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 (or near 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. Finally, we establish efficient solutions with proven performance guarantees.*

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 [14]. Moreover, congested links usually result in poor performance and high variance. For such circumstances, multipath routing can offer steady and smooth data streams [6].

Previous studies and proposals on multipath routing have focused on *heuristic methods*. In [18], 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 [23] 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 [24] and [26] 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 [19], a scheme was 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.

Through comprehensive simulations we show that multipath solutions obtained by *optimal* congestion reduction schemes are *fundamentally* more efficient than solutions obtained by heuristics. Specifically, we show that if the traffic distribution mechanism of the ECMP or OSPF-OMP schemes had been optimal, the network congestion would have decreased by more than 2.5; moreover, these simulations indicate that optimal traffic distribution mechanisms become significantly more efficient by just slightly alleviating the requirement to route along shortest paths. 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 [19]: 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 [21]) 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.

* Conference version in Proc. IFIP Networking 2005.

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. However, we establish a computationally efficient 2-approximation scheme for the problem i.e., our algorithm provides a solution that, in terms of congestion, is within a factor of at most 2 away from the optimum.

Organization: In Section 2, we introduce some terminology and definitions, and formulate the main problems considered in this study. In Section 3, we consider the problem of minimizing congestion under path quality constraints and provide both accurate as well as approximate solutions. In Section 4, we investigate the problem of minimizing congestion subject to a restriction on the number of paths per destination; we show that the problem is NP-hard and provide a computationally efficient 2-approximation scheme. In Section 5, we present simulation results that demonstrate the major advantage of optimal congestion-reduction schemes over two well-known heuristics. Finally, Section 6 summarizes the main results and discusses future directions for future research.

2. Model and Problem Formulation

A *network* is represented by a connected 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 $p=(v_0,v_1,\ldots v_h)$, such that, for $0\le n\le h-1$, $(v_n,v_{n+1})\in E$. A path is *simple* if all its nodes are distinct. Given a source node $s\in V$ and a target node $t\in V$, $P^{(s,t)}$ is the set of (all) directed paths in $G(V,E)$ from s to t. Let $P_{simple}^{(s,t)}\subseteq P^{(s,t)}$ represent the set of (all) *simple paths* in $G(V,E)$ from s to t. Finally, for each path $p\in P_{simple}^{(s,t)}$ and link $e\in E$, define a link-path indicator $\delta_e(p)$, which is 1 if link e is contained in p, and is 0 otherwise.

We consider a *link state* routing environment, where each source node has an image of the entire network. Each link $e\in E$ is assigned a *length* $l_e\in\mathbb{Z}^+$ and a *capacity* $c_e\in\mathbb{Z}^+$. Given a (non-empty) path p, the *length* $L(p)$ of p is defined as the sum of lengths of its links, namely, $L(p)\triangleq\sum_{e\in p}l_e$.

We consider two types of network flow representations. In the *path flow* representation, each variable $f_p\ge 0$ is the flow on some simple path $p\in P_{simple}^{(s,t)}$. Given two nodes $s,t\in V$ and a (flow) demand γ, we say that a path flow f is feasible *iff* it satisfies (1) the flow demand requirement, i.e., $\sum_{p\in P_{Simple}^{(s,t)}} f_p=\gamma$, and (2) the capacity constraints i.e., $\sum_{p\in P_{Simple}^{(s,t)}}\delta_e(p)\cdot f_p\le c_e$ for each $e\in E$. In the *link flow* representation, each variable $f_e\ge 0$ is the flow on some link $e\in E$. Given two nodes $s,t\in V$ and a demand γ, a link flow f is feasible *iff* there exists some feasible path flow $\widetilde{f}$ for the given instance such that $f_e=\sum_{p\in P_{Simple}^{(s,t)}}\delta_e(p)\cdot\widetilde{f_p}$ for each $e\in E$.

We proceed to formulate the criterion for congestion. Given a network $G(V,E)$ and a link flow $\{f_e\}$, the value $\frac{f_e}{c_e}$ is the *link congestion factor* and the value $\max_{e\in E}\left\{\frac{f_e}{c_e}\right\}$ is the *network congestion factor*. As noted in [3],[14],[24], the network congestion factor provides a good indication of congestion. In the appendix, we show that the problem of minimizing the network congestion factor is equivalent to the well-known Maximum Flow Problem [1]. Hence, when there are no restrictions on the paths (in terms of the number of paths or the length of each path), one can find a path flow that minimizes the network congestion factor in polynomial time through a standard max-flow algorithm.

We are ready to formulate the two problems considered in this study. The first problem aims at minimizing the network congestion factor subject to a restriction on the "quality" (i.e., length) of each of the chosen paths.

Problem RMP (Restricted Multipath) Given are a network $G(V,E)$, two nodes $s,t\in V$, a length $l_e>0$ and a capacity $c_e>0$ for each link $e\in E$, a demand $\gamma>0$ and a *length restriction* L for each routing path. Find a feasible path flow that minimizes the network congestion factor such that, if $P\subseteq P^{(s,t)}$ is the set of paths in $P^{(s,t)}$ that are assigned a positive flow, then, for each $p\in P$, it holds that $L(p)\le L$.

Remark 1: For convenience, and without loss of generality, we assume that the length l_e of each link $e\in E$ is not larger than the length restriction L. Clearly, links that are longer than L can be erased.

The next problem considers the requirement to limit the number of different paths over which a given demand is shipped while at the same time minimizing the network congestion factor.

Problem KPR (K-Path 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 number of routing paths K. Find a feasible path flow that minimizes the network congestion factor, such that, if $P\subseteq P^{(s,t)}$ is the set of paths in $P^{(s,t)}$ that are assigned a positive flow, then $|P|\le K$.

Remark 2: In both problems, the source-destination pair (s,t) is assumed to be connected i.e., $|P^{(s,t)}|\ge 1$.

3. Minimizing Congestion under Path Quality Constraints

In this section we investigate Problem RMP, i.e., the problem of minimizing congestion under path quality constraints. In subsection 3.1, we prove that Problem RMP is computational intractable. Accordingly, in subsection 3.2 we establish a pseudo-polynomial solution and in subsection 3.3 we design an ε-optimal approximation scheme for the problem.

