)}$ represents the set of *simple paths* from i to j in the network. Finally, for each path $p \in P^{(s,t)}$ and link $e \in E$, $\Delta_e(p)$ counts the number of occurrences of the link e in the path p . For example, given a non-simple path $p = (v_0, v_1, v_2, v_3, v_1, v_2, v_4)$ and a link $e = (v_1, v_2)$, we have $\Delta_e(p) = 2$.

Each link $e \in E$ is assigned a *weight* $w_e \in \mathbb{Z}^+$ and a *capacity* $c_e \in \mathbb{Z}^+$. We assume that the link weights w_e constitute an additive metric. We consider a *link state* routing environment, where each source node has an image of the entire network.

Definition 2.1 Given a (non-empty) path p , the weight $W(p)$ of p is defined as the sum of weights of its links, namely, $W(p) = \sum_{e \in p} w_e$.

Definition 2.2 Given a (non-empty) path p , the capacity $C(p)$ of p is defined as the capacity of its bottleneck link, namely, $C(p) = \text{Min}_{e \in E} \{c_e\}$.

Definition 2.3 Let $G(V, E)$ be a network. A *path flow* is a real-valued function $f : P \rightarrow \mathbb{R}^+ \cup \{0\}$ that satisfies the following two properties:

Capacity constraints: For each $e \in E$, $\sum_{p \in P} \Delta_e(p) \cdot f(p) \leq c_e$.

Flow demand: For each commodity $(i, j) \in V \times V$, $\sum_{p \in P^{(i,j)}} f(p) = \gamma^{(i,j)}$.

Definition 2.4 Given is a path flow $f : P \rightarrow \mathbb{R}^+ \cup \{0\}$ over a network $G(V, E)$. A *link flow of a commodity* $(i, j) \in V \times V$ is a real-valued function

$f : E \times V \times V \rightarrow \mathbb{R}^+ \cup \{0\}$ that satisfies, for each link $e \in E$: $f_e^{(i,j)} \triangleq \sum_{p \in P} \Delta_e(p) \cdot f(p)$.

Denote $f_e \triangleq \sum_{(i,j) \in V \times V} f_e^{(i,j)}$.

Definition 2.5 Let $G(V, E)$ be a network. A *cycle flow of a commodity* $(i, j) \in V \times V$ is a real-valued function $f : T(G) \times V \times V \rightarrow \mathbb{R}^+ \cup \{0\}$.

Note that, for a given link flow of a commodity $(i, j) \in V \times V$, the path flow $f : P^{(i,j)} \rightarrow \mathbb{R}^+ \cup \{0\}$ is not necessarily unique. In addition, the path flow representation (which is of size $O(|P|)$) may have exponential size (with respect to the network representation). However, it follows from the flow decomposition theorem [12] that any path flow assigned for a commodity $(i, j) \in V \times V$ (i.e. the collection of pairs $(p, f(p))$ for each $p \in P^{(i,j)}$) has a corresponding path flow with at most M paths and cycles with a positive flow that share the same link flow representation.

Definition 2.6 Given a network $G(V, E)$ and a link flow $\{f_e\}$, the value $\frac{f_e}{c_e}$ is the *link congestion factor*.

Definition 2.7 Given a network $G(V, E)$ and a link flow $\{f_e\}$, the *network congestion factor* is the largest link congestion factor in the network, i.e., $\max_{e \in E} \left\{ \frac{f_e}{c_e} \right\}$.

As noted in [1],[15],[41] the network congestion factor provides a good indication of congestion.

We now establish that minimizing the network congestion factor is equivalent for the single commodity case to maximizing the flow i.e., the objective function of the well known Max Flow Problem [12].

Theorem 2.1 Given a network $G(V, E)$ two nodes $\{s, t\}$ capacities $\{c_e\}$ and a demand γ . $\{f_e\}$ is a solution to the instance $\langle G(V, E), \{c_e\}, \{s, t\} \rangle$ of the Maximum Flow Problem that transfers $F \geq \gamma$ flow units from s to t iff $\left\{ f_e \cdot \frac{\gamma}{F} \right\}$ is a link flow that transfer γ flow units from s to t such that the network congestion factor is minimized.

The proof to the Theorem appears in the Appendix.

We are now ready to formulate the main two problems considered in this study. Both minimize the network congestion factor subject to different considerations. We saw that, in the single commodity case, minimizing the network congestion factor is equivalent to maximizing the total throughput. Therefore, all these problems remain equivalent for the single commodity case if we consider a different objective function,

namely that of maximizing the throughput. However, whereas maximizing the throughput is not well defined for the multi-commodity case, minimizing the network congestion factor is well defined both for the single commodity and the multi-commodity cases.

We proceed to present the first problem. We are given a network with one or more commodities that need to transfer some flow demand subject to some given QoS requirements. The goal of a QoS multipath routing scheme is to identify several paths for each commodity, each meeting the QoS requirement, such that the load over the most utilized links in the network is minimized. This can be formulated as follows.

Problem RMP (Restricted Multipath) Given are a network $G(V, E)$, for each link $e \in E$ a weight $w_e > 0$ and a capacity $c_e > 0$ and, for each commodity $(i, j) \in V \times V$, a demand $\gamma^{(i,j)}$ and a weight restriction $W^{(i,j)}$. Find a path flow that minimizes the network congestion factor, such that, if $P_1^{(i,j)} \subseteq P^{(i,j)}$ is the collection of all paths in $P^{(i,j)}$ that are assigned with a positive flow, then, for each $p \in P_1^{(i,j)}$, it holds that $W(p) \leq W^{(i,j)}$.

We shall later prove that Problem RMP is intractable. We will also show that the solution can be used in order to support end-to-end reliability requirements in multipath routing schemes.

Problem RMP did not limit the number of different paths over which a commodity is shipped. However, as explained in the Introduction, in practice it is essential to limit this number. Accordingly, we define the following *K-Path Routing (KPR)* problem.

Problem KPR (K-Path Routing) Given are a network $G(V, E)$, for each link $e \in E$ a capacity $c_e > 0$, and, for each commodity $(i, j) \in V \times V$, a demand $\gamma^{(i,j)}$ and a *split restriction* $K^{(i,j)}$. Find a path flow that minimizes the network congestion factor, such that, if $P_3^{(i,j)} \subseteq P^{(i,j)}$ is the collection of all paths in $P^{(i,j)}$ that are assigned with a positive flow, then $|P_3^{(i,j)}| \leq K^{(i,j)}$.

3. Solution of problem RMP

In this section we aim at solving problem RMP, i.e., the problem of minimizing congestion subject to additive QoS requirements. In addition, we present an important application that supports end-to-end reliability requirements. First we establish that the problem is intractable.

3.1 Intractability of Problem RMP.

We show that Problem RMP can be reduced to the *Partition* problem [23].

Theorem 3.1 Problem RMP is NP-hard.

Proof First, let us define the single-commodity case of problem RMP as a decision problem.

Given are a network $G(V, E)$, for each link $e \in E$, a weight $w_e > 0$ and a capacity $c_e > 0$, and, for a commodity $(s, t) \in V \times V$, a demand $\gamma > 0$ and a weight restriction W . Is there a path flow with network congestion factor of at most α such that, if path p transfers a positive amount of flow then $W(p) \leq W$?

Consider the following instance of the Partition problem; given an ordered set of elements a_1, a_2, \dots, a_{2n} that constitute a set A with size $s(a) \in \mathbb{Z}^+$ for each $a \in A$, find a subset $A' \subseteq A$ such that A' contains exactly one element of a_{2i-1}, a_{2i} for $1 \leq i \leq n$ and $\sum_{a \in A'} s(a) = \sum_{a \in A - A'} s(a)$.

We transform Partition to RMP as follows (see also Fig. 3.1):

- Given an element $a_i \in A$ with size $s(a_i)$, define a unit capacity link $u_i \rightarrow v_i$ with weight $s(a_i)$.
- For each link $u_{2i-1} \rightarrow v_{2i-1}$ $1 \leq i \leq n$, define a link $v_{2i-1} \rightarrow u_{2i+1}$ and a link $v_{2i-1} \rightarrow u_{2i+2}$. Assign to both a unit capacity and a zero weight.
- For each link $u_{2i} \rightarrow v_{2i}$, $1 \leq i \leq n$, define a link $v_{2i} \rightarrow u_{2i+1}$ and a link $v_{2i} \rightarrow u_{2i+2}$. Assign to both a unit capacity and a zero weight.
- Define links $s \rightarrow u_1, s \rightarrow u_2$ and links $v_{2n-1} \rightarrow t, v_{2n} \rightarrow t$. Assign to each a unit capacity and a zero weight.
- Set: $W \leftarrow \frac{1}{2} \cdot \sum_{a \in A} s(a)$ and $\gamma \leftarrow 2$.

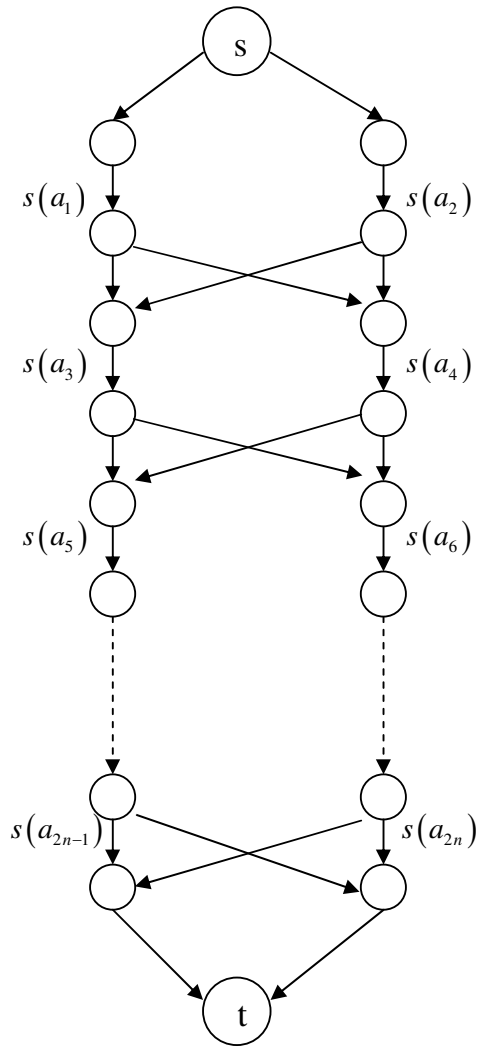


Fig. 3.1: Reduction of Partition to RMP

We shall prove that it is possible to transfer 2 flow units over paths whose weights are not larger than W without exceeding the network congestion factor of $\alpha = 1$ *iff* there is a subset $A' \subseteq A$ such that A' contains exactly one element of a_{2i-1}, a_{2i} for $1 \leq i \leq n$ and $\sum_{a \in A'} s(a) = \sum_{a \in A-A'} s(a)$. (Remark: We refer to elements and their sizes interchangeably.)

\Leftarrow : Suppose there is a subset $A' \subseteq A$ such that A' contains exactly one of a_{2i-1}, a_{2i} for $1 \leq i \leq n$ and $\sum_{a \in A'} s(a) = \sum_{a \in A-A'} s(a)$. Then, it is easy to see that the selection of the links that represents the elements in A' and the zero weight links that connect those links constitutes a path. Also, it is easy to see that this path is disjoint to the path that the complement subset $A - A'$ defines. Since all capacities equal to 1, we have two disjoint paths that can transfer together exactly 2 units of flow without violating the

congestion constraint $\alpha = 1$. The weight restriction is preserved since the two defined paths have weight of $\frac{1}{2} \cdot \sum_{a \in A} s(a)$, which was defined to be the weight restriction W .

\Rightarrow : Suppose there is a path flow that transfers two flow units over paths that are not bigger than W . It is easy to see that all paths in the graph must be simple since the graph is a DAG. Select one path that transfers a positive flow and denote it as p . Define an empty set S . For every link in p , with weight $s(a_i)$, insert the element a_i into S . Since all links in the graph have one unit of capacity, the selected path p is not able to transfer more than one unit of flow. Now, delete all the links that constitute path p . Since p is simple and since it transfers at most one unit of flow, there must be another path that is disjoint to the selected path that transfers a positive flow over the links that were left in the graph. For each link in that path with size $s(a_i)$, insert the element a_i into a different set S' . We will now prove that $A = S \cup S'$, $S \cap S' = \emptyset$ and, finally, $\sum_{a \in S} s(a) = \sum_{a \in S'} s(a)$.

Since S and S' were constructed out of disjoint paths, it is obvious that $S \cap S' = \emptyset$. Since every path must traverse either $s(a_{2i-1})$ or $s(a_{2i})$ for each $1 \leq i \leq n$, and since both paths are disjoint, $S \cup S' = \{a_i\}_{i=1}^{2n} = A$.

Since both paths have weights that are not longer than W , we have:

$$\sum_{a \in S} s(a) \leq W, \quad \sum_{a \in S'} s(a) \leq W. \quad (1)$$

Since $W \triangleq \frac{1}{2} \cdot \sum_{a \in A} s(a)$ and $S \cup S' = A$, we get:

$$\sum_{a \in S} s(a) + \sum_{a \in S'} s(a) = \sum_{a \in A} s(a) = 2 \cdot W. \quad (2)$$

Note that if variables x_1, x_2 satisfy $x_1 \leq B$, $x_2 \leq B$ and in addition $x_1 + x_2 = 2 \cdot B$, it follows that $x_1 = x_2 = B$. Accordingly, we conclude from (1) and (2) that

$$\sum_{a \in S} s(a) = \sum_{a \in S'} s(a) = W.$$

Thus, problem RMP is NP hard. ■

3.2 Linear Programming Formulation

In this section we present a linear programming formulation for Problem RMP. To that end, we need some additional notation.