3.1. Intractability of Problem RMP

We show that Problem RMP can be reduced to the *Partition Problem* [12].

Theorem 1: Problem RMP is NP-hard.

Proof: Consider the following instance of the Partition problem; given a set of elements $a_1,a_2,\ldots, a_{2n}$ 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 every $1\le i\le n$ and $\sum_{a\in A'} s(a) = \sum_{a\in A-A'} s(a)$.

We transform Partition to RMP as follows (see also Fig. 1):

1. Given an element $a_i\in A$ with size $s(a_i)$, define a unit capacity link $u_i\rightarrow v_i$ with length $s(a_i)$.
2. For each link $u_{2i-1}\rightarrow v_{2i-1}$, $1\le i\le n$, 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 length.
3. For each link $u_{2i}\rightarrow v_{2i}$, $1\le i\le n$, 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 length.
4. 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 length.
5. Set: $L\leftarrow \frac{1}{2}\cdot\sum_{a\in A} s(a)$ and $\gamma\leftarrow 2$.

We shall prove that it is possible to transfer 2 flow units over paths whose lengths are not larger than L without exceeding a 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 every $1\le i\le n$ and $\sum_{a\in A'} s(a) = \sum_{a\in A\setminus A'} s(a)$. (Remark: We refer to elements and their sizes interchangeably.)

$\Leftarrow$: Suppose there exists a subset $A'\subseteq A$ such that A' contains exactly one of a_{2i-1},a_{2i} for each $1\le i\le n$ and $\sum_{a\in A'} s(a) = \sum_{a\in A\setminus A'} s(a)$. Then, it is easy to see that the selection of the links that represents the elements in A' and the zero length 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 $A\setminus A'$ 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 length restriction is preserved since the two defined paths have length of $0.5\cdot\sum_{a\in A} s(a)$, which was defined to be the length restriction L.

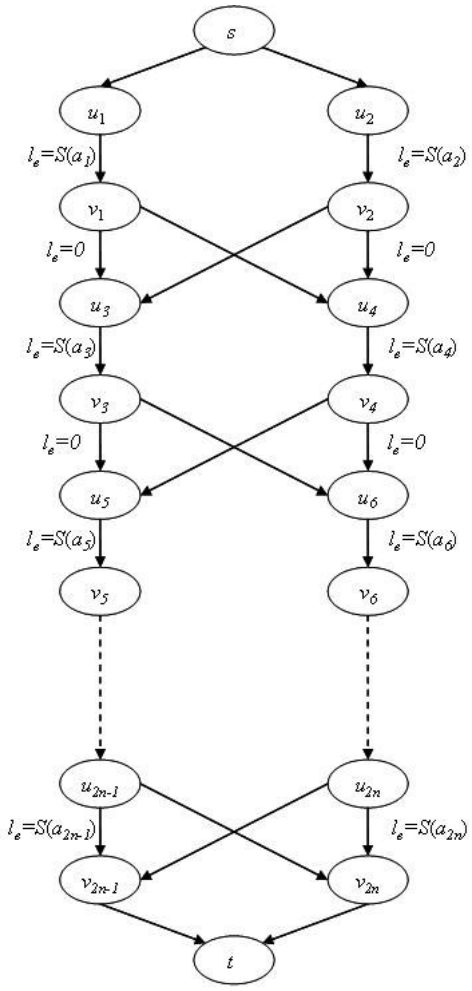

Fig. 1 Reduction of Partition to RMP

$\Rightarrow$: Suppose there is a path flow that transfers two flow units over paths that are not longer than L. Select one path that transfers a positive flow and denote it as p. Define an empty set S. For every link in p, with length $s(a_i)$, 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 transfers at most one unit of flow, there must be another path that is disjoint to the selected path and 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 $1\le i\le n$, and since both paths are disjoint, $S\cup S' = \{a_i\}_{i=1}^{2n} = A$. Finally, since both paths have lengths that are not longer than L, we have:

$$\sum_{a\in S} s(a) \le L, \quad \sum_{a\in S'} s(a) \le L. \qquad \text{(i)}$$

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

$$\sum_{a\in S} s(a) + \sum_{a\in S'} s(a) = \sum_{a\in A} s(a) = 2\cdot L. \qquad \text{(ii)}$$

Note that if variables x_1, x_2 satisfy $x_1 \le B$, $x_2 \le B$ and in addition $x_1 + x_2 = 2B$, it follows that $x_1 = x_2 = B$. Accordingly, we conclude from (i) and (ii) that $\sum_{a\in S} s(a) = \sum_{a\in S'} s(a) = L$.

Thus, problem RMP is NP hard. ■

3.2. Pseudo-Polynomial Algorithm for Problem RMP

The first step towards obtaining a solution to Problem RMP is to define it as a linear program. To that end, we need some additional notation.

Recall that we are given a network $G(V,E)$, two nodes $s,t \in V$, a length $l_e > 0$ and a capacity $c_e > 0$ for each link $e \in E$, a demand $\gamma > 0$ and a length restriction L for each routing path. Let α be the network congestion factor. Denote by f_e^{λ} the total flow along $e=(u,v) \in E$ that has been routed from s to u through paths with a total length of λ. 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 a linear program over the variables $\left\{\left\{f_e^{\lambda}\right\}, \alpha\right\}$, as specified in Fig 2.