We are given a network $G(V, E)$, for each link $e \in E$ a weight $w_e \in \mathbb{Z}^+$ and a capacity $c_e > 0$, and, for each commodity $(i, j) \in V \times V$ a demand $\gamma^{(i,j)}$ and a weight restriction $W^{(i,j)}$. Let α be the network congestion factor. Define $f_e^{\omega, (i,j)}$ as the flow of commodity (i, j) over link $e = (u, v) \in E$ that has traversed through paths $p \in P^{(i,u)}$ of total weight $W(p) = \omega$. Finally, for each $v \in V$, denote by $O(v)$ the set of links that emanate from v , and by $I(v)$ the set of links that enter that node, namely $O(v) = \{(v, l) | (v, l) \in E\}$ and $I(v) = \{(w, v) | (w, v) \in E\}$. Then, Problem RMP can be formulated as the following linear program:

<p>Minimize α</p> <p>s.t</p> $\sum_{e \in O(v)} f_e^{\omega, (i,j)} - \sum_{e \in I(v)} f_e^{\omega - w_e, (i,j)} = 0 \quad , \forall (i, j) \in \beta \quad , \forall v \in V - \{i, j\} \quad , \forall \omega \in [0, W^{(i,j)}] \quad (1)$ $\sum_{e \in O(i)} f_e^{\omega, (i,j)} - \sum_{e \in I(i)} f_e^{\omega - w_e, (i,j)} = 0 \quad , \forall (i, j) \in \beta \quad , \forall \omega \in [1, W^{(i,j)}] \quad (2)$ $\sum_{e \in O(i)} f_e^{0, (i,j)} = \gamma^{(i,j)} \quad , \forall (i, j) \in \beta \quad (3)$ $\sum_{(i,j) \in \beta} \sum_{\omega=0}^{W^{(i,j)}} f_e^{\omega, (i,j)} \leq c_e \cdot \alpha \quad , \forall e \in E \quad (4)$ $f_e^{\omega, (i,j)} = 0 \quad , \forall \omega < 0 \quad , \forall (i, j) \in \beta \quad , \forall e \in E \quad (5)$ $f_e^{\omega, (i,j)} \geq 0 \quad , \forall (i, j) \in \beta \quad , \forall \omega \in [0, W^{(i,j)}] \quad , \forall e \in E \quad (6)$ $\alpha \geq 0 \quad (7)$

Fig. 3.2 Program RMP

Note that the variables of the linear program are the link flows $\{f_e^{\omega, (i,j)}\}$ and the network congestion factor α .

In this linear programming formulation the objective function is to minimize the network congestion factor. Constraints (1), (2) and (3) are nodal flow conservation constraints. Equation (1) states that the traffic flowing out of node v , which has traversed through paths $p \in P^{(i,v)}$ of weight $W(p) = \omega$, has to be equal to the traffic flowing into node v , through paths $p' \in P^{(i,u)}$ and links $e = (u, v) \in E$, such that $W(p') + w_e = \omega$; since $\omega \in [0, W^{(i,j)}]$, the weight restriction is preserved for each commodity $(i, j) \in \beta$; finally, equation (1) must be satisfied for each node other than the source node and the destination node for each commodity with a positive demand.

Equation (2) extends for each $(i, j) \in \beta$ the validation of equation (1) to hold for traffic that encounters node i after it has already traveled over paths with non-zero weights. Informally, equation (2) states that "old" traffic that was already traversing over at least one link must satisfy constraint (1). Equation (3) states that, for each commodity $(i, j) \in \beta$, the traffic flowing out of source i , which has already traversed paths of weight $\omega = 0$, must be equal to the demand $\gamma^{(i,j)}$. Equation (4) is the link capacity utilization constraint. Expression (5) rules out non-feasible flows, and Expression (6) and (7) restricts all variables to be non-negative.

We can solve Program RMP using any polynomial time algorithm for linear programming [24]. The solution to problem RMP is then achieved by decomposing the link flow $\{f_e^{\omega,(i,j)}\}$ into a path flow that satisfies the weight restriction. This is done by Algorithm PFC, specified in Fig. 3.3, which is an (elaborated) generalization of the Flow Decomposition Algorithm [12].

For each commodity $(i, j) \in V \times V$ the algorithm uses the link flow $\{f_e^{\omega,(i,j)}\}$ in order to define paths that transfer a total flow of $\gamma^{(i,j)}$ flow units over paths that have total weight not larger than $W^{(i,j)}$. At each iteration, the algorithm uses Procedure Path Construction, specified in Fig. 3.4, in order to define a path with end-to-end weight of at most $W^{(i,j)}$ whose corresponding link flows $\{f_e^{\omega,(i,j)}\}$ are all positive. The flow over this path is defined to be equal to the smallest flow $f_e^{\omega,(i,j)}$ that belongs to the path. Then, the algorithm subtracts the flow that traverses through that path from the demand $\gamma^{(i,j)}$ and from each variable $f_e^{\omega,(i,j)}$ in the path. The algorithm stops when the demand $\gamma^{(i,j)}$ is zeroed for each $(i, j) \in V \times V$. Thus, the resulting path flow transfers $\gamma^{(i,j)}$ flow units from source i to destination j over paths with weight of at most $W^{(i,j)}$ for each $(i, j) \in V \times V$.

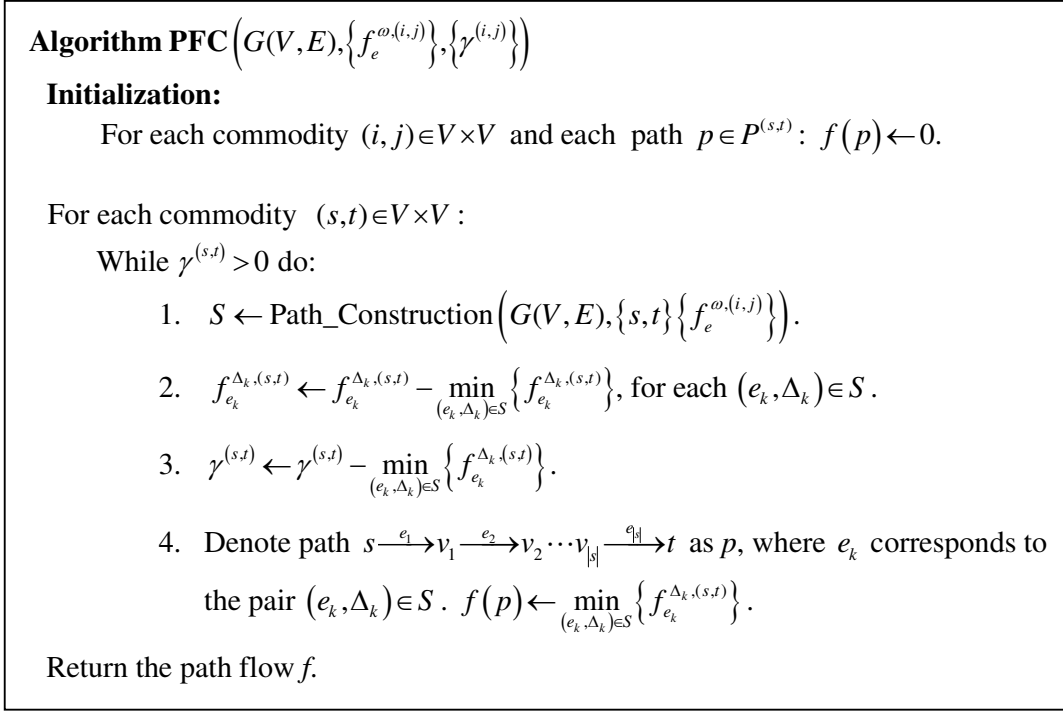


Fig 3.3: Algorithm Path Flow Construction (PFC)

We turn to explain the main idea behind Procedure Path Construction, which is specified in Fig. 3.4. The procedure identifies a path $p: s \xrightarrow{e_1} u_1 \xrightarrow{e_2} u_2 \cdots u_{h-1} \xrightarrow{e_h} t$ whose corresponding link flows $\{f_{e_1}^{0(s,t)}, f_{e_2}^{w_{e_1}(s,t)}, \dots, f_{e_h}^{w_{e_1} + \dots + w_{e_{h-1}}(s,t)}\}$ are all positive. This is done by employing the following property that characterizes the solution to Program RMP. If a *positive* flow $f_e^{\omega-w_e(s,t)}$ enters through link $e=(u,v)$ into node $v \in V - \{s,t\}$, then there exists a *positive* flow $f_{e'}^{\omega(s,t)}$ that emanates out of v through a link $e'=(v,w)$. Since each positive flow $f_e^{\omega(s,t)}$ must satisfy $\omega \leq W^{(s,t)}$ it follows that, if we follow these positive flows from the source s , we establish a path that satisfies the weight restriction $W^{(s,t)}$. We now prove that if we follow these positive flows for a finite number of times we eventually find a positive flow that enters into the destination i.e., we eventually identify a directed path from s to t . We note that this path is not necessarily simple.

Lemma 1: Consider the nodes $\{u_k\}$ identified by Procedure Path Construction (Fig. 3.4) for the input $\langle G(V, E), \{s, t\}, \{f_e^{\omega(i,j)}\} \rangle$. There exists an h , $h \leq W^{(s,t)} / \min_{e \in E} \{w_e\}$, such that the sequence (u_0, u_1, \dots, u_h) is a path from s to t with a weight of at most $W^{(s,t)}$.

Proof: It follows from constraint (3) that, if $\gamma^{(s,t)} > 0$, then there exists some link $e_0=(u_0, u_1)$ such that the variable $f_{e_0}^{0(s,t)}$ is positive. Then, from constraints (1) and (2), it follows that, if $u_1 \neq t$, then there exists some link $e_1=(u_1, u_2)$ such that the variable $f_{e_1}^{w_{e_0}(s,t)}$ is positive. Thus, applying constraints (1) and (2) for any index k , it follows

that, if there exists a positive variable $f_{e_k}^{\Delta_k, (s,t)}$ where $\Delta_k \triangleq \sum_{l \in [0, k-1]} W_{e_l}$, then, unless $u_{k+1}=t$, there exists a link $e_{k+1}=(u_{k+1}, u_{k+2})$ such that the variable $f_{e_{k+1}}^{\Delta_k + d_{e_k}, (s,t)}$ is positive. However, since it follows from constraint (5) that $f_e^{\omega, (s,t)} = 0$ for each $\omega > W^{(s,t)}$, and since $\Delta_k > W^{(s,t)}$ for each $k \geq W^{(s,t)} / \min_{e \in E} \{w_e\}$, it follows that there exists a k , $k \leq W^{(s,t)} / \min_{e \in E} \{w_e\}$, such that $u_k=t$. Finally, as the procedure selects only positive variables $f_e^{\omega, (s,t)}$, it follows from (5) and (6) that $\omega \in [0, W^{(s,t)}]$. Thus the path (u_0, u_1, \dots, u_k) has a weight of at most $W^{(s,t)}$. ■

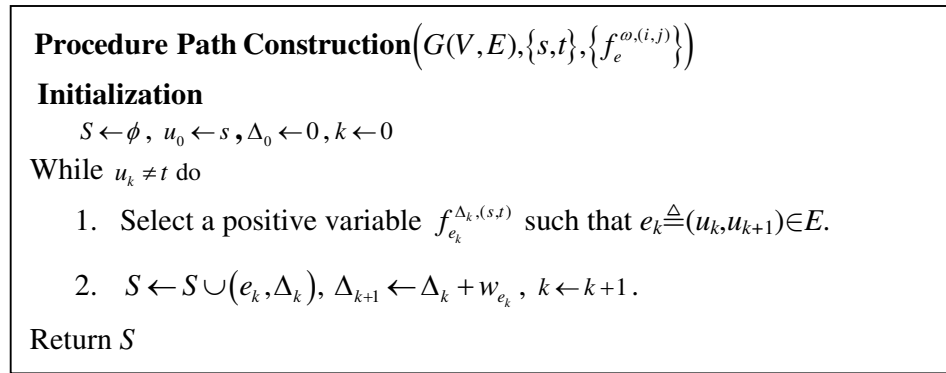


Fig 3.4: Procedure Path Construction

Note that each iteration of Algorithm PFC zeroes at least one variable $f_e^{\omega, (i,j)}$. Therefore, Algorithm PFC iterates for no more than the number of variables $\{f_e^{\omega, (i,j)}\}$. In addition, it follows from Lemma 1 that the complexity of Procedure Path Construction is at most $W^{(s,t)}$ for the input $\langle G(V, E), \{s, t\}, \{f_e^{\omega, (i,j)}\} \rangle$. Since $W^{(s,t)} \leq \left| \{f_e^{\omega, (i,j)}\} \right|$, it follows that Algorithm PFC has a polynomial complexity with respect to the number of variables $\{f_e^{\omega, (i,j)}\}$ that are used by Program RMP.

For completion, in Fig. 3.5 we specify Algorithm RMP, which solves Problem RMP.

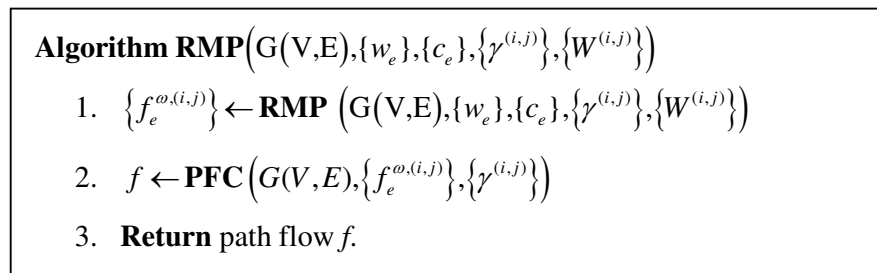


Fig 3.5 Algorithm RMP

In Algorithm RMP, the complexity incurred by solving Program RMP (step 1) is polynomial in the number of variables $\{f_e^{\omega, (i,j)}\}$ [24]. In addition, as shown, the

complexity of Algorithm PFC (step 2) is polynomial in $\left| \left\{ f_e^{\omega,(i,j)} \right\} \right|$. Thus, the complexity of Algorithm RMP is polynomial in the number of variables $\left\{ f_e^{\omega,(i,j)} \right\}$ that are used by Program RMP.

Nonetheless, each commodity with a positive demand $\gamma^{(i,j)}$ has at least $M \cdot W^{(i,j)}$ variables. Hence, the total number of variables that are used in order to formulate the problem is $M \cdot \sum_{(i,j) \in \beta} W^{(i,j)}$, which may be exponential with respect to the input. Note, however, that when the hop count metric is considered (i.e., $w_e \equiv 1$), the number of variables formulating the problem is polynomial. Therefore, for that case we have a polynomial solution.

3.3 Approximation Scheme for Problem RMP

In this section, we establish an approximation scheme for problem RMP. Specifically, we present an approximation scheme for the case where the weight restrictions $W^{(i,j)}$ are of the same order of magnitude for each $(i,j) \in \beta$, namely $\frac{W^{(i_1,j_1)}}{W^{(i_2,j_2)}} = O(1)$ for every pair of commodities $(i_1,j_1), (i_2,j_2) \in \beta$. We note that since end-to-end delays are normally in the order of milliseconds, the case of $\frac{W^{(i_1,j_1)}}{W^{(i_2,j_2)}} = O(1)$ for every pair of commodities $(i_1,j_1), (i_2,j_2) \in \beta$ is the typical case.

The following approximation scheme (Fig. 3.6) reduces the complexity of Algorithm RMP by reducing the number of variables that are used by Program RMP. Since we have already seen that the number of variables is $M \cdot \sum_{(i,j) \in \beta} W^{(i,j)}$, it follows that, in order to have a polynomial number of variables (and therefore polynomial complexity) it is essential to reduce the value of each weight restriction. To that end, we scale the weight restriction of each commodity into a smaller integral value. As a result, we must also scale the weight of each link. However, in order to ensure that the optimal network congestion factor does not increase, we relax the new weight restriction with respect to its original value. This is done by *rounding up* the weight restrictions and *rounding down* the weight of each link.

RMP Approximation Scheme $\left(G, \{c_e\}_{e \in E}, \{w_e\}_{e \in E}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta}, \{W^{(i,j)}\}_{(i,j) \in \beta}, \varepsilon \right)$

Parameters :

G – network

$\{c_e\}_{e \in E}$ – capacities

$\{w_e\}_{e \in E}$ – weights

$\{\gamma^{(i,j)}\}_{(i,j) \in \beta}$ – demands

$\{W^{(i,j)}\}_{(i,j) \in \beta}$ – weight restrictions

ε – approximation parameter

variables

W^{\max} – largest weight restriction

W^{\min} – smallest weight restriction

$\tilde{G}(\tilde{V}, \tilde{E})$ – auxiliary graph

$$1 \quad W^{\min} \leftarrow \min_{(i,j) \in \beta} \{W^{(i,j)}\}, W^{\max} \leftarrow \max_{(i,j) \in \beta} \{W^{(i,j)}\}$$

$$2 \quad \delta \leftarrow \frac{W^{\min} \cdot \varepsilon}{N}$$

3 Construct network $\tilde{G}(\tilde{V}, \tilde{E})$ as follows:

a. $\tilde{V} \leftarrow V$

b. $\tilde{E} \leftarrow \{e \in E \mid w_e \leq W^{\max}\}$

c. For each $\tilde{e} \in \tilde{E}$:

$$\tilde{w}_e \leftarrow \left\lfloor \frac{w_e}{\delta} \right\rfloor, \tilde{c}_e \leftarrow c_e.$$

4 For each commodity $(i,j) \in \beta$:

$$\widetilde{W}^{(i,j)} \leftarrow \left\lceil \frac{W^{(i,j)}}{\delta} \right\rceil.$$

5 Solve instance $\left\langle \tilde{G}, \{\tilde{c}_e\}_{e \in \tilde{E}}, \{\tilde{w}_e\}_{e \in \tilde{E}}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta}, \{\widetilde{W}^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$

of problem RMP using Algorithm RMP. Let path flow g represent the solution.

6 Construct a path flow f as follows:

a. Let $h: P^{(i,j)} \rightarrow P_{\text{simple}}^{(i,j)}$ map each path $(v_0, v_1, \dots, v_{h-1}, v_h, \dots, v_h, v_{h+1}, \dots, v_n) \in P^{(i,j)}$ into a simple path $(v_0, v_1, \dots, v_{h-1}, v_h, v_{h+1}, \dots, v_n) \in P_{\text{simple}}^{(i,j)}$.

b. For each $p \in P_{\text{simple}}^{(i,j)}$, $f(p) \triangleq \sum_{p' \mid h(p')=p} g(p')$

7. Return path flow f .

Fig. 3.6 RMP Approximation Scheme

Definition 3.1 Given an instance of problem RMP, α^* is the network congestion factor of the optimal solution.

Theorem 3.2 Given an instance $\left\langle G, \{c_e\}_{e \in E}, \{w_e\}_{e \in E}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta}, \{W^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$ of problem RMP and an approximation parameter ε , the output of the *RMP Approximation Scheme*, specified in Fig. 3.6, is a path flow f that satisfies the following:

- For each commodity $(i, j) \in \beta$, $\sum_{p \in P^{(i,j)}} f(p) = \gamma^{(i,j)}$ i.e., the flow demand requirement is satisfied for all commodities.
- If α^* is the network congestion factor of the optimal solution, then, for each $e \in E$, it holds that $\sum_{(i,j) \in \beta} \sum_{p \in P^{(i,j)}} \Delta_e(p) \cdot f(p) \leq \alpha^* \cdot c_e$, i.e., the network congestion factor is at most α^* .
- For each commodity $(i, j) \in \beta$, if $p \in P^{(i,j)}$ and $f(p) > 0$ then $W(p) \leq (1 + \varepsilon) \cdot W^{(i,j)}$ i.e., the end-to-end delay is violated by a factor of at most $(1 + \varepsilon)$.

Proof

- Since equation (3) in Program RMP specifies for each commodity $(i, j) \in \beta$, $\sum_{e \in O(i)} f_e^{0,(i,j)} = \gamma^{(i,j)}$, it follows that every path flow representation that corresponds to link flow $\{f_e^{\omega,(i,j)}\}$ must satisfy for each commodity $(i, j) \in \beta$, $\sum_{p \in P^{(i,j)}} f(p) = \gamma^{(i,j)}$.
- Since the algorithm rounds down the link weights and rounds up the weight restrictions, it follows that the resulting instance $\left\langle \tilde{G}, \tilde{w}_e, \tilde{c}_e, \{\gamma^{(i,j)}\}_{(i,j) \in \beta}, \{\tilde{W}^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$ of problem RMP (that appears in line (5)) *relaxes* the original constraints. Hence, the resulting network congestion factor is at most α^* .
- Define θ_e to be the discretization error resulted from rounding the weight w_e of link $e \in E$ to a multiple of δ i.e., for each link $e \in E$, $\theta_e \triangleq w_e - \left\lfloor \frac{w_e}{\delta} \right\rfloor \cdot \delta$.
Note that $\theta_e = w_e - \left\lfloor \frac{w_e}{\delta} \right\rfloor \cdot \delta \leq w_e - \left(\frac{w_e}{\delta} - 1 \right) \cdot \delta = \delta$. Since by construction, the paths that are used by path flow f are simple, every such path contains at most $N - 1$ links. Therefore, the total error in evaluating the weight of these paths is at most $(N - 1) \cdot \delta$. Applying similar considerations, it is easy to see that the discretization error resulting from rounding the weight restrictions to multiples of δ is at most δ . Therefore, each path $p \in P^{(i,j)}$ that is used by

path flow f (i.e., $f(p) > 0$) has total weight of at most $W^{(i,j)} + N \cdot \delta = W^{(i,j)} + N \cdot \frac{W^{\min} \cdot \varepsilon}{N} = W^{(i,j)} + W^{\min} \cdot \varepsilon \leq W^{(i,j)} (1 + \varepsilon)$. ■

Given an instance of Problem RMP and an approximation parameter ε , we now show that the proposed approximation scheme for Problem RMP is polynomial.

Theorem 3.3 Given an instance $\left\langle G, \{c_e\}_{e \in E}, \{w_e\}_{e \in E}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta}, \{W^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$ of problem RMP and an approximation parameter ε , RMP Approximation Scheme has a polynomial complexity with respect to the input and the approximation parameter ε .

Proof Clearly, the complexity of RMP Approximation Scheme is determined by step (5). As was already explained in section 3.2, the complexity of this step is polynomial in the number of variables that are used by Program RMP to solve the corresponding instance of step (5). Thus, in order to prove the theorem, we only need to show that Program RMP solves instance $\left\langle \tilde{G}, \{\tilde{c}_e\}_{e \in \tilde{E}}, \{\tilde{w}_e\}_{e \in \tilde{E}}, \{\tilde{\gamma}^{(i,j)}\}_{(i,j) \in \beta}, \{\tilde{W}^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$ of step (5), using a polynomial number of variables with respect to the input and the value of ε .

We first prove that, if $w_e \leq W^{\max}$ for each $e \in E$, then the number of variables considered by Program RMP is at most $2 \cdot M \cdot |\beta| \cdot W^{\max}$. To that end, consider the set of variables $\{f_e^{\omega, (i,j)}\}$ that formulates Program RMP. For a link $e \in E$ and a commodity $(i, j) \in \beta$, the number of variables $f_e^{\omega, (i,j)}$ that are used by Program RMP is determined by the number of values that ω can take. Since equations (1) - (7) of Program RMP refer both to the variable $f_e^{\omega - w_e, (i,j)}$ and to the variable $f_e^{\omega, (i,j)}$, for each $\omega \in [0, W^{(i,j)}]$, then for a given commodity $(i, j) \in \beta$ and a link $e \in E$, the set $\{f_e^{\hat{\omega}, (i,j)}\}$ where $\hat{\omega} \in [0, W^{(i,j)}] \cup [-w_e, W^{(i,j)} - w_e] = [-w_e, W^{(i,j)}]$ contains all the variables that are used by Program RMP. Since $w_e \leq W^{\max}$ for each $e \in E$, it follows that the linear program uses, for a commodity $(i, j) \in \beta$ and a link $e \in E$, at most $w_e + W^{(i,j)} \leq W^{\max} + W^{(i,j)} \leq 2 \cdot W^{\max}$ different variables. Thus, the total number of variables in Program RMP is at most $\sum_{e \in E} \sum_{(i,j) \in \beta} 2 \cdot W^{\max} = 2 \cdot \sum_{e \in E} \sum_{(i,j) \in \beta} W^{\max} = 2 \cdot M \cdot \sum_{(i,j) \in \beta} W^{\max} = 2 \cdot M \cdot |\beta| \cdot W^{\max}$.

We shall now use this finding, namely that, if $w_e \leq W^{\max}$ for each $e \in E$, then the number of variables for an instance $\left\langle G, \{c_e\}_{e \in E}, \{w_e\}_{e \in E}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta}, \{W^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$ of Problem RMP is $2 \cdot M \cdot |\beta| \cdot W^{\max}$, in order to show that the number of variables for the

instance $\left\langle \tilde{G}, \{\tilde{c}_e\}_{e \in \tilde{E}}, \{\tilde{w}_e\}_{e \in \tilde{E}}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta}, \{\widetilde{W}^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$ of step (5) is $O\left(\frac{N \cdot M \cdot |\beta|}{\varepsilon}\right)$. Since, by construction, $\tilde{E} = \{e \in E \mid w_e \leq W^{\max}\}$, it follows that, for each $\tilde{e} \in \tilde{E}$, $\tilde{w}_e = \left\lfloor \frac{w_e}{\delta} \right\rfloor \leq \left\lceil \frac{W^{\max}}{\delta} \right\rceil = \widetilde{W}^{\max}$. Thus, the number of variables in Program RMP for instance $\left\langle \tilde{G}, \{\tilde{c}_e\}_{e \in \tilde{E}}, \{\tilde{w}_e\}_{e \in \tilde{E}}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta}, \{\widetilde{W}^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$ is at most $2 \cdot M \cdot |\beta| \cdot \widetilde{W}^{\max}$. Hence, since $\widetilde{W}^{\max} = \left\lceil \frac{W^{\max}}{\delta} \right\rceil$, it follows that this number of variables is at most $2 \cdot M \cdot |\beta| \cdot \left(\frac{W^{\max}}{\delta} + 1\right)$. From the assumption that $\frac{W^{(i_1, j_1)}}{W^{(i_2, j_2)}} = O(1)$, for each $(i_1, j_1), (i_2, j_2) \in \beta$, it follows that $\frac{W^{\max}}{W^{\min}} = O(1)$. Therefore, since $\delta \triangleq \frac{W^{\min} \cdot \varepsilon}{N}$, the total number of variables that are used by Program RMP in order to solve instance $\left\langle \tilde{G}, \{\tilde{c}_e\}_{e \in \tilde{E}}, \{\tilde{w}_e\}_{e \in \tilde{E}}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta}, \{\widetilde{W}^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$ of step (5) is at most $2 \cdot M \cdot |\beta| \cdot \left(\frac{W^{\max}}{\delta} + 1\right) = 2 \cdot M \cdot |\beta| \cdot \left(\frac{W^{\max}}{W^{\min}} \cdot \frac{N}{\varepsilon} + 1\right) = O\left(\frac{N \cdot M \cdot |\beta|}{\varepsilon}\right)$. ■

3.4 Applications for Program RMP

Problem RMP may arise in several forms. In the single-commodity case, it adds an additive restriction to the well-known Maximum Flow Problem, which applies to paths that carry a positive flow. This restriction may be important in multipath routing schemes where additive QoS metrics, such as delay and jitter, are considered. In this section, we show that Program RMP can be used in order to support multipath routing with end-to-end reliability requirements, i.e., when we need multipath routing schemes that choose paths with a "good" probability of success.

The notion of reliability can be implemented by assigning to each link in the network a failure probability and restricting all paths that carry positive flow to have an end-to-end success probability that is larger than some given lower bound. This is formulated by the following problem.

Problem ReMP (Reliable Multipath) Given are a network $G(V, E)$, for each link $e \in E$, a failure probability $p_e \geq 0$ and a capacity $c_e > 0$, and, for each commodity $(i, j) \in \beta$, a demand $\gamma^{(i,j)} > 0$ and a success probability restriction $\Pi^{(i,j)}$. Find a path flow $f: P \rightarrow \mathbb{R}^+ \cup \{0\}$ that minimizes the network congestion factor such that, if $p \in P^{(i,j)}$ and $f(p) > 0$, then $\prod_{e \in p} (1 - p_e) \geq \Pi^{(i,j)}$.

We establish an approximation scheme for Problem ReMP that reduces Problem ReMP into Problem RMP, as follows. The approximation scheme considers a set of classes $\left\{1, \left(1 + \frac{\varepsilon}{N}\right)^{-1}, \left(1 + \frac{\varepsilon}{N}\right)^{-2}, \dots\right\}$ where $\varepsilon > 0$ is a given approximation parameter.

Then, each link with a success probability of $1 - p_e$ is assigned with a weight i if $\left(1 + \frac{\varepsilon}{N}\right)^{-(i+1)} \leq 1 - p_e \leq \left(1 + \frac{\varepsilon}{N}\right)^{-i}$. Thus, restricting the end-to-end weight of each

simple path to B results with an end-to-end probability of at least $\left(1 + \frac{\varepsilon}{N}\right)^{-(B+N)}$. Then,

the value of B is determined from the success probability restriction $\Pi^{(i,j)}$. This value is set to relax the original instance of Problem ReMP in order to get a network congestion factor that is not larger than the optimal network congestion factor. Finally, the number of variables that are produced for the corresponding instance of Problem RMP is determined by the approximation parameter ε . Fig. 3.7 describes the approximation scheme for Problem ReMP.

ReMP Approximation Scheme $\left\langle G(V, E), \{p_e\}_{e \in E}, \{c_e\}_{e \in E}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta}, \{\Pi^{(i,j)}\}_{(i,j) \in \beta}, \varepsilon \right\rangle$

Parameters :

$G(V, E)$ – network

$\{p_e\}_{e \in E}$ – failure probabilities

$\{c_e\}_{e \in E}$ – capacities

$\{\gamma^{(i,j)}\}_{(i,j) \in \beta}$ – demands

$\{\Pi^{(i,j)}\}_{(i,j) \in \beta}$ – probability restrictions

ε – approximation parameter

1. Delete all links with failure probability $p_e = 1$. Let \tilde{E} be the resulting set of links, and let p_{\max} be the link with the largest failure probability that is smaller than 1.