Program RMP $\left(G(V,E), \{s,t\}, \{l_e\}, \{c_e\}, \gamma, L\right)$

Minimize α (1)

Subject to:

$$\sum_{e\in O(v)} f_e^{\lambda} - \sum_{e\in I(v)} f_e^{\lambda - l_e} = 0 \qquad \forall v \in V \setminus \{s,t\}, \forall \lambda \in [0,L] \qquad (2)$$

$$\sum_{e\in O(s)} f_e^{\lambda} - \sum_{e\in I(s)} f_e^{\lambda - l_e} = 0 \qquad \forall \lambda \in [1,L] \qquad (3)$$

$$\sum_{e\in O(s)} f_e^{0} = \gamma \qquad (4)$$

$$\sum_{\lambda=0}^{L} f_e^{\lambda} \le c_e \cdot \alpha \qquad \forall e \in E \qquad (5)$$

$$f_e^{\lambda} = 0 \qquad \forall e \in E,\ \lambda \notin [0, L - l_e] \qquad (6)$$

$$f_e^{\lambda} \ge 0 \qquad \forall e \in E,\ \lambda \in [0,L] \qquad (7)$$

$$\alpha \ge 0 \qquad (8)$$

Fig. 2. Program RMP

The objective function (1) minimizes the network congestion factor. Constraints (2), (3) and (4) are nodal flow conservation constraints. Equation (2) states that the traffic flowing out of node v, which has traversed through paths $p \in P^{(s,v)}$ of length $L(p) = \lambda$, has to be equal to the traffic flowing into node v, through paths $p' \in P^{(s,u)}$ and links $e=(u,v) \in E$, such that $L(p') + l_e = \lambda$; since $\lambda \in [0,L]$, the length restriction is obeyed; finally, equation (2) must be satisfied for each node other than the source s and the target t. Equation (3) extends the validity of equation (2) to hold for traffic that encounters source s after it has already passed through paths with non-zero length. Informally, equation (3) states that "old" traffic that emanates from s *not* for the first time (through a directed cycle that contains the source s) must satisfy the nodal flow conservation constraint of equation (2), which *solely* focuses on nodes from $V \setminus \{s,t\}$. Equation (4) states that the total traffic flowing out of source s, which has traversed paths of length $L = 0$, must be equal to the demand γ. Informally, equation (4) states that the total "new" traffic that emanates from the source s for the first time must satisfy the flow demand γ. Equation (5) is the link capacity utilization constraint. It states that the maximum link utilization is not larger than the value of the variable α i.e., the network congestion factor is at most α. Expression (6) rules out non-feasible flows and Expressions (7) and (8) restrict all variables to be non-negative.

We can solve Program RMP (Fig. 1) using any polynomial time algorithm for linear programming [16]. The solution to the problem is then achieved by decomposing the output of Program RMP (i.e., link flow $\left\{f_e^{\lambda}\right\}$) into a path flow that satisfies the length restriction L. Standard flow techniques that transform flows along links into flows along paths (e.g., the Flow Decomposition Algorithm [1]), cannot be used for our purpose since they do not respect the length restrictions. This is illustrated by the following example.

Example 1: Consider the network depicted in Fig. 3. Suppose that the flow along each link is equal to one unit i.e., $f_{e_i} = 1$ for each $i \in [1,6]$. Moreover, assume that the length restriction is 4. There are several path flows that can be decomposed out of the link flow $\{f_e\}$. For example, one such is the path flow f_1 that assigns one unit of flow to each of the paths $\{e_1, e_4\}$, $\{e_2, e_5\}$, $\{e_3, e_6\}$; indeed, since f_1 transfers one unit of flow along each link, its link flow representation is $\{f_e\}$. However, since the length of the path $\{e_3, e_6\}$ is 6, the path flow f_1 violates the length restriction. On the other hand, if we decompose the link flow $\{f_e\}$ into the path flow that assigns one unit to each of the paths $\{e_1, e_6\}$, $\{e_2, e_5\}$, $\{e_3, e_4\}$, the length restriction is satisfied on all paths.

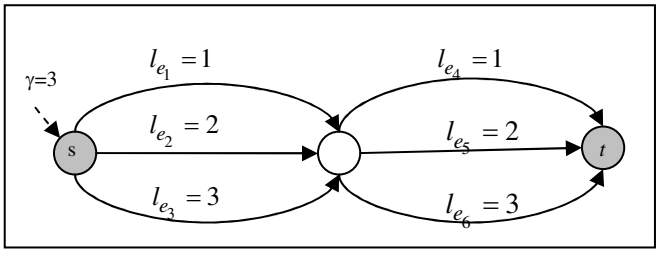

Fig 3. A single link flow can be decomposed into several path flows. Some of them satisfy the length restriction and the rest violate it.

Our goal is therefore to establish an efficient algorithm that decomposes link flow $\{f_e^\lambda\}$, into a path flow that satisfies the length restrictions. Accordingly, consider Algorithm PFC, which is specified in Fig.4 and is outlined as follows.

Algorithm PFC $\left(G(V,E),\{s,t\},\{f_e^\lambda\},\gamma\right)$

A. Initialization:
For each path $p \in P^{(s,t)}$: $g_p \leftarrow 0$.

B. While $\gamma > 0$ do:

1. $\mathcal{P} \leftarrow$ Path_Construction$\left(G(V,E),\{s,t\},\{f_e^\lambda\}\right)$.
2. $f_{e_i}^{\Lambda_i} \leftarrow f_{e_i}^{\Lambda_i} - \min_{(e_i,\Lambda_i)\in\mathcal{P}}\{f_{e_i}^{\Lambda_i}\}$, for each $(e_i,\Lambda_i)\in\mathcal{P}$.
3. $\gamma \leftarrow \gamma - \min_{(e_i,\Lambda_i)\in\mathcal{P}}\{f_{e_i}^{\Lambda_i}\}$.
4. Let p be the path that corresponds to $\mathcal{P}$ i.e., $e_i \in p$ *iff* $(e_i,\Lambda_i)\in\mathcal{P}$. Perform $g_p \leftarrow \min_{(e_i,\Lambda_i)\in\mathcal{P}}\{f_{e_i}^{\Lambda_i}\}$.

C. Convert each non-simple path that carries a positive flow in g into a simple path as follows:

a. Let $h: P^{(s,t)} \to P_{simple}^{(s,t)}$ map each path $p=(v_0,v_1,\cdots v_{h-1},v_h,\cdots,v_h,v_{h+1},\cdots v_n)\in P^{(s,t)}$ into the corresponding simple path $p'=(v_0,v_1,\cdots v_{h-1},v_h,v_{h+1},\cdots v_n)\in P_{simple}^{(s,t)}$.

b. For each $p\in P_{simple}^{(s,t)}$, $f_p \triangleq \sum_{\hat{p}\,|\,h(\hat{p})=p} g_{\hat{p}}$.