2. Define the following set of classes: $\left\{1, \left(1 + \frac{\varepsilon}{N}\right)^{-1}, \left(1 + \frac{\varepsilon}{N}\right)^{-2}, \dots, \left(1 + \frac{\varepsilon}{N}\right)^k, 0\right\}$,

$$\text{where } k = \left\lceil \log_{1 + \frac{\varepsilon}{N}}(1 - p_{\max}) \right\rceil.$$

3. To each link $e \in \tilde{E}$ that satisfies $\left(1 + \frac{\varepsilon}{N}\right)^{-(i+1)} \leq 1 - p_e \leq \left(1 + \frac{\varepsilon}{N}\right)^{-i}$, assign a weight $w_e = i$.

4. For each probability restriction that satisfies $\left(1 + \frac{\varepsilon}{N}\right)^{-(i+1)} \leq \Pi^{(i,j)} \leq \left(1 + \frac{\varepsilon}{N}\right)^{-i}$, assign a weight restriction $W^{(i,j)} = i + 1$.

5. Solve the instance $\left\langle G(V, \tilde{E}), \{c_e\}_{e \in \tilde{E}}, \{w_e\}_{e \in \tilde{E}}, \{W^{(i,j)}\}_{(i,j) \in \beta}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$ of problem RMP using Algorithm RMP.

6. Construct a path flow f as follows:

a. Let $h: P^{(i,j)} \rightarrow P_{\text{simple}}^{(i,j)}$ map each path $(v_0, v_1, \dots, v_{h-1}, v_h, \dots, v_h, v_{h+1}, \dots, v_n) \in P^{(i,j)}$ into a simple path $(v_0, v_1, \dots, v_{h-1}, v_h, v_{h+1}, \dots, v_n) \in P_{\text{simple}}^{(i,j)}$.

b. For each $p \in P_{\text{simple}}^{(i,j)}$, $f(p) \triangleq \sum_{p' | h(p')=p} g(p')$.

7. Return the path flow f .

Fig. 3.7 ReMP Approximation Scheme

Theorem 3.4 Given an instance $\left\langle G(V, E), \{p_e\}_{e \in E}, \{c_e\}_{e \in E}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta}, \{\Pi^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$ of Problem ReMP and an approximation parameter ε , the *ReMP Approximation Scheme* satisfies the following:

- a. For each commodity $(i, j) \in \beta$, $\sum_{p \in P^{(i,j)}} f(p) = \gamma^{(i,j)}$ i.e., the flow demand requirement is satisfied for all commodities.
- b. The network congestion factor of the output is not larger than the network congestion factor of the optimal solution.
- c. If the output assigns to a path $p \in P^{(i,j)}$ a positive flow then $\prod_{e \in p} (1 - p_e) \geq \frac{\Pi^{(i,j)}}{(1 + \varepsilon)}$; i.e. the restriction on the probability of success is relaxed by at most a factor of $(1 + \varepsilon)$.
- d. The scheme has a polynomial complexity with respect to the input and the given approximation parameter ε .

Proof

- a. In step (5) of the algorithm, an instance $\left\langle G(V, \tilde{E}), \{c_e\}_{e \in \tilde{E}}, \{w_e\}_{e \in \tilde{E}}, \{W^{(i,j)}\}_{(i,j) \in \beta}, \{\gamma^{(i,j)}\}_{(i,j) \in \beta} \right\rangle$ of Problem RMP is solved by Algorithm RMP that satisfies the flow demand requirements for each commodity $(i, j) \in \beta$. Thus, the returned path flow f satisfies the flow demand requirements of all commodities.
- b. Let α be the network congestion factor of the resulting path flow and let α^* be the network congestion factor of the optimal solution. We have to prove that $\alpha \leq \alpha^*$. To that end, consider the given instance of Problem ReMP and the corresponding instance of Problem RMP that was specified in step (5) of the ReMP Approximation Scheme. We will prove that the set of feasible paths of instance ReMP is contained in the set of feasible paths of instance RMP. Proving that will show that the constraints over the transformed instance of Problem RMP are relaxed with respect to the original constraints over the given instance of Problem ReMP. To that end, we consider a feasible path p with respect to the given instance of Problem ReMP. We will show that path p is also feasible with respect to the instance of Problem RMP in step (5). Since p is a feasible path with respect to the given instance of Problem ReMP, it satisfies $\prod_{e \in p} (1 - p_e) \geq \Pi^{(i,j)}$. By the transformation to problem RMP, each link $e \in E$ with a probability p_e such that $\left(1 + \frac{\varepsilon}{N}\right)^{-(i+1)} \leq 1 - p_e \leq \left(1 + \frac{\varepsilon}{N}\right)^{-i}$ is assigned with a weight value $w_e = i$. Therefore, $(1 - p_e) \leq \left(1 + \frac{\varepsilon}{N}\right)^{-w_e}$. In addition, each end-to-end reliability

constraint $\Pi^{(i,j)}$ such that $\left(1 + \frac{\varepsilon}{N}\right)^{-(i+1)} \leq \Pi^{(i,j)} \leq \left(1 + \frac{\varepsilon}{N}\right)^{-i}$ is transformed into

a weight constraint $W^{(i,j)} = i+1$ that results with $\left(1 + \frac{\varepsilon}{N}\right)^{-W^{(i,j)}} \leq \Pi^{(i,j)}$.

Therefore, it follows that $\left(1 + \frac{\varepsilon}{N}\right)^{-W^{(i,j)}} \leq \Pi^{(i,j)} \leq \prod_{e \in p} (1 - p_e) \leq \left(1 + \frac{\varepsilon}{N}\right)^{-\sum_{e \in p} w_e}$.

Since $\left(1 + \frac{\varepsilon}{N}\right)^{-W^{(i,j)}} \leq \left(1 + \frac{\varepsilon}{N}\right)^{-\sum_{e \in p} w_e}$, we conclude that $\sum_{e \in p} w_e \leq W^{(i,j)}$. Thus, path p is also feasible with respect to the instance of Problem RMP that is specified in step (5).

- c. We prove now that, if path $p \in P^{(i,j)}$ is assigned with a positive amount of flow by the ReMP Approximation Scheme, then $\prod_{e \in p} (1 - p_e) \geq \frac{\Pi^{(i,j)}}{(1 + \varepsilon)}$. Denote the success probability of path p as $\Psi(p)$. Then,

$$\Psi(p) = \prod_{e \in p} (1 - p_e) \geq \prod_{e \in p} \left(1 + \frac{\varepsilon}{N}\right)^{-(w_e+1)}.$$

Since the solution to Problem RMP consists of simple paths, and since each simple path contains at most $N - 1$ links, we conclude that:

$$\Psi(p) \geq \prod_{e \in p} \left(1 + \frac{\varepsilon}{N}\right)^{-(w_e+1)} \geq \left(1 + \frac{\varepsilon}{N}\right)^{-(W(p)+(N-1))} \geq \left(1 + \frac{\varepsilon}{N}\right)^{-W^{(i,j)}} \cdot \left(1 + \frac{\varepsilon}{N}\right)^{-(N-1)} \quad (1)$$

From the way the algorithm transforms the probability restriction into a weight restriction, we conclude that

$$\left(1 + \frac{\varepsilon}{N}\right)^{-W^{(i,j)}} \leq \Pi^{(i,j)} \leq \left(1 + \frac{\varepsilon}{N}\right)^{-W^{(i,j)}+1}. \quad (2)$$

Therefore, from (1) and (2):

$$\begin{aligned} \Psi(p) &\geq \left(1 + \frac{\varepsilon}{N}\right)^{-W^{(i,j)}} \cdot \left(1 + \frac{\varepsilon}{N}\right)^{-(N-1)} = \left(1 + \frac{\varepsilon}{N}\right)^{-W^{(i,j)}+1} \cdot \left(1 + \frac{\varepsilon}{N}\right)^{-(N-1)-1} \geq \\ &\geq \Pi^{(i,j)} \left(1 + \frac{\varepsilon}{N}\right)^{-N} = \frac{\Pi^{(i,j)}}{\left(1 + \frac{\varepsilon}{N}\right)^N} \approx \frac{\Pi^{(i,j)}}{\left(1 + N \cdot \frac{\varepsilon}{N}\right)} = \frac{\Pi^{(i,j)}}{(1 + \varepsilon)}. \end{aligned}$$

- d. In order to prove that the complexity of the ReMP Approximation Scheme is polynomial, it suffices to show that the number of variables formulating the problem is polynomial with respect to the input and the approximation

parameter ε [24]. We saw in the proof for Theorem 3.3 that the number of variables formulating any instance of problem RMP is never larger than $2 \cdot M \cdot |\beta| \cdot W^{\max}$, where W^{\max} is the largest weight restriction. Since, by

construction, $W^{(i,j)} = i+1 \Leftrightarrow \left(1 + \frac{\varepsilon}{N}\right)^{-(i+1)} \leq \Pi^{(i,j)} \leq \left(1 + \frac{\varepsilon}{N}\right)^{-i}$, it follows

that $\Pi^{(i,j)} \leq \left(1 + \frac{\varepsilon}{N}\right)^{-(W^{(i,j)}-1)}$. Thus, $\log_{1+\frac{\varepsilon}{N}} \Pi^{(i,j)} \leq -(W^{(i,j)} - 1)$

$\Rightarrow W^{(i,j)} \leq 1 - \log_{1+\frac{\varepsilon}{N}} \Pi^{(i,j)}$. Let $\Pi^{\min} \triangleq \min_{(i,j) \in \beta} \{\Pi^{(i,j)}\}$. The largest weight

restriction, namely W^{\max} , must therefore satisfy,

$W^{\max} \leq 1 - \log_{1+\frac{\varepsilon}{N}} \Pi^{\min} \approx 1 - \frac{N}{\varepsilon} \log(\Pi^{\min})$. Therefore, the number of variables is

no larger than $2 \cdot M \cdot |\beta| \cdot \left(1 - \frac{N}{\varepsilon} \log(\Pi^{\min})\right) = O\left(\frac{N \cdot M \cdot |\beta| \cdot |\log(\Pi^{\min})|}{\varepsilon}\right)$. ■

4. Solution of Problem KPR

In this section we aim at solving Problem KPR i.e., the problem of minimizing the congestion subject to a restriction on the number of routing paths per destination. In the following subsection, we show that Problem KPR is NP-hard for the general case and polynomial if the restriction on the number of routing paths is larger than the number of links M (i.e., $K \geq M$). Therefore, in this section we present an efficient approximation for the case that $K < M$. We note that both the negative result regarding the computational intractability of Problem KPR and the approximation scheme that we establish for the problem address the single commodity case.

4.1 Intractability of Problem KPR

Theorem 4.1 Problem KPR is NP-hard for the single commodity case.

Proof First, let us define the single commodity case of Problem KPR as a decision problem.

Given are a network $G(V, E)$, two nodes s and t , a capacity $c_e > 0$ for each link $e \in E$, a demand γ and a *split restriction* K . Is there a path flow with a network congestion factor of at most α such that, if $P \subseteq P^{(s,t)}$ is the collection of all paths in $P^{(s,t)}$ that are assigned with a positive flow, then $|P| \leq K^{(i,j)}$?

The following problem, termed the *single source unsplittable flow problem*, was shown to be intractable in [18].

Given a network $G(V, E)$, a capacity $c_e > 0$ for each link $e \in E$, a set of commodities $(s, t_1), (s, t_2), \dots, (s, t_K)$ and a corresponding demand $\gamma^{(s,t_i)}$ for each $i \in [1, K]$. Is there an assignment of traffic to paths such that for each $1 \leq i \leq k$ the demand $\gamma^{(s,t_i)}$ is routed over a single path $p \in P^{(s,t_i)}$ without violating the capacity constraints?

We transform the Single Source Unsplittable Flow Problem to Problem KPR as follows (see also Fig. 4.1):

- We add an aggregated target T .
- For each $i \in [1, K]$ we add a link $t_i \rightarrow T$ with a capacity $\gamma^{(s,t_i)}$.

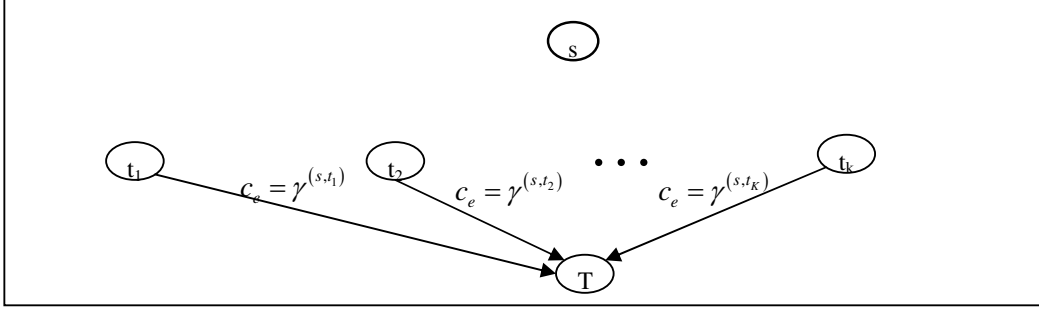


Fig. 4.1: Reducing the Single Source Unsplittable Flow Problem into Problem KPR

Claim: There exists a path flow that transfers $\gamma = \sum_{i=1}^K \gamma^{(s,t_i)}$ flow units from s to T over at most K paths without exceeding a network congestion factor of $\alpha = 1$ iff it is possible to find an assignment of the demands $\gamma^{(s,t_1)}, \gamma^{(s,t_2)}, \dots, \gamma^{(s,t_k)}$ to paths such that for each $1 \leq i \leq K$ the demand $\gamma^{(s,t_i)}$ is routed over a single path without violating the capacity constraints.

\Leftarrow : For each $1 \leq i \leq K$, there is exactly one path from s to t_i that carries $\gamma^{(s,t_i)}$ flow units without violating the capacity constraints. Hence, there are exactly K paths from s to T that transfers together $\gamma = \sum_{i=1}^K \gamma^{(s,t_i)}$ flow units without exceeding a network congestion factor of $\alpha = 1$.

\Rightarrow : It is possible to find an assignment of the demands $\gamma^{(s,t_1)}, \gamma^{(s,t_2)}, \dots, \gamma^{(s,t_k)}$ to paths such that for each $1 \leq i \leq K$ the demand $\gamma^{(s,t_i)}$ is routed over a single path without violating the capacity constraints. Hence, there exists a path flow that transfers $\gamma = \sum_{i=1}^K \gamma^{(s,t_i)}$ flow units from s to T over at most K paths without exceeding a network congestion factor of $\alpha = 1$. ■

Although the single commodity case of Problem KPR is NP hard, it is important to note that when the restriction on the number of paths is larger than the number of links M (i.e., $K \geq M$), the problem admits a polynomial solution. Specifically, the *flow decomposition algorithm* [12] can transform every link flow that transfers γ flow units into a path flow that transfers the same flow demand while employing at most M different routing paths; hence, there is no restriction on the number of paths when $K \geq M$. Thus, given an instance $\langle G(V, E), \{c_e\}, \{s, t\}, \gamma, K \rangle$ of Problem KPR, if $K \geq M$ it follows from Theorem 2.1 that it is sufficient to calculate a maximum flow $\{f_e\}$, multiply by the factor $\frac{\gamma}{F}$ the flow over each link $e \in E$, and decompose the resulting link flow (i.e., link flow $\left\{ f_e \cdot \frac{\gamma}{F} \right\}$) into a path flow. The resulting path flow employs at most M paths while minimizing the network congestion factor.

4.1 Problem σ -IR (σ - Integral Routing)

The solution to Problem KPR is based on a solution to an auxiliary problem that is formulated and solved in this subsection. We begin with the following definition.

Definition 4.1 Given are a network $G(V, E)$, for each link $e \in E$ a capacity $c_e > 0$, and, for each commodity $(i, j) \in V \times V$, a demand $\gamma^{(i,j)}$ and a value $\sigma^{(i,j)} \in \mathbb{R}^+$. The path flow of commodity (i, j) is said to be $\sigma^{(i,j)}$ -integral, if every path $p \in P^{(i,j)}$ is assigned with a flow value that is integral in $\sigma^{(i,j)}$.

Problem σ -IR (σ - Integral Routing) Given are a network $G(V, E)$, for each link $e \in E$ a capacity $c_e > 0$ and for each commodity $(i, j) \in V \times V$, a demand $\gamma^{(i,j)}$ and a value $\sigma^{(i,j)} \in \mathbb{R}^+$. For each commodity $(i, j) \in V \times V$, find a $\sigma^{(i,j)}$ -integral path flow that satisfies the demand $\gamma^{(i,j)}$ and minimizes the network congestion factor.

We shall establish a solution to Problem σ -IR for the case of a single commodity. To that end, we first show that the optimal network congestion factor is always an element of a predefined finite set.

4.1.1 What are the possible network congestion factors?

In this subsection we observe that the optimal network congestion factor must belong to a set that consist of at most $M \cdot \frac{\gamma}{\sigma}$ elements.

Definition 4.2 Given an instance $\langle G(V, E), \{s, t\}, \{c_e\}, \gamma, \sigma \rangle$ of Problem σ -IR, We say that the given instance has a *feasible solution*, if network $G(V, E)$ consists of a directed path from source s to target t .

We note that a given instance may have a feasible solution even in the case where it is not possible to transfer the entire flow demand without violating the capacities constraints. In that case the optimal network congestion factor will have a value that is *larger than 1*.

Theorem 4.2 Given an instance $\langle G(V, E), \{s, t\}, \{c_e\}, \gamma, \sigma \rangle$ of Problem σ -IR that has a feasible solution. The optimal network congestion factor must be a member in set $\left\{ \frac{i \cdot \sigma}{c_e} \mid e \in E, i \in \left[0, \frac{\gamma}{\sigma} \right], i \in \mathbb{Z} \right\}$.

Proof Let F be the set of all σ -integral flow vectors that transfer γ flow units from s to t , and let $\bar{\alpha}$ be its corresponding set of network congestion factors i.e., $\bar{\alpha} = \left\{ \max_{e \in E} \left\{ \frac{f_e}{c_e} \right\} \mid \{f_e\} \in F \right\}$. Since the given instance has a feasible solution, it follows

that $F \neq \emptyset$, and therefore $\bar{\alpha} \neq \emptyset$. By the definition of $\bar{\alpha}$, it follows that for each $\alpha \in \bar{\alpha}$, there is a flow vector $\{f_e\} \in F$ and a link $e \in E$ such that $f_e = \alpha \cdot c_e$. On the other hand, since all flow vectors in F are σ -integral and transfer at most γ flow units over each link, it follows that for each $\{f_e\} \in F$ and for each $e \in E$, $\frac{f_e}{\sigma} \in \mathbb{Z}$ and $f_e \leq \gamma$. Thus, combining both arguments, it follows that, for each $\alpha \in \bar{\alpha}$ there is a link $e \in E$ such that $\frac{\alpha \cdot c_e}{\sigma} \in \mathbb{Z}$ and $\alpha \cdot c_e \leq \gamma$. In other words, for each $\alpha \in \bar{\alpha}$ there is a link $e \in E$, and an integer $i \in \mathbb{Z}$ such that $\alpha = \frac{i \cdot \sigma}{c_e}$ and $\alpha \leq \frac{\gamma}{c_e}$. Finally, since $\alpha \leq \frac{\gamma}{c_e}$, it follows that $\frac{i \cdot \sigma}{c_e} \leq \frac{\gamma}{c_e}$ and therefore $i \leq \frac{\gamma}{\sigma}$. Thus, we conclude that for each $\alpha \in \bar{\alpha}$ there is a link $e \in E$, and an integer $i \leq \frac{\gamma}{\sigma}$ such that $\alpha = \frac{i \cdot \sigma}{c_e}$. Therefore,

$$\bar{\alpha} = \left\{ \frac{i \cdot \sigma}{c_e} \mid e \in E, i \in \left[0, \frac{\gamma}{\sigma} \right], i \in \mathbb{Z} \right\}.$$

Since the set $\bar{\alpha}$ contains the network congestion factors that corresponds to all σ -integral flow vectors that transfer γ flow units from s to t , it follows that the optimal network congestion factor, denoted by α^* , must also be an element in that set, i.e., $\alpha^* \in \bar{\alpha}$. ■

We henceforth use $\bar{\alpha}$ in order to denote the set $\left\{ \frac{i \cdot \sigma}{c_e} \mid e \in E, i \in \left[0, \frac{\gamma}{\sigma} \right], i \in \mathbb{Z} \right\}$.

Observe that $|\bar{\alpha}| \leq M \cdot \frac{\gamma}{\sigma}$. In addition, note that the set $\bar{\alpha}$ may contain values that are larger than 1. However, since the optimal solution of Problem σ -IR has the minimum network congestion factor, these infeasible values will be considered only when there are no feasible solutions for the given instance.

4.1.2 Procedure Test

Before solving Problem σ -IR, we introduce *Procedure Test* that is specified in Fig. 4.2. This procedure is given an instance $\langle G(V, E), \{s, t\}, \{c_e\}, \gamma, \sigma \rangle$ of Problem σ -IR and a congestion restriction α . If there exists a solution to the given instance such that the network congestion factor is at most α , then the procedure returns it. Otherwise, the procedure returns Fail.

We explain now the main idea behind Procedure Test. Initially, the procedure multiplies all link capacities by a factor of α in order to satisfy the restriction on the network congestion factor. Then it rounds down the capacity of each link to a multiple of σ , and applies any standard Maximum Flow Algorithm over the resulting network. Since all link capacities are σ -integral, the maximum flow algorithm determines a σ -integral link flow that transfers a maximum flow demand. If this link flow transfers from s to t at least γ flow units, then the procedure returns it. Otherwise the procedure fails. The formal description of the procedure is specified in Fig. 4.2.

Procedure Test $(G, \{s, t\}, \{c_e\}, \gamma, \sigma, \alpha)$

Parameters :

- G – network
- (s, t) – commodity
- $\{c_e\}$ – capacities
- γ – demand
- σ – integrality restriction
- α – network congestion factor

1. For each $e \in E$

$$\tilde{c}_e \leftarrow \left\lfloor \frac{\alpha \cdot c_e}{\sigma} \right\rfloor.$$
2. Let $\langle G, (s, t), \{\tilde{c}_e\} \rangle$ be an instance of the Maximum Flow Problem.
Solve this instance using the *push-relabel* method [25]. Denote by $\{f_e\}$ the respective solution, and by F the total transferred flow from s to t .
3. If $F \geq \frac{\gamma}{\sigma}$
Return the link flow $\{\sigma \cdot f_e\}$
Else
Return Fail

Fig. 4.2: Procedure Test

Definition 4.3 We say that Procedure Test *succeeds* whenever it does not return Fail.

Lemma 4.1 If Procedure Test succeeds for an input $\langle G, (s, t), \{c_e\}, \gamma, \sigma, \alpha \rangle$, then the returned link flow $\{f_e\}$ can be decomposed into a σ -integral path flow that transfers at least γ flow units with a network congestion factor of at most α .

Proof Suppose that Procedure Test succeeds with an input $\langle G, (s, t), \{c_e\}, \gamma, \sigma, \alpha \rangle$. Therefore, by construction, the solution $\{f_e\}$ to the instance $\langle G, (s, t), \{\tilde{c}_e\} \rangle$ of the Maximum Flow Problem transfers at least $\frac{\gamma}{\sigma}$ flow units. Thus, it is easy to see that the returned link flow $\{\sigma \cdot f_e\}$, transfers a total flow of at least $\sigma \cdot \frac{\gamma}{\sigma} = \gamma$ units. In addition, since the solution to the instance $\langle G, (s, t), \{\tilde{c}_e\} \rangle$ satisfies the capacity constraint $f_e \leq \tilde{c}_e$, for each link $e \in E$, it follows that $\sigma \cdot f_e \leq \sigma \cdot \tilde{c}_e$ holds for each $e \in E$ and therefore the link flow $\{\sigma \cdot f_e\}$ satisfies, $\sigma \cdot f_e \leq \sigma \cdot \tilde{c}_e = \sigma \cdot \left\lfloor \frac{\alpha \cdot c_e}{\sigma} \right\rfloor \leq \alpha \cdot c_e$, for each link $e \in E$. Hence, the returned link flow transfers at least γ flow units for the instance $\langle G, (s, t), \{c_e\} \rangle$, such that the network congestion factor is at most α . It remains to be shown that link flow $\{\sigma \cdot f_e\}$ can be decomposed into a σ -integral path flow. To that end, note that the capacities of the instance $\langle G, (s, t), \{\tilde{c}_e\} \rangle$ are integral. Therefore, the *Push-Relabel* method returns an *integral* link flow [25]. Since the link flow $\{f_e\}$ has only integral values, the returned link flow $\{\sigma \cdot f_e\}$ is σ -integral. Thus, by employing the Flow Decomposition Algorithm [12], a σ -integral path flow can be decomposed out of the returned σ -integral link flow.

Theorem 4.3 Given are an instance $\langle G, (s, t), \{c_e\}, \gamma, \sigma \rangle$ of Problem σ -IR and an input $\langle G, (s, t), \{c_e\}, \gamma, \sigma, \alpha \rangle$ for Procedure Test. Denote by α^* the optimal network congestion factor corresponding to $\langle G, (s, t), \{c_e\}, \gamma, \sigma \rangle$. Then, Procedure Test succeeds *iff* $\alpha \geq \alpha^*$.

Proof

\Rightarrow : Suppose that Procedure Test succeeds for the input $\langle G, (s, t), \{c_e\}, \gamma, \sigma, \alpha \rangle$. Therefore, by Lemma 4.1, there exists a solution for the instance $\langle G, (s, t), \{c_e\}, \gamma, \sigma \rangle$ of Problem σ -IR, such that the network congestion factor is at most α . Therefore, $\alpha \geq \alpha^*$.

\Leftarrow : Suppose that $\alpha \geq \alpha^*$. Therefore, there exists a solution for the instance $\langle G, (s, t), \{c_e\}, \gamma, \sigma \rangle$ of Problem σ -IR with a network congestion factor of at most α .

In other words, there exists a σ -integral path flow that transfers at least γ units of flow over the instance $\langle G, \{s, t\}, \{\alpha \cdot c_e\} \rangle$ of the Maximum Flow Problem. Since the path flow is σ -integral, we can reduce the capacities $\{\alpha \cdot c_e\}_{e \in E}$ to become multiples of σ without affecting the capability to transfer the flow demand γ . Therefore, it is possible to transfer γ units of flow over the instance $\langle G, (s, t), \left\{ \left\lfloor \frac{\alpha \cdot c_e}{\sigma} \right\rfloor \cdot \sigma \right\} \rangle$ of the Maximum Flow Problem. Hence, it is possible to transfer $\frac{\gamma}{\sigma}$ units of flow over the instance $\langle G, (s, t), \left\{ \left\lfloor \frac{\alpha \cdot c_e}{\sigma} \right\rfloor \right\} \rangle = \langle G, (s, t), \{\tilde{c}_e\} \rangle$ of the Maximum Flow Problem. Thus, by construction, Procedure Test does not return Fail, i.e., it succeeds. ■

4.1.3 Solving Problem σ -IR

We now solve problem σ -IR for the single commodity case. To that end, we introduce *Algorithm σ -IR*, specified in Fig. 4.3. Given an instance $\langle G, \{s, t\}, \{c_e\}, \gamma, \sigma \rangle$ of Problem σ -IR, Algorithm σ -IR finds the smallest $\alpha \in \bar{\alpha}$ (where the set $\bar{\alpha}$ is defined in subsection 4.1.1) such that Procedure Test succeeds for the input $\langle G, \{s, t\}, \{c_e\}, \gamma, \sigma, \alpha \rangle$. Since Theorem 4.2 establishes that the set $\bar{\alpha}$ must contain the optimal network congestion factor, Algorithm σ -IR discovers the optimal network congestion factor. Finally, as Procedure Test succeeds with $\alpha_1 \in \bar{\alpha}$, it must succeed with every $\alpha \geq \alpha_1$, Algorithm σ -IR employs a *binary search* in order to efficiently discover the smallest $\alpha \in \bar{\alpha}$ for which Procedure Test succeeds. Algorithm σ -IR is formally described as follows.

Remark 4.1: a binary search over $\bar{\alpha}$ is possible only if the elements of $\bar{\alpha}$ are *sorted*. However, sorting the set $\bar{\alpha}$ consumes $O(|\bar{\alpha}| \cdot \log|\bar{\alpha}|)$ operations [10], which may be exponential with respect to the input. Nevertheless, for each $e \in E$ the set $\bar{\alpha}_e \triangleq \left\{ i \cdot \frac{\sigma}{c_e} \mid i \in \left[0, \frac{\gamma}{\sigma} \right] \cap \mathbb{Z} \right\}$ is already sorted i.e., the value of the i -th element in the set is $i \cdot \frac{\sigma}{c_e}$. Hence, for each $e \in E$ it is possible to employ a binary search over the set $\bar{\alpha}_e$ in order to find the smallest $\alpha \in \bar{\alpha}_e$ such that procedure Test succeeds within $O\left(\log \frac{\gamma}{\sigma}\right)$ executions of Procedure Test. Since $\bar{\alpha} = \bigcup_{e \in E} \bar{\alpha}_e$, it follows that applying this strategy for each $e \in E$ produces a set of at most M different values that includes the value of the smallest $\alpha \in \bar{\alpha}$ such that Procedure Test succeeds. Therefore, by selecting the smallest value in the resulting set, it is possible to find the smallest $\alpha \in \bar{\alpha}$ such that Procedure Test succeeds within $O\left(M \cdot \log \frac{\gamma}{\sigma}\right)$ executions of Procedure Test.