D. Return path flow f.

Fig 4. Algorithm Path Flow Construction (PFC)

Algorithm PFC is an iterative algorithm that identifies at each iteration a single path with a length of at most L whose corresponding link flows $\{f_e^\lambda\}$ are all positive; the path is identified through Procedure Path Construction, which is specified in Fig. 5. The flow over the corresponding path is defined to be equal to the smallest flow f_e^λ that belongs to the path. Then, the algorithm subtracts the flow that traverses through that path from the demand γ and from each variable f_e^λ in the path. The (iterative) algorithm repeats this process until the demand γ is zeroed. Thus, the resulting path flow transfers γ flow units along paths with length of at most L. Finally, since Procedure Path Construction might return non-simple paths, the algorithm converts all non-simple paths in the resultant path flow into simple paths by eliminating their loops.

We turn to explain the main idea behind Procedure Path Construction (Fig. 5). The procedure identifies a path $p\in P^{(s,t)}$, $p: s \xrightarrow{e_0} u_1 \xrightarrow{e_1} u_2 \cdots u_{h-1} \xrightarrow{e_{h-1}} t$, whose corresponding variables $\left\{f_{e_0}^0, f_{e_1}^{l_{e_0}}, \cdots, f_{e_{h-1}}^{l_{e_0}+\cdots+l_{e_{h-2}}}\right\}$ are all positive. Consider the positive variable $f_{e_{h-1}}^{l_{e_0}+\cdots+l_{e_{h-2}}}$. Since equation (6) of Program RMP zeroes each variable f_e^λ with a length $\lambda \notin [0, L-l_e]$, it holds for the positive variable $f_{e_{h-1}}^{l_{e_0}+\cdots+l_{e_{h-2}}}$ that $l_{e_0}+l_{e_1}+\cdots+l_{e_{h-2}} \le L-l_{e_{h-1}}$; hence, $\sum_{i=0}^{h-1} l_{e_i} \le L$. In other words, each path $p\in P^{(s,t)}$ with a corresponding sequence of positive variables $\left\{f_{e_0}^0, f_{e_1}^{l_{e_0}}, \cdots, f_{e_{h-1}}^{l_{e_0}+\cdots+l_{e_{h-2}}}\right\}$ has a length $L(p)$, $L(p)=\sum_{i=0}^{h-1} l_{e_i} \le L$. Thus, in order to establish a path $p\in P^{(s,t)}$ with a length of at most L, it is sufficient to find a sequence of positive variables $\left\{f_{e_0}^0, f_{e_1}^{l_{e_0}}, \cdots, f_{e_{h-1}}^{l_{e_0}+\cdots+l_{e_{h-2}}}\right\}$ such that the link e_0 emanates from source s and the link e_{h-1} enters into the destination t i.e., $e_0 \in O(s)$ and $e_{h-1}\in I(t)$. To that end, we employ the following property that characterizes the solutions of Program RMP. If a *positive* flow $f_e^{\lambda-l_e}$ enters through the link $e=(u,v)$ into the node $v\in V\setminus\{s,t\}$, it follows that there is some link $e'=(v,w)$ such that $f_{e'}^\lambda > 0$. Therefore, since $f_{e''}^0 > 0$ for some link e'' that emanates from the source s, it is possible to follow positive flows (variables) from s in order to construct a sequence of positive variables $\left\{f_{e_0}^0, f_{e_1}^{l_{e_0}}, \cdots, f_{e_k}^{l_{e_0}+\cdots+l_{e_{k-1}}}, \cdots\right\}$ such that $e_0 \in O(s)$. We now prove that, by following these positive variables for a finite number of times, we eventually encounter a positive variable $f_{e_k}^{l_{e_0}+\cdots+l_{e_k}}$ such that e_k enters the destination t i.e., we eventually identify a directed path from s to t with a length of at most L.

Procedure Path_Construction $\left(G(V,E),\{s,t\},\{f_e^\lambda\}\right)$

Initialization
$\mathcal{P} \leftarrow \phi$, $u_0 \leftarrow s$, $\Lambda_0 \leftarrow 0$, $i \leftarrow 0$

While $u_i \neq t$ do

1. Select a positive variable $f_{e_i}^{\Lambda_i}$ such that $e_i \triangleq (u_i,u_{i+1})\in E$.
2. $\mathcal{P} \leftarrow \mathcal{P} \cup \{(e_i,\Lambda_i)\}$, $\Lambda_{i+1} \leftarrow \Lambda_i + l_{e_i}$, $i \leftarrow i+1$.

Return $\mathcal{P}$

Fig 5. Procedure Path Construction

Lemma 1*:* Consider the nodes $\{u_i\}$ identified by Procedure Path Construction (step 1). There exists an h, $h \le L$, such that the sequence $(u_0,u_1,\ldots,u_h)$ is a path from s to t.

Proof: It follows from constraint (4) that, since $\gamma>0$, there exists some link $e_0=(u_0,u_1)$ such that the variable $f_{e_0}^0$ is positive. Then, from constraints (2) and (3), it follows that, if $u_1 \neq t$, then there exists some link $e_1=(u_1,u_2)$ such that the variable $f_{e_1}^{l_{e_0}}$ is positive. Thus, applying constraints (2) and (3) for any index i, it follows that, if there exists a positive variable $f_{e_i}^{\Lambda_i}$ where $\Lambda_i \triangleq \sum_{k=0}^{i-1} l_{e_k}$, then, unless $u_{i+1}=t$, there exists a link $e_{i+1}=(u_{i+1},u_{i+2})$ such that the variable $f_{e_{i+1}}^{\Lambda_i+l_{e_i}}$ is positive. Therefore, if $f_{e_i}^{\Lambda_i}=0$ for each $i>K$, it must hold that there exists an index j, $j \leq K$, such that $u_j=t$. Hence, in order to establish the lemma, it is sufficient to show that $f_{e_i}^{\Lambda_i}=0$ for each $i>L$.

Indeed, since $l_e \in \mathbb{Z}^+$ for each $e \in E$, it follows that $\Lambda_{i+1}-\Lambda_i \geq 1$ for any index i; hence $\Lambda_i > L$ for each $i>L$. Therefore, since it follows from constraint (6) that $f_{e_i}^{\Lambda_i}=0$ for each $\Lambda_i > L$, it holds that $f_{e_i}^{\Lambda_i}=0$ for each $i>L$. ■

For completion, in Fig. 6 we specify Algorithm RMP, which solves Problem RMP.

Algorithm RMP $\left(\mathrm{G}(\mathrm{V},\mathrm{E}),\{s,t\},\{l_e\},\{c_e\},\gamma,L\right)$

1. $\left\{f_e^\lambda\right\} \leftarrow$ **Program RMP** $\left(G(V,E),\{s,t\},\{l_e\},\{c_e\},\gamma,L\right)$
2. $f \leftarrow$ **Algorithm PFC** $\left(G(V,E),\{s,t\},\left\{f_e^\lambda\right\},\gamma\right)$
3. **Return** path flow *f*.

Fig. 6. Algorithm RMP

Next, we consider the complexity of Algorithm RMP. First, it follows from [16] that the complexity incurred by solving the linear program of step 1 is polynomial both in the number of variables $\left\{f_e^\lambda\right\}$ and in the number of constraints needed to formulate Program RMP. Thus, since both of these numbers are in the order of $M \cdot L$, the complexity of step 1 is polynomial in $O(M \cdot L)$. Consider now the complexity incurred by step 2 (Algorithm PFC). Since, by construction, each iteration of Algorithm PFC zeroes at least one variable f_e^λ, it follows that Algorithm PFC iterates for no more than the number of variables $\left\{f_e^\lambda\right\}$. Moreover, since the complexity of each iteration is dominated by the complexity of Procedure Path Construction, which, according to Lemma 1, consumes $O(L)$ operations, the complexity of Algorithm PFC is $L \cdot \left|\left\{f_e^\lambda\right\}\right| = O\left(M \cdot L^2\right)$. Thus, we conclude that the overall complexity of Algorithm RMP is polynomial in $M \cdot L^2$ i.e., Algorithm RMP is a *pseudo-polynomial time algorithm* [12]. Whenever the value of L is polynomial in the size of the problem, Algorithm RMP is a *polynomial optimal* algorithm for Problem RMP. One such case is when the *hop count* metric is considered (i.e., $l_e \equiv 1$), since then $L \leq N-1$.

3.3. An ε-Optimal Approximation Scheme for Problem RMP

In the previous subsection we established an optimal polynomial solution to Problem RMP for the case where the length restrictions are sufficiently small. In this subsection we turn to consider the solution to Problem RMP for *arbitrary* length restrictions. As Theorem 1 establishes that Problem RMP is NP-hard for this general case, we focus on the design of an efficient algorithm that approximates the optimal solution.

Our main result is the establishment of an ε–optimal approximation scheme, which is termed the *RMP Approximation Scheme*. This scheme is based on Algorithm RMP, specified in the previous subsection, which was shown to have a complexity that is polynomial in $M \cdot L^2$. Given an instance of Problem RMP and an approximation parameter ε, the RMP Approximation Scheme reduces the complexity of Algorithm RMP by first scaling down the length restriction L by a factor $\Delta = \frac{L \cdot \varepsilon}{N}$ and then rounding it into an integer. Obviously, as a result, it must also scale down the length of each link. However, in order to ensure that the optimal network congestion factor does not increase, the RMP Approximation Scheme relaxes the constraints of the new instance with respect to the constraints of the original instance. Specifically, after the RMP Approximation Scheme scales down the length restriction and the length of each link by the factor Δ, it rounds *up* the length restriction and rounds *down* the length of each link. Then, it invokes Algorithm RMP over the new instance, in order to construct a path flow that minimizes congestion while satisfying the relaxed length restrictions. In Theorem 2, we establish that the resulting path flow violates the length restriction by a factor of at most $(1+\varepsilon)$ and has a network congestion factor that is not larger than the optimal network congestion factor. The RMP Approximation Scheme is specified in Fig. 7.

RMP Approximation Scheme $\left(\mathrm{G},\{\mathrm{s,t}\},\{\mathrm{c_e}\},\{l_e\},\gamma,L,\varepsilon\right)$

2. $\Delta \leftarrow \frac{L \cdot \varepsilon}{N}$.
3. Let $\left\langle G(V,E),\{s,t\},\left\{\widetilde{c_e}\right\},\left\{\widetilde{l_e}\right\},\gamma,\widetilde{L}\right\rangle$ be an instance of Problem RMP such that:
   a. $\widetilde{L} \leftarrow \left\lceil \frac{L}{\Delta} \right\rceil$
   b. For each link $e \in E$:
   $\widetilde{l_e} \leftarrow \left\lfloor \frac{l_e}{\Delta} \right\rfloor$, $\widetilde{c_e} \leftarrow c_e$.
4. Invoke *Algorithm RMP* over the instance $\left\langle G(V,E),\{s,t\},\left\{\widetilde{c_e}\right\},\left\{\widetilde{l_e}\right\},\gamma,\widetilde{L}\right\rangle$ of Problem RMP. Let path flow *f* represent the output.
5. Return Path flow *f*.

Fig 7. RMP Approximation Scheme

Theorem 2: Given an instance $<G,\{s,t\},\{c_e\},\{l_e\},\gamma,L>$ of problem RMP and an approximation parameter ε, the *RMP Approximation Scheme* has a complexity that is polynomial in $1/\varepsilon$ and in the size of the network. Moreover, the output of the scheme is a path flow f that satisfies the following:

a. $\sum_{p\in P^{(s,t)}} f_p = \gamma$ i.e., the flow demand requirement is satisfied.

b. If α^* is the network congestion factor of the optimal solution, then, for each e$\in E$, it holds that $\sum_{p\in P^{(s,t)}} \delta_e(p)\cdot f_p \le \alpha^*\cdot c_e$, i.e., the network congestion factor is at most α^*.

c. For each path $p\in P^{(s,t)}$, if $f_p>0$ then p is simple and $L(p)\le(1+\varepsilon)\cdot L$, i.e., the length restriction is violated by a factor of at most $(1+\varepsilon)$.