Algorithm σ -IR $(G, \{s, t\}, \{c_e\}, \gamma, \sigma)$ **Parameters :**

- G – network
- $\{s, t\}$ – commodity
- $\{c_e\}_{e \in E}$ – capacities
- γ – demand
- σ – integrality restriction

Variables :

- α – network congestion factor
- $\{f_e\}_{e \in E}$ – link flow

1. **For each** $e \in E$, Perform a binary search over the set $\overline{\alpha_e}$ in order to find the smallest $\alpha \in \overline{\alpha_e}$ such that **Test** $(G, \{s, t\}, \{c_e\}, \gamma, \sigma, \alpha) \neq \mathbf{FAIL}$.
2. **If** the binary search has failed i.e., for each $e \in E$ there is no $\alpha \in \overline{\alpha_e}$ such that **Test** $(G, \{s, t\}, \{c_e\}, \gamma, \sigma, \alpha) \neq \mathbf{FAIL}$, **Return Fail**.

Else

Denote by α_{\min} the smallest network congestion factor that was output in the search process i.e., $\alpha_{\min} = \min_{e \in E} \{ \min \overline{\alpha_e} \}$.

3. Perform, $\{f_e\} \leftarrow \mathbf{Test}(G, \{s, t\}, \{c_e\}, \gamma, \sigma, \alpha_{\min})$.
4. Execute the *Flow Decomposition Algorithm* [12] over the link flow $\{f_e\}$, in order to obtain a path flow $f : P \rightarrow \mathbb{R}^+ \cup \{0\}$. Let $(p_i)_{i=1}^{\xi}$ be the resulting set of paths that transfer a positive flow.
5. **Return** $(p_i, f(p_i))_{i=1}^{\xi}$.

Fig. 4.3 Algorithm σ -IR

In order to prove that Algorithm σ -IR identifies an optimal solution for Problem σ -IR, we first establish that, if Algorithm σ -IR fails, there is no feasible solution for the given instance. Then we prove that, if it succeeds, it identifies an optimal solution.

Theorem 4.4 Given is an instance $\langle G, \{s, t\}, \{c_e\}, \gamma, \sigma \rangle$ of Problem σ -IR. If Algorithm σ -IR returns Fail, then there is no feasible solution for the given instance.

Proof By construction, if Algorithm σ -IR fails then there is no $\alpha \in \overline{\alpha}$ such that **Test** $(G, \{s, t\}, \{c_e\}, \gamma, \sigma, \alpha) \neq \mathbf{FAIL}$. Thus, according to Theorem 4.3, the optimal network congestion factor, denoted by α^* , is larger than the largest element in set $\overline{\alpha}$.

Thus, $\alpha^* \notin \bar{\alpha}$. Finally, according to Theorem 4.2, it follows that there is no feasible solution for the instance $\langle G, \{s, t\}, \{c_e\}, \gamma, \sigma \rangle$. ■

Theorem 4.5 Given is an instance $\langle G, \{s, t\}, \{c_e\}, \gamma, \sigma \rangle$ of Problem σ -IR. If Algorithm σ -IR does not fail, then it returns a σ -integral path flow that transfers at least γ flow units from s to t , such that the network congestion factor is minimized. Thus, Algorithm σ -IR identifies an optimal solution for Problem σ -IR.

Proof Since Algorithm σ -IR does not fail, it follows by construction that the value α_{\min} , identified in step (2) of the algorithm, satisfies $\mathbf{Test}(G, \{s, t\}, \{c_e\}, \gamma, \sigma, \alpha_{\min}) \neq \text{FAIL}$. Thus, by Lemma 4.1, the returned link flow $\{f_e\}$ can be decomposed into a σ -integral path flow that transfers at least γ flow units, such that the network congestion factor is at most α_{\min} . Finally, this path flow is constructed using the Flow Decomposition Algorithm (step 3).

It remains to be shown that α_{\min} is the optimal network congestion factor for the given instance. However, this is obvious, since α_{\min} is defined to be the smallest $\alpha \in \bar{\alpha}$ such that $\mathbf{Test}(G, \{s, t\}, \{c_e\}, \gamma, \sigma, \alpha) \neq \text{FAIL}$. Therefore, it follows by Theorem 4.3, that α_{\min} equals to the smallest $\alpha \in \bar{\alpha}$ such that $\alpha \geq \alpha^*$, where α^* denotes the optimal network congestion factor for the given instance. However, since we established in Theorem 4.2 that $\alpha^* \in \bar{\alpha}$, it follows that $\alpha_{\min} = \alpha^*$. ■

Theorem 4.6 The complexity of Algorithm σ -IR is $O\left(M \cdot T(M, N) \cdot \log \frac{\gamma}{\sigma}\right)$, where $T(N, M)$ is the running time of the Push-Relabel Algorithm.

Proof It follows from Remark 4.1 that the number of times that Procedure Test is invoked in step 1 of Algorithm σ -IR is $O\left(M \cdot \log \frac{\gamma}{\sigma}\right)$. Therefore, since the most time consuming part of Procedure Test is the execution of the Push-Relabel Algorithm, which consumes $O(T(N, M))$ operations, the complexity of step 1 is $O\left(M \cdot T(M, N) \cdot \log \frac{\gamma}{\sigma}\right)$. Finally, since steps 2-5 have a smaller complexity with respect to step 1, the complexity of Algorithm σ -IR is $O\left(M \cdot T(M, N) \cdot \log \frac{\gamma}{\sigma}\right)$. ■

4.2 Approximation Scheme for Problem KPR

We now employ the solution to Problem σ -IR in order to derive an efficient approximation scheme for Problem KPR. This scheme is based on the following results.

Theorem 4.7 Given are an instance $\langle G(V, E), \{c_e\}, \{\gamma^{(i,j)}\}, \{K^{(i,j)}\} \rangle$ of Problem KPR and a value $r \geq 1$. If the optimal network congestion factor of the given instance is α^* , then, for $\sigma^{(i,j)} = \frac{\gamma^{(i,j)}}{K^{(i,j)} \cdot r}$, the optimal network congestion factor for the instance $\langle G, \{c_e\}, \{\gamma^{(i,j)}\}, \{\sigma^{(i,j)}\} \rangle$ of Problem σ -IR is at most $\left(1 + \frac{1}{r}\right) \cdot \alpha^*$.

Proof Consider the optimal solution of the instance $\langle G(V, E), \{c_e\}, \left\{\left(1 + \frac{1}{r}\right) \cdot \gamma^{(i,j)}\right\}, \{K^{(i,j)}\} \rangle$ of problem KPR. Obviously, the optimal network congestion factor is $\left(1 + \frac{1}{r}\right) \alpha^*$. Round down the path flow of the considered optimal solution to be a multiple of $\frac{\gamma^{(i,j)}}{r \cdot K^{(i,j)}}$ for each path $p \in P^{(i,j)}$. Therefore, the resulting path flow is $\frac{\gamma^{(i,j)}}{r \cdot K^{(i,j)}}$ -integral for each $p \in P^{(i,j)}$. In this process, the flow over each path is reduced by at most $\frac{\gamma^{(i,j)}}{r \cdot K^{(i,j)}}$ flow units. Since there are no more than $K^{(i,j)}$ paths, the demand of each commodity is reduced by at most $\frac{\gamma^{(i,j)}}{r}$ flow units. Since initially each commodity transferred $\left(1 + \frac{1}{r}\right) \cdot \gamma^{(i,j)}$ flow units, it follows that, after the rounding, each commodity transfers at least $\gamma^{(i,j)}$ flow units.

Thus, we identified a path flow that satisfies the instance $\langle G, \{c_e\}, \{\gamma^{(i,j)}\}, \{\sigma^{(i,j)}\} \rangle$ of Problem σ -IR, where $\sigma^{(i,j)} = \frac{\gamma^{(i,j)}}{K^{(i,j)} \cdot r}$ for each $(i, j) \in V \times V$. In addition, since in this process we only reduce flow, the network congestion factor is at most $\left(1 + \frac{1}{r}\right) \alpha^*$. ■

Theorem 4.8 Given an instance $\langle G, \{c_e\}, \{\gamma^{(i,j)}\}, \{\sigma^{(i,j)}\} \rangle$ of Problem σ -IR, for each commodity $(i, j) \in V \times V$, the total number of paths $p \in P^{(i,j)}$ that transfer a positive amount of flow is at most $\left\lceil \frac{\gamma^{(i,j)}}{\sigma^{(i,j)}} \right\rceil$.

Proof Since, for each commodity $(i, j) \in V \times V$, the flow is $\sigma^{(i,j)}$ -integral, each path $p \in P^{(i,j)}$ that transfers a positive amount of flow must transfer at least $\sigma^{(i,j)}$ flow units. Therefore, for each commodity $(i, j) \in V \times V$, there are no more than $\left\lceil \frac{\gamma^{(i,j)}}{\sigma^{(i,j)}} \right\rceil$ different paths. ■

We are now ready to define an approximation scheme for Problem KPR, and evaluate its performance. To that end, we employ Algorithm σ -IR that solves Problem σ -IR for the single commodity case.

KPR Approximation Scheme Given a single commodity instance $\langle G, \{s, t\}, \{c_e\}, \gamma, K \rangle$ of Problem KPR and a real number $r \geq 1$, define a single commodity instance $\langle G, \{s, t\}, \{c_e\}, \gamma, \sigma \rangle$ of Problem σ -IR, with $\sigma = \frac{\gamma}{K \cdot r}$, and solve it through Algorithm σ -IR.

Given an instance of Problem KPR, it follows from Theorem 4.7 that the *KPR Approximation Scheme* returns a solution with a network congestion factor that is larger by a factor of at most $\left(1 + \frac{1}{r}\right)$ than the optimal network congestion factor. In addition, by Theorem 4.8, it follows that the returned solution consists of at most

$\left\lceil \frac{\gamma}{\sigma} \right\rceil = \left\lceil \frac{\gamma}{\frac{\gamma}{K \cdot r}} \right\rceil = \lceil K \cdot r \rceil$ different paths with positive flow. We summarize the above discussion as follows.

Corollary 4.1 Given are a single commodity instance $\langle G, (s, t), \{c_e\}, \gamma, K \rangle$ of Problem KPR and a real number $r \geq 1$. The KPR Approximation Scheme produces at most $\lceil K \cdot r \rceil$ paths with a network congestion factor that is at most $\left(1 + \frac{1}{r}\right)$ times larger than the optimal network congestion factor of the given instance. ■

Since we proved that the complexity of Algorithm σ -IR is $O\left(M \cdot T(M, N) \cdot \log \frac{\gamma}{\sigma}\right)$, and since $\sigma = \frac{\gamma}{K \cdot r}$, it follows that the complexity of the KPR Approximation scheme is $O\left(M \cdot T(M, N) \cdot \log(K \cdot r)\right) = O\left(M \cdot T(N, M) \cdot (\log(K) + \log(r))\right)$, i.e., polynomial in the input size.

Remark 4.2 Note that for $r=1$, the KPR Approximation Scheme is a 2-approximation scheme i.e., it provides a solution that satisfies the restrictions on the number of paths while producing a network congestion factor at most twice the optimum; the complexity of the algorithm for this case is $O(M \cdot \log(K) \cdot T(N, M))$. Since $K \leq M$ and since a single max flow computation can be obtained within $O(M \cdot N \cdot \log N)$ operations [25], it follows that the running time of the algorithm for $r=1$ is $O(M^2 \cdot N \cdot \log^2 N)$.

Since $r=1$ is an important special case where the restriction on the number of paths is not violated, in the following we show how to decrease the computational complexity of the KPR Approximation Scheme for $r=1$ into $O(M \cdot \log N \cdot (M + N \cdot \log N))$.

Consider Algorithm σ -IR that is specified in Fig. 4.3. In step 1 of the algorithm a binary search is performed over the set $\bar{\alpha}_e$ for each $e \in E$ in order to find the smallest $\alpha \in \bar{\alpha}_e$ such that procedure Test succeeds. Then, among the corresponding set of at most M different values, the smallest value is selected (step 2); thus, resulting with the smallest value $\alpha \in \bar{\alpha}$ such that Procedure Test succeeds. The improvement for the case of $r=1$ is obtained by first sorting *all* the elements of $\bar{\alpha}$ and then invoking a binary search in order to find the smallest $\alpha \in \bar{\alpha}$ such that Procedure Test succeeds.

Consider the computational complexity of the KPR Approximation Scheme when the proposed improvement is attained. For $r=1$ the KPR Approximation Scheme defines an instance $\langle G, \{s, t\}, \{c_e\}, \gamma, \sigma \rangle$ of Problem σ -IR, with $\sigma = \frac{\gamma}{K}$; then it solves the corresponding instance through the modified version of Algorithm σ -IR (i.e., the version that first sorts the elements of $\bar{\alpha}$ and then invoke the binary search). Since $\sigma = \frac{\gamma}{K}$ for $r=1$, it is easy to see from Theorem 4.2 that the set $\bar{\alpha}$ consists of $K \cdot M$ elements (for the corresponding case of $r=1$). We use this observation in order to establish the computational complexity of the modified version of Algorithm σ -IR as follows.

The corresponding computational complexity of the modified version of Algorithm σ -IR is determined by the computational complexity of the process that sorts the elements of $\bar{\alpha}$, and the process that executes the binary search in order to find the smallest $\alpha \in \bar{\alpha}$ such that Procedure Test succeeds. Sorting all the elements of $\bar{\alpha}$ requires $|\bar{\alpha}| \cdot \log |\bar{\alpha}| = O(K \cdot M \cdot \log(K \cdot M))$ operations [10]. Therefore, Since $K \leq M$, the sorting process requires $O(K \cdot M \cdot \log(K \cdot M)) \leq O(M^2 \cdot \log M^2) \leq O(M^2 \cdot \log(N^2)) = O(M^2 \cdot \log N)$ operations.

Next, invoking a binary search so as to find the smallest $\alpha \in \bar{\alpha}$ such that Procedure Test succeeds requires $O(\log |\bar{\alpha}|) \leq O(\log M^2) \leq O(\log(N^2)) = O(\log N)$ executions of Procedure Test. Since the most time consuming part of Procedure Test is the execution of the max flow computation, which can be obtained within

$O(M \cdot N \cdot \log N)$ operations [25], the binary search over the sorted set $\bar{\alpha}$ consumes $O(M \cdot N \cdot \log^2 N)$ operations. Thus, the total complexity of the modified version of Algorithm σ -IR (hence, the KPR Approximation Scheme) for the case of $r=1$ is $O(M^2 \cdot \log N + M \cdot N \cdot \log^2 N) = O(M \cdot \log N (M + N \cdot \log N))$.

4.3 Extensions to Problem KPR

In this section we briefly describe some additional results that are related to Problem KPR. More specifically, one result deals with the dual problem, which restricts the network congestion factor and minimizes the number of paths for routing and another result adds a cost restriction.

4.3.1. Problem MPR

Problem MPR is defined as follows:

Problem MPR (Minimum Paths for Routing) Given are a network $G(V, E)$, two nodes $s, t \in V$, A capacity $c_e > 0$ for each link $e \in E$, a demand $\gamma > 0$ and a restriction on the network congestion factor α . Find a path flow with a network congestion factor of at most α , such that, if $P \subseteq P^{(s,t)}$ is the set of all paths in $P^{(s,t)}$ that are assigned with a positive flow, then $|P|$ is minimized.

A solution to Problem MPR can be derived by employing the KPR Approximation Scheme (specified in section 4.2) as follows.

Given an instance of Problem MPR, denote by K^* the number of different paths that carry a positive flow in the optimal solution. Observe that K^* must take values only in the range $[1, M]$ (since every link flow can be decomposed into the sum of at most M paths and cycles that carry a positive flow). Therefore, given an instance $\langle G, (s, t), \{c_e\}, \gamma, \alpha \rangle$ of Problem MPR a real number $r \geq 1$, one can define, for each $K \in [1, M]$, an instance $\langle G, (s, t), \{c_e\}, \gamma, K \rangle$ of Problem KPR, and solve the resulting instance using the KPR Approximation Scheme for the given r . If the resulting network congestion factor is larger than $\left(1 + \frac{1}{r}\right) \cdot \alpha$, a smaller value of K shall be chosen, otherwise a larger one shall be chosen. We terminate when we get the smallest K such that the network congestion factor is at most $\left(1 + \frac{1}{r}\right) \cdot \alpha$. This approximation scheme shall be termed as the MPR Approximation Scheme.

Theorem 4.9 Given an instance $\langle G, (s, t), \{c_e\}, \gamma, \alpha \rangle$ of Problem MPR, the MPR Approximation Scheme finds a path flow with at most K^* paths that transfer at least γ units of flow, such that the network congestion factor is at most $\left(1 + \frac{1}{r}\right) \cdot \alpha$.

Proof By construction, the output of the MPR Approximation Scheme is a path flow that transfers at least γ flow units such that the network congestion factor is at most $\left(1 + \frac{1}{r}\right) \cdot \alpha$. Therefore, we only need to prove that the number of defined paths is indeed K^* . Denote by K^{**} the number of different paths that the MPR Approximation Scheme calculated for the instance $\langle G, (s, t), \{c_e\}, \gamma, \alpha \rangle$. We have to prove that $K^* \geq K^{**}$. By construction, K^{**} is the smallest $K \in [1, M]$ such that the optimal network congestion factor for the instance $\langle G, (s, t), \{c_e\}, \gamma, K \rangle$ of Problem KPR is smaller than $\left(1 + \frac{1}{r}\right) \cdot \alpha$. Therefore, for the instance $\langle G, (s, t), \{c_e\}, \gamma, K^{**} - 1 \rangle$, the KPR Approximation Scheme returns a network congestion factor that is larger than $\left(1 + \frac{1}{r}\right) \cdot \alpha$. Since, for a given instance $\langle G, (s, t), \{c_e\}, \gamma, K \rangle$ of Problem KPR and a real number $r \geq 1$, the KPR Approximation Scheme finds a path flow with a network congestion factor that is at most $\left(1 + \frac{1}{r}\right)$ times larger than the optimum, the optimal network congestion factor for the instance $\langle G, (s, t), \{c_e\}, \gamma, K^{**} - 1 \rangle$ of Problem KPR is larger than α . Therefore, since the optimal solution for the given instance of Problem MPR produces a network congestion factor of at most α we conclude that $K^* > K^{**} - 1$, hence $K^* \geq K^{**}$. ■

We note that, by employing Procedure Test (introduced in section 4.1) directly, a *better* time complexity approximation scheme can be derived. We omit the detailed description of this approximation scheme, and only note that, instead of solving an instance $\langle G, (s, t), \{c_e\}, \gamma, K \rangle$ of Problem KPR for each $K \in [1, M]$, we execute Procedure Test directly, for each $K \in [1, M]$, with an input $\left\langle G, (s, t), \{c_e\}, \gamma, \sigma, \left(1 + \frac{1}{r}\right) \cdot \alpha \right\rangle$ such that $\sigma = \frac{\gamma}{K \cdot r}$.

4.3.2 Problem BCKPR

Another restriction that can be added to Problem KPR is a "cost", as follows.

Problem BCKPR (Bounded Cost K-Path Routing) Given are a network $G(V, E)$, for each link $e \in E$, a capacity $c_e > 0$ and a weight (i.e., "cost") $w_e > 0$, and, for each commodity $(i, j) \in V \times V$, a demand $\gamma^{(i, j)}$, a split restriction $K^{(i, j)}$ and a weight ("cost") restriction $C^{(i, j)}$. Find a path flow that minimizes the network congestion factor, subject to the following:

$$\blacksquare \sum_{e \in E} w_e \cdot f_e^{(i, j)} \leq C^{(i, j)}.$$

- If $P_x^{(i,j)} \subseteq P^{(i,j)}$ is the collection of all paths in $P^{(i,j)}$ that are assigned with positive flow, then $|P_x^{(i,j)}| \leq K^{(i,j)}$.

An approximation scheme for Problem BCKPR for the single commodity case can be derived by substituting the Maximum Flow algorithm in Procedure Test with a Minimum Cost Flow algorithm [12]. Then, if $\sum_{e \in E} w_e \cdot f_e \leq C$, Procedure Test will return Fail and a larger network congestion factor will be chosen by Algorithm σ -IR. We omit the precise description of the approximation scheme and its proof.

5. Simulation Results

In this section, we present a comparison between an optimal solution to multipath routing and that provided by a heuristic scheme such as the (popular) Equal Cost MultiPath (ECMP) routing scheme.

Following the lines of [44], we generated 10,000 random topologies as follows. First, for each topology we located the source and the destination at the diagonally opposite corners of a square area of unit dimension. Then, we randomly spread 18 nodes over the square. Finally, we introduced a link between each two nodes u and v , with the following probability, which depended on the distance between them, $\delta(u,v)$:

$$p(u,v) = \alpha \cdot \exp\left[\frac{-\delta(u,v)}{\beta \cdot \sqrt{2}}\right],$$

using $\alpha=5$ and $\beta=0.3$. In all runs, we assumed that (a) the link capacities are distributed uniformly in [1,25] MB/sec (b) the link weights are distributed uniformly in [1,4].

For each topology, we conducted the following measurements: (a) we measured the network congestion factor produced by invoking ECMP; (b) we measured the network congestion factor produced by an optimal assignment of traffic to shortest paths and to paths with a weight that is equal to $1.17 \cdot W^*$, $1.33 \cdot W^*$, $1.5 \cdot W^*$, $1.67 \cdot W^*$, $1.83 \cdot W^*$, $2 \cdot W^*$ and $2.17 \cdot W^*$, where W^* is the weight of a shortest path. Our results are summarized in Fig. 5.1. Note that if the ECMP scheme had an optimal traffic distribution mechanism, the network congestion factor could be reduced by a *factor of 3*. Moreover by relaxing the requirement to route along shortest paths by 33%, the network congestion factor is *10 times smaller* than with the standard ECMP. Thus, by employing Algorithm RMP or its ε -optimal approximation with $W \approx 1.33 \cdot W^*$ congestion can be reduced by a factor of 10 with respect to that produced by ECMP.

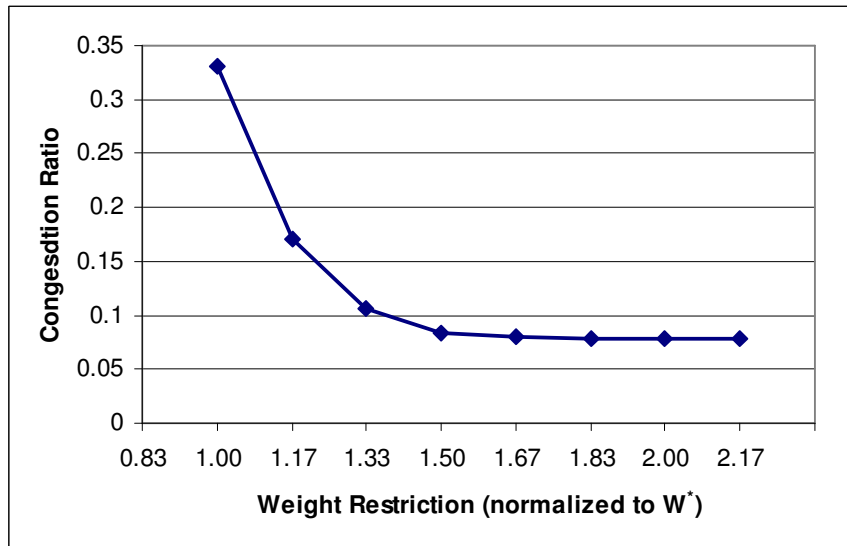


Fig. 5.1. The ratio between the network congestion produced by an optimal multipath routing assignment (for several end-to-end weight restrictions) and the network congestion produced by ECMP.

6. Future research

During this work we observed that multipath routing offers many advantages in contexts that are not necessarily related to congestion avoidance or load balancing. In the following we present a brief description of some ideas for future research.

Multipath routing and survivability

Multipath routing can be used in order to improve resilience and avoid congestion. The combination of both benefits can be obtained by employing the idea of *diversity coding* [3], which adds redundant information to the data stream, like error detection and correction codes. Then, in order to increase fault tolerance, the redundant information is routed along paths that are disjoint to the paths that are used to transfer the original data stream. Therefore, it is desired to develop new multipath routing schemes that also engage the diversity coding concept. For example, it is desired to develop schemes for multipath routing that maximize the total flow (or minimize the congestion) and satisfy a fundamental property that restricts each path that transfer positive data flow to have an adequate set of disjoint paths with enough bandwidth to protect this flow.

Multipath routing and security

Splitting the traffic among several node-disjoint paths has the advantage that no node (except the source and the target) gets to see the whole data stream. Therefore, in practice, reconstructing the data stream is possible only at the target node. This inherent advantage of multipath routing can be used in order to enhance security.

In order to exploit this idea, we should identify several node-disjoint paths, and transfer a *limited* portion of the total flow demand over each. This limitation is a key point, since otherwise a single path could transfer about all of the flow.

Accordingly, it is desirable to incorporate this notion into multipath routing schemes in order to satisfy security requirements.

Energy efficiency and multipath routing

In wireless networks, where nodes have limited power resources (batteries), energy conserving routing must be employed in order to maximize the network lifetime, i.e., the time until the first battery drains-out [29]. Since energy consumption is proportional to a node's transmission rate, it is easy to see that splitting the traffic among several paths can prolong the network lifetime, i.e., the time until the first battery drains out. Therefore, it is of interest to design and explore practical multipath routing schemes that incorporate energy conservation policies for such environments.

Recovery schemes for multipath routing

Multipath routing can provide additional benefits in networks where resource reservation must be made before data can be sent along a route (e.g. ATM). Consider for example a routing scheme that uses multipath routing in order to reduce congestion. Then, in case of a failure in one of the paths, we may split the data stream

that was traveling over the failed path among the remaining paths, *without* any additional path computation or resource reservation. This fast recovery property can be obtained in several ways. For example, it may be employed by a restriction that, upon a path failure, the sum of the *spare* capacities of the remaining paths is not smaller than the flow that was traveling over the failed path.

Distributed multipath routing schemes

So far, we considered multipath routing schemes that employed source routing; hence, all routing decisions were determined at the source node. Although source routing is simple and intuitive, there are many cases that the use of hop-by-hop routing is preferable. Hence, investigation of distributed multipath routing schemes is called for.

Designing practical distributed multipath routing schemes raises several challenges. One major challenge is to limit the buffer space usage at each node in the network. In other words, if $h_v(i)$ denotes the number of next hops that are allocated for a target i in the routing table of a node v , a distributed multipath routing scheme needs to cope with the restriction that $\sum_{i \in V} h_v(i)$ cannot exceed the available buffer space at node $v \in V$.

Dynamic multipath routing schemes

So far, we have investigated multipath routing schemes where all demands are known in advance. However, in practice, demands may arrive one at a time without any a-priory knowledge regarding future demands. Therefore, it is important to investigate on-line multipath routing schemes that don't rely on such a-priory knowledge.

We note that we are currently working on this issue and have obtained some important results.

Appendix

This Appendix contains the proofs of Theorem 2.1.

Theorem 2.1 Given a network $G(V,E)$, capacities $\{c_e\}_{e \in E}$ and a pair of nodes s,t , denote by F the maximum net flow that can be transferred from s to t . Then, for every $\gamma \leq F$, the link flow $\{f_e\}_{e \in E}$ is a maximum flow *iff* the link flow $\{\frac{\gamma}{F} \cdot f_e\}_{e \in E}$ transfers γ flow units from s to t and is such that the network congestion factor is minimized.

Proof: \Rightarrow : It is easy to see that, since $\{f_e\}_{e \in E}$ is a vector that satisfies the capacity and the flow conservation constraints, the vector $\{\frac{\gamma}{F} \cdot f_e\}_{e \in E}$ must also satisfy these constraints. In addition, since $\{f_e\}$ transfers F flow units from s to t (i.e., $\sum_{e \in O(s)} f_e - \sum_{e \in I(s)} f_e = F$), it holds that

$\sum_{e \in O(s)} \frac{\gamma}{F} \cdot f_e - \sum_{e \in I(s)} \frac{\gamma}{F} \cdot f_e = \frac{\gamma}{F} \cdot (\sum_{e \in O(s)} f_e - \sum_{e \in I(s)} f_e) = \frac{\gamma}{F} \cdot F = \gamma$. It remains to be shown that the network congestion factor $\alpha \triangleq \max_{e \in E} \left\{ \frac{\frac{\gamma}{F} \cdot f_e}{c_e} \right\}$ is minimal among all link flows that transfer

γ flow units from s to t . Since $\{f_e\}_{e \in E}$ is a maximum flow, there must be a link $e \in E$ that satisfies $f_e = c_e$; hence, $\alpha = \frac{\gamma}{F} \cdot \max_{e \in E} \left\{ \frac{f_e}{c_e} \right\} = \frac{\gamma}{F}$; therefore, we have to show that the minimum network congestion factor equals $\frac{\gamma}{F}$.