Proof: We first show that the complexity of the RMP Approximation Scheme is polynomial in $1/\varepsilon$ and in the size of the network. To that end, recall that we have shown in the previous subsection that the complexity incurred by invoking Algorithm RMP over any instance (of Problem RMP) with a length restriction L is polynomial in $M\cdot L^2$. Therefore, since the RMP Approximation Scheme invokes Algorithm RMP over an instance with a length restriction $\tilde{L}=\left\lceil \frac{L}{\Delta}\right\rceil \le \frac{N}{\varepsilon}+1$, it follows that its complexity is polynomial in $O\left(M\cdot\left(\frac{N}{\varepsilon}\right)^2\right)$. We turn to prove parts (a), (b) and (c) of the Theorem.

a. Since equation (4) in Program RMP specifies that $\sum_{e\in O(s)} f_e^0 = \gamma$, it follows that every path flow representation $\{f_p\}$ of the link flow $\{f_e^\lambda\}$ must satisfy that $\sum_{p\in P^{(s,t)}} f_p = \gamma$.

b. Let α be the network congestion factor of the output f. We have to show that $\alpha\le\alpha^*$. In the previous subsection we have shown that Algorithm RMP returns an optimal solution to Problem RMP. Thus, we only have to show that the space of feasible solutions of the instance $<G,\{s,t\},\{\tilde{c}_e\},\{\tilde{l}_e\},\gamma,\tilde{L}>$ (that constitutes an input to Algorithm RMP in step 4) contains the space of feasible solutions of the given instance $<G,\{s,t\},\{c_e\},\{l_e\},\gamma,L>$. Hence, it is sufficient to show that the length restrictions in $<G,\{s,t\},\{\tilde{c}_e\},\{\tilde{l}_e\},\gamma,\tilde{L}>$ are *relaxed* with respect to the length restrictions in $<G,\{s,t\},\{c_e\},\{l_e\},\gamma,L>$. Specifically, given a path p in $G(V,E)$ that satisfy $L(p)\le L$ with respect to $\{l_e\}$, we now show that $L(p)\le\tilde{L}$ with respect to $\{\tilde{l}_e\}$. By the selection of p it holds that $\sum_{e\in p} l_e = L(p)\le L$; hence, $\sum_{e\in p}\frac{l_e}{\Delta}\le\frac{L}{\Delta}$. Therefore, it holds that $\sum_{e\in p}\left\lfloor\frac{l_e}{\Delta}\right\rfloor\le\left\lceil\frac{L}{\Delta}\right\rceil$; hence, by definition, $\sum_{e\in p}\tilde{l}_e\le\tilde{L}$.

c. First, note that, by construction, the output of Algorithm RMP carries positive flow only over simple paths; hence, p is simple. We turn to show that $L(p)\le(1+\varepsilon)\cdot L$. Since p transfers a positive flow, it holds that $\sum_{e\in p}\tilde{l}_e\le\tilde{L}$; hence, by definition $\sum_{e\in p}\left\lfloor\frac{l_e}{\Delta}\right\rfloor\le\left\lceil\frac{L}{\Delta}\right\rceil$. Therefore, it holds that $\sum_{e\in p}\left(\frac{l_e}{\Delta}-1\right)\le\frac{L}{\Delta}+1$; hence, $\sum_{e\in p} l_e \le L+\left(|p|+1\right)\cdot\Delta$, where $|p|$ is the hop count of path p. Finally, since p is simple, it holds that $|p|\le N-1$; hence, $L(p)=\sum_{e\in p} l_e \le L+N\cdot\Delta = = L+L\cdot\varepsilon = L\cdot(1+\varepsilon)$.

Thus, the Theorem is established. ■

3.4. Extensions

In the following, we outline two important extensions to Problem RMP.

Multi-Commodity Extensions: In order to simplify the presentation of the solution to Problem RMP, we focused in this paper only on the single commodity case i.e., we assume that only one source-destination pair exists in every instance of Problem RMP. Following basically the same lines as in Subsections 3.1 and 3.2, we present in [4] a pseudo-polynomial solution and an ε-optimal approximation for the multi-commodity extension of Problem RMP.

End-to-End Reliability Constraints: When traffic is split among multiple paths, a failure in any of these paths may result in the failure of the entire transmission. As a result, multipath routing may be more sensitive to network failures than single-path routing. One solution is to assign to each link $e\in E$ a failure probability p_e (which can be estimated out of available failure statistics of each network component) and then to minimize the network congestion factor such that the traffic is routed solely along paths with a success probability larger than some given lower bound Π. Based on Algorithm RMP, it is possible to construct an ε–optimal approximation scheme for such a problem. Specifically, for any approximation parameter ε, the ε-optimal approximation scheme constructs an instance of Problem RMP by assigning a length $l_e=\left\lfloor \log_{1+\frac{\varepsilon}{N}}(1-p_e)\right\rfloor$ to each link e and a length restriction $L=\left\lceil \log_{1+\frac{\varepsilon}{N}}\Pi\right\rceil$; then, it invokes Algorithm RMP over the resulting instance. In [4], we show that the solution is obtained within a time that is polynomial in $\frac{1}{\varepsilon}$; moreover, it is shown to minimize the network congestion factor while violating the requirement on the end-to-end success probability by a factor of at most $(1+\varepsilon)$. Due to space limits, the details are omitted here.

4. Minimizing Congestion with K Routing Paths

In this section we investigate Problem KPR, which minimizes congestion while routing traffic along at most K different paths. In subsection 4.1 we prove that Problem KPR is NP-hard in the general case but admits a (straightforward) polynomial solution when the restriction on the number of paths is larger than the number of links (i.e., $K \geq M$). Accordingly, in subsections 4.2 and 4.3 we devise a 2- approximation scheme for the more interesting case where $K<M$.

4.1. (In)tractability of Problem KPR

The following Theorem establishes the intractability of Problem KPR for the general case.

Theorem 3: Problem KPR is NP-hard.

Proof: We reduce Problem KPR to the *Single-Source Unsplittable Flow Problem* that was shown to be NP hard in [17] and is defined as follows: given are a network $G(V,E)$, a capacity $c_e>0$ for each link $e \in E$, a set of source-destination pairs (s,t_1), $(s,t_2),\ldots,(s,t_k)$ associated with demands $\gamma_1, \gamma_2,\ldots, \gamma_k$; is there an assignment of traffic to paths such that for each $1 \leq i \leq k$ the demand γ_i is routed over a single path $p \in P^{(s,t_i)}$ without violating the capacity constraints?

The Single Source Unsplittable Flow Problem is transformed to Problem KPR as follows (see Fig. 8): Add an "aggregated" target T; then, for each $i \in [1,k]$ add a link $t_i \rightarrow T$ with a capacity γ_i.

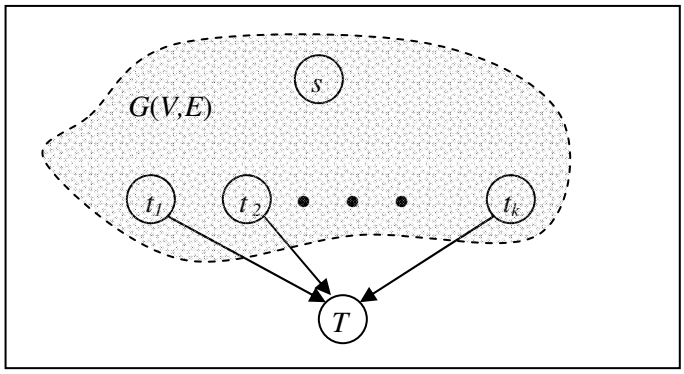

Fig 8. Reducing the Single-Source Unsplittable Flow Problem into Problem KPR.

We shall prove that it is possible to find an assignment of the demands $\gamma_1, \gamma_2,\ldots, \gamma_k$ to paths such that for each $1 \leq i \leq k$ the demand γ_i is routed over a single path $p \in P^{(s,t_i)}$ without violating the capacity constraints *iff* there exists a path flow that transfers $\gamma = \sum_{i=1}^{k} \gamma_i$ flow units from s to T over at most k paths without exceeding a network congestion factor of α=1.

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

$\Leftarrow$: There exists a path flow f that transfers $\gamma = \sum_{i=1}^{k} \gamma_i$ flow units from s to T over at most k paths without exceeding a network congestion factor of α=1. Consider the cut (A,A') where $A=V$ and $A'=\{T\}$. Since the capacity of the cut (A,A') is equal to the demand γ, the path flow f employs each link $t_i \rightarrow T$ where $1 \leq i \leq k$. Yet, since it employs at most k paths, f must have exactly one path with positive flow between s and t_i for each $1 \leq i \leq k$; moreover, since the flow over each link $t_i \rightarrow T$ must equal to the capacity of that link, it holds for each $1 \leq i \leq k$ that the (single) path that f employs in order to carry traffic from s to t_i transfers γ_i flow units.

Thus, Problem KPR is NP hard. ■

Although Problem KPR is NP hard in the general case, we now show that it admits a polynomial solution when the restriction on the number of paths is larger than the number of links (i.e.,$K \geq$M). Specifically, in the appendix we show that it is possible to obtain a flow that minimizes the network congestion factor with a single execution of a max-flow algorithm. Moreover, using the *Flow Decomposition Algorithm* [12], it is possible to transform in polynomial time *every* link flow representation into a path flow representation that admits at most M routing paths. Therefore, with a single execution of a max-flow algorithm followed by a single execution of the Flow Decomposition Algorithm, it is possible to solve Problem KPR in polynomial time in the case $K \geq$M.

4.2. The Integral Routing Problem

Our approximation scheme (for the case $K<M$) is based on solving an auxiliary problem that minimizes congestion while restricting the flow along each path to be *integral* in $\gamma/$K. In order to formulate the corresponding problem, consider first the following definition.

Definition 1: Given are a network $G(V,E)$, a capacity $c_e>0$ for each link $e \in E$, a demand γ and an integer $K<M$. A feasible path flow f is said to be $\gamma/$K-*integral*, if for each path $p \in P^{(s,t)}$, it holds that $f(p)$ is a multiple of $\gamma/$K.

Problem Integral 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 an integer $K<M$. Find a γ/K-integral path flow that minimizes the network congestion factor, such that the demand γ is satisfied.

The following observation shall be used in order to construct a polynomial solution to the Integral Routing Problem.

Lemma 2: Given an instance $<G,\{s,t\},\{c_e\},\gamma,K>$ of the Integral Routing Problem, the optimal network congestion factor is included in the set $\overline{\alpha} \triangleq \left\{ \frac{i \cdot \gamma}{K \cdot c_e} \middle| e \in E, i \in [0,K] \cap \mathbb{Z} \right\}$.