By way of contradiction, suppose that there exists a link flow $\{\tilde{f}_e\}_{e \in E}$ that transfers γ flow units from s to t such that $\tilde{\alpha} = \max_{e \in E} \left\{ \frac{\tilde{f}_e}{c_e} \right\} < \frac{\gamma}{F}$. Consider the link flow $\left\{ \frac{1}{\tilde{\alpha}} \cdot \tilde{f}_e \right\}_{e \in E}$. Since $\tilde{\alpha} = \max_{e \in E} \left\{ \frac{\tilde{f}_e}{c_e} \right\}$, it holds that

$\max_{e \in E} \left\{ \frac{\frac{1}{\tilde{\alpha}} \cdot \tilde{f}_e}{c_e} \right\} = \frac{1}{\tilde{\alpha}} \cdot \max_{e \in E} \left\{ \frac{\tilde{f}_e}{c_e} \right\} = 1$; hence the capacity constraints are satisfied for

$\left\{ \frac{1}{\tilde{\alpha}} \cdot \tilde{f}_e \right\}_{e \in E}$. Moreover, it is given that $\sum_{e \in O(s)} \tilde{f}_e - \sum_{e \in I(s)} \tilde{f}_e = \gamma$; hence,

$\sum_{e \in O(s)} \frac{1}{\tilde{\alpha}} \cdot \tilde{f}_e - \sum_{e \in I(s)} \frac{1}{\tilde{\alpha}} \cdot \tilde{f}_e = \frac{1}{\tilde{\alpha}} \cdot (\sum_{e \in O(s)} \tilde{f}_e - \sum_{e \in I(s)} \tilde{f}_e) = \frac{\gamma}{\tilde{\alpha}} > \frac{\gamma}{\gamma/F} = F$. Thus, link flow $\left\{ \frac{1}{\tilde{\alpha}} \cdot \tilde{f}_e \right\}_{e \in E}$

transfers more than F flow units from s to t without violating the capacity constraints. However, this contradicts the fact that F is the maximum net flow that can be carried between s and t .

\Leftarrow : Suppose that $\{\frac{\gamma}{F} \cdot f_e\}_{e \in E}$ is a link flow that transfers γ flow units from s to t such that the network congestion factor is minimized. We have to show that $\{f_e\}_{e \in E}$ is a max-flow i.e., it is a link flow that transfers F flow units from s to t . Since $\{\frac{\gamma}{F} \cdot f_e\}_{e \in E}$ satisfies the flow conservation constraints it follows that $\{f_e\}_{e \in E}$ also satisfies the flow conservation constraints. Moreover, since $\{\frac{\gamma}{F} \cdot f_e\}_{e \in E}$ transfers

γ flow units from s to t , $\sum_{e \in O(s)} f_e - \sum_{e \in I(s)} f_e = \frac{F}{\gamma} \cdot (\sum_{e \in O(s)} \frac{\gamma}{F} \cdot f_e - \sum_{e \in I(s)} \frac{\gamma}{F} \cdot f_e) = \frac{F}{\gamma} \cdot \gamma = F$ i.e.,

$\{f_e\}_{e \in E}$ transfers F flow units from s to t . It is left to be shown that $\{f_e\}_{e \in E}$ satisfies the capacity constraints. To that end, it is sufficient to show that the network congestion factor of $\{f_e\}_{e \in E}$ is at most 1. Thus, it is sufficient to show that the network congestion factor of $\{\frac{\gamma}{F} \cdot f_e\}_{e \in E}$ is at most $\frac{\gamma}{F}$. Hence, since we assume that $\{\frac{\gamma}{F} \cdot f_e\}_{e \in E}$ has the minimum network congestion factor, we only have to show the existence of some link flow that transfers γ units from s to t and has a network congestion factor of at most $\frac{\gamma}{F}$. In the following, we construct such a flow. More specifically, we show that, if $\{\tilde{f}_e\}_{e \in E}$ is a maximum link flow from s to t , then $\{\frac{\gamma}{F} \cdot \tilde{f}_e\}_{e \in E}$ is a link flow that transfers γ units from s to t and has a network congestion factor of at most $\frac{\gamma}{F}$.

Considering the maximum flow $\{\tilde{f}_e\}_{e \in E}$, we first show that $\{\frac{\gamma}{F} \cdot \tilde{f}_e\}_{e \in E}$ has a network congestion factor of at most $\frac{\gamma}{F}$. Since $\{\tilde{f}_e\}_{e \in E}$ must satisfy the capacity constraints, it holds that $\tilde{f}_e \leq c_e$ for each $e \in E$; hence, $\frac{\tilde{f}_e}{c_e} \leq 1$ for each $e \in E$ and, in particular, $\max_{e \in E} \left\{ \frac{\tilde{f}_e}{c_e} \right\} \leq 1$. Thus, the network congestion factor of $\{\frac{\gamma}{F} \cdot \tilde{f}_e\}_{e \in E}$ is $\max_{e \in E} \left\{ \frac{\frac{\gamma}{F} \cdot \tilde{f}_e}{c_e} \right\} = \frac{\gamma}{F} \cdot \max_{e \in E} \left\{ \frac{\tilde{f}_e}{c_e} \right\} \leq \frac{\gamma}{F}$. We turn to show that $\{\frac{\gamma}{F} \cdot \tilde{f}_e\}_{e \in E}$ assigns γ flow units to paths from s to t . To that end, it is sufficient to show that the total net flow out of the source s for $\{\frac{\gamma}{F} \cdot \tilde{f}_e\}_{e \in E}$ is γ . Moreover, since the total net flow out of s for $\{\tilde{f}_e\}_{e \in E}$ is F , it holds that $\sum_{e \in O(s)} \frac{\gamma}{F} \cdot \tilde{f}_e - \sum_{e \in I(s)} \frac{\gamma}{F} \cdot \tilde{f}_e = \frac{\gamma}{F} \cdot (\sum_{e \in O(s)} \tilde{f}_e - \sum_{e \in I(s)} \tilde{f}_e) = \frac{\gamma}{F} \cdot F = \gamma$ i.e., $\{\frac{\gamma}{F} \cdot \tilde{f}_e\}_{e \in E}$ assigns γ flow units to paths from s to t . Finally, it is easy to see that since the conservation constraints are satisfied for $\{\tilde{f}_e\}_{e \in E}$ they are also satisfied for $\{\frac{\gamma}{F} \cdot \tilde{f}_e\}_{e \in E}$; moreover, since we assume that $\gamma \leq F$ and we have shown that the network congestion factor of $\{\frac{\gamma}{F} \cdot \tilde{f}_e\}_{e \in E}$ is at most $\frac{\gamma}{F}$, it holds that $\{\frac{\gamma}{F} \cdot \tilde{f}_e\}_{e \in E}$ satisfies the capacity constraints. ■

References

- [1] S. Iyer, S. Bhattacharyya, N. Taft, N. McKeon, C. Diot, "A measurement Based Study of Load Balancing in an IP Backbone", Sprint ATL Technical Report, TR02-ATL-051027, May 2002.
- [2] D. Bertsekas and R. Gallager, "Data networks", Prentice-Hall, 1992.
- [3] A. Tsirigos and Z. J. Haas, "Multipath Routing in Mobile Ad Hoc Networks or How to Route in the Presence of Topological Changes", In Proceedings of IEEE MILCOM 2001, pages 878-883, October 2001.
- [4] M. Kodialam and T. V. Lakshman, "Restorable Dynamic Quality of Service Routing", IEEE Communications Magazine, vol. 40, no. 6, June 2002.
- [5] D. Thaler and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", IETF RFC 2991, 2000.
- [6] C. Villamizar, "OSPF Optimised Multipath (OSPF-OMP)", Internet Draft <draft-ietf-ospf-omp-02.txt>, 1998.
- [7] S. Nelakuditi and Zhi-Li Zhang, "On Selection of Paths for Multipath Routing", In Proc. IWQoS, 2001, Karlsruhe, Germany.
- [8] I. Cidon, R. Rom and Y. Shavitt, "Analysis of Multi-Path Routing", IEEE Transactions on Networking, 7:885-896, 1999.
- [9] W. Lai, Ed. and D. McDysan, Ed., "Network Hierarchy and Multilayer Survivability", IETF RFC 3386, November 2002.
- [10] T. Cormen, C. Leiserson, and R. Rivest, Introduction to Algorithms. Cambridge, MA: The MIT Press, 2001.
- [11] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin and O. Waarts, "On-Line Routing of Virtual Circuits with Applications to Load Balancing and Machine Scheduling", Journal of the ACM, vol. 44, no. 3, pages 486-504, May 1997.
- [12] R. K. Ahuja, T. L. Magnanti and J. B. Orlin, "Network Flows: Theory, Algorithm, and Applications", Prentice Hall, 1993.
- [13] D.D. Sleator and R.E. Tarjan, "Amortized efficiency of list update and paging rules", Communications of the ACM, vol. 28, no. 2, pages 202-208, February 1985.
- [14] J.Moy, "OSPF Version 2", IETF RFC 2328, April 1998.
- [15] Y. Wang and Z. Wang, "Explicit Routing Algorithms For Internet Traffic Engineering", In Proceedings of ICCN'99, Boston, October 1999.

- [16] Yanxia Jia, Ioanis Nikolaidis and P. Gburzynski, "Multiple Path QoS Routing", Proceedings of ICC'01 Finland, pages 2583-2587, June 2001.
- [17] S. Chen and K. Nahrstedt, "An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions", IEEE network Magazine, Special Issue on Transmission and Distribution of Digital Video, vol. 12, num. 6, pages 64-79, December 1998.
- [18] J.Kleinberg, "Single-Source Unsplittable Flow", in proceeding of the 37th annual symposium on foundations of computer science, October 1996 pages 23-25.
- [19] Y. Dinitz, N. Garg and M. X. Goemans, "On the Single –Source Unsplittable Flow Problem", Combinatorica , 19, pages 68-77.
- [20] F. Shahrokhi and D. Matula, "The Maximum Concurrent Flow Problem" Journal of ACM, vol. 37, no. 2, pages 318-334, 1990.
- [21] O. Hauser, M. Kodialam and T.V. Lakshman "Capacity Design of Fast Path Restorable Optical Networks“, IEEE/ACM Transactions on Networking, October 2000.
- [22] A. Pitsillides, S. Nikolopoulos and D. Tipper, "Addressing Network Survivability Issues by Finding the K-best Paths Through a Trellis Graph", Proceedings of IEEE INFOCOM '97, Kobe, Japan, April 1997.
- [23] M. R. Garey and D. S. Johnson, "Computers and Intractability", W.H. Freeman and Co., 1979.
- [24] N. Karmarkar, "A New Polynomial-Time Algorithm For Linear Programming", Combinatorica, vol. 4, pages 373-395, 1984.
- [25] A. V. Goldberg and R. E. Tarjan, "A New Approach to The Maximum Flow Problem", Journal of ACM, vol.35, no. 4, pages 921-940, 1988.
- [26] A. Bley, "On the Complexity of Vertex-Disjoint Length-Restricted Path Problems", Konrad-Zuse-Zentrum fur Informationstechnik Berlin Tech, Report SC-98-20, 1998.
- [27] G. Ausiello, P.Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela and M. Protasi, "Complexity and Approximation: Combinatorial Optimization Problems and Their Approximation Properties", Springer-Verlag, 1999.
- [28] E. Comer, "Internetworking with TCP/IP Volume I: Principles, Protocols and Architecture", Prentice Hall, 3rd edition, 1995.
- [29] G. Zussman and A. Segall, "Energy Efficient Routing in Ad Hoc Disaster Recovery Networks", In Proceedings of IEEE INFOCOM 2003, April 2003.

- [30] D. Awduche, J. Malcolm, M. O'Dell, and J. McManus, "Requirements for traffic engineering over MPLS" , Internet Draft <draft-awduche-mpls-traffic-eng-00.txt>, April, 1998.
- [31] A. E. I. Widjaja, "Mate: MPLS Adaptive Traffic Engineering", Internet Draft <draft-widjaja-mpls-mate-00.txt>, August, 1998.
- [32] S. Plotkin, "Competitive Routing of Virtual Circuits in ATM Networks", IEEE J. Selected Areas in Communications, vol. 13, pages 1128-1136, August 1995.
- [33] J.M. Akinpelu, "The Overload Performance of Engineered Networks With Non-Hierarchical and Hierarchical Routing", AT&T Bell Laboratories Technical Journal, vol. 63, no. 7, September 1984.
- [34] G.R. Ash, "Dynamic Network Evolution, with Examples from AT&T's Evolving Dynamic Network", IEEE Communications Magazine, pages 26-39, July 1995.
- [35] B.R. Hurley, C.J.R. Seidl and W.F. Sewell, "A Survey of Dynamic Routing Methods for Circuit-Switched Traffic", IEEE Communications Magazine, vol. 25, no. 9, September 1987.
- [36] Y. Azar, B. Kalyanasundaram, S. Plotkin, K. Pruhs, and O. Waarts, "On-line Load Balancing of Temporary Tasks", In Proceedings of Workshop on Algorithms and Data Structures, pages 119-130, August 1993.
- [37] R. Ogier, B. Bellur, and N. Taft-Plotkin, "An Efficient Algorithm for Computing Shortest and Widest Maximally Disjoint Paths", SRI International Technical Report ITAD-1616-TR-170, November 1998.
- [38] N. Taft-Plotkin, B. Bellur, and R. Ogier, "Quality-of -Service Routing Using Maximally Disjoint Paths", In Proceedings of IEEE IWQoS99, pages 119-128, London, UK, January 1999.
- [39] M. Kodialam and T. V. Lakshman, "Restorable Dynamic Quality of Service Routing", IEEE Communication Magazine, vol. 40, no. 6, June 2002.
- [40] A. Ouorou, P. Mahey, and J.-Ph. Vial, "A Survey of Algorithms For Convex Multicommodity Flow Problems", Management Science, vol. 46, pages 126-147, January 2000.
- [41] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell and J. McManus, "Requirements for Traffic Engineering Over MPLS", IETF RFC 2702, September 1999.
- [42] J. Semke, J. Mahdavi, and M. Mathis, "Automatic TCP buffer tuning", In Proc. ACM SIGCOMM'98 Conference, 1998.
- [43] V. Paxson, "End-to-End Routing Behavior in the Internet", in proc. ACM SIGCOM, 1996.

- [44] B. M. Waxman, *Routing of Multipoint Connections*, IEEE Journal on Selected Areas in Communications, 6:1617-1622,1988.
- [45] E. Rosen, A. Viswanathan, and R. Callon. "Multiprotocol Label Switching Architecture". IETF RFC 3031, 2001.