Proof: In order to prove the lemma, it is sufficient to show that the set $\overline{\alpha}$ contains the network congestion factor of *all* γ/K-integral path flows that transfer γ flow units from s to t over $G(V,E)$. Assume that f is one such path flow (i.e., γ/K-integral path flow that transfers γ flow units from s to t), and denote by α its network congestion factor. In order to prove the Theorem we now show that $\alpha \in \overline{\alpha}$. Since f is a γ/K-integral path flow it holds by definition that f_p is a multiple of γ/K for each path $p \in P^{(s,t)}$. Hence, the flow over each link $e \in E$ must also be a multiple of γ/K i.e., for each $e \in E$ there exists an integer i in the range $[0,K]$ such that $f_e = \frac{i \cdot \gamma}{K}$; therefore, for each $e \in E$ it holds that $\frac{f_e}{c_e} \in \left\{ \frac{i \cdot \gamma}{K \cdot c_e} \middle| i \in [0,K] \cap \mathbb{Z} \right\}$. In particular, $\max_{e \in E} \left\{ \frac{f_e}{c_e} \right\} \in \bigcup_{e \in E} \left\{ \frac{i \cdot \gamma}{K \cdot c_e} \middle| i \in [0,K] \cap \mathbb{Z} \right\} = \left\{ \frac{i \cdot \gamma}{K \cdot c_e} \middle| e \in E, i \in [0,K] \cap \mathbb{Z} \right\} = \overline{\alpha}$; hence, $\alpha \in \overline{\alpha}$. ■

Remark 3: Observe that the size of $\overline{\alpha}$ is polynomial in the network size i.e., $\left|\overline{\alpha}\right| \le M \cdot (K+1) = O(M^2)$.

We now introduce *Procedure Test* (Fig. 9), which is given an instance $<G,\{s,t\},\{c_e\},\gamma,K>$ of the Integral Routing Problem and a restriction α on the network congestion factor. If there exists a γ/K–integral path flow for the given instance with a network congestion factor of at most α, then the procedure returns it (and is said to *succeed*). Otherwise the procedure returns *Fail*.

Procedure Test $\left(G,\{s,t\},\{c_e\},\gamma,K,\alpha\right)$

1. Multiply all link capacities by α and round down each capacity to the nearest multiple of $\frac{\gamma}{K}$ i.e., set $\widetilde{c_e} \leftarrow \frac{\gamma}{K} \cdot \left\lfloor \frac{\alpha \cdot c_e}{\gamma/K} \right\rfloor$ for each $e \in E$.
2. Solve the instance $\left\langle G,(s,t),\{\widetilde{c_e}\} \right\rangle$ of the Maximum Flow Problem using the *Push Relabel Algorithm* [13]. Let the link flow $\{f_e\}$ represent the solution, and let F be the total transferred flow from s to t.
3. If $F \ge \gamma$
   Return the link flow $\{f_e\}$.
   Else
   Return Fail.

Fig. 9: Procedure Test

We turn to explain the main idea behind Procedure Test. Initially, the procedure multiplies all link capacities by a factor of α in order to impose the restriction on the network congestion factor; indeed, multiplying all capacities by α assures that the flow f_e along each link $e \in E$ is at most $\alpha \cdot c_e$; therefore, for each $e \in E$, the link congestion factor $\frac{f_e}{c_e}$, and, in particular, the network congestion factor $\max_{e \in E} \left\{ \frac{f_e}{c_e} \right\}$, are at most α. Next, the procedure rounds down the capacity of each link to the nearest multiple of γ/K; since the flow over each path in every solution to the Integral Routing Problem is γ/K-integral, such a rounding has no effect on the capability to transfer the flow demand γ. Finally, the procedure applies any standard maximum flow algorithm that returns an integral link flow when all capacities are integral. Since all capacities are γ/K-integral, the maximum flow algorithm determines a γ/K-integral link flow that transfers the maximum amount of flow without violating the restriction α on the network congestion factor. If this link flow succeeds to transfer at least γ flow units from s to t, then the procedure returns it. Otherwise, the procedure fails.

Theorem 4: Given are an instance $<G,\{s,t\},\{c_e\},\gamma,K>$ of the Integral Routing Problem and a restriction α on the network congestion factor. Denote by α^* the corresponding optimal network congestion factor. Then, Procedure Test succeeds for the instance $<G,\{s,t\},\{c_e\},\gamma,K>$ with the restriction α *iff* $\alpha \ge \alpha^*$.

Proof: $\Rightarrow$: Suppose that Procedure Test succeeds for the instance $<G,\{s,t\},\{c_e\},\gamma,K>$ with the restriction α. Then, by construction, the procedure returns a link flow $\{f_e\}$. We first show that $\{f_e\}$ is a feasible solution to the instance $<G,\{s,t\},\{c_e\},\gamma,K>$ (i.e., $\{f_e\}$ satisfies the demand and the integrality constrains) while producing a network congestion factor of at most α. By construction (steps 2 and 3), $\{f_e\}$ is a solution to the instance $\left\langle G,(s,t),\{\widetilde{c_e}\} \right\rangle$ of the Maximum Flow Problem that meets the demand requirement γ. Moreover, since a solution to the instance $\left\langle G,(s,t),\{\widetilde{c_e}\} \right\rangle$ satisfies the capacity constraint $f_e \le \widetilde{c_e}$ for each link $e \in E$, it follows that $f_e \le \frac{\gamma}{K} \cdot \left\lfloor \frac{\alpha \cdot c_e}{\gamma/K} \right\rfloor \le \alpha \cdot c_e$ for each $e \in E$; hence, $\max_{e \in E} \left\{ \frac{f_e}{c_e} \right\} \le \alpha$. Finally, we show that $\{f_e\}$ can be decomposed into a γ/K-integral path flow. To that end, note that the capacities of the instance $\left\langle G,(s,t),\{\widetilde{c_e}\} \right\rangle$ are integral in γ/K. Therefore, the *Push-Relabel* method returns a γ/K-*integral* link flow [13], which can be decomposed (using the Flow Decomposition Algorithm [1]) into a γ/K-*integral* path flow. We therefore conclude that $\{f_e\}$ is a feasible solution for $<G,\{s,t\},\{c_e\},\gamma,K>$ and has a network congestion factor of at most α. Hence, since α^* is the smallest network congestion factor among all feasible solutions to the instance

$<G,\{s,t\},\{c_e\},\gamma,K>$, it holds that $\alpha \geq \alpha^*$.

$\Leftarrow$: Suppose that $\alpha \geq \alpha^*$ i.e., the given restriction on the network congestion factor is not smaller than the optimal network congestion factor. Then, there exits a feasible solution for the instance $<G,\{s,t\},\{c_e\},\gamma,K>$ of Problem Integral Routing that produces a network congestion factor of at most α. Thus, there exists a γ/K-integral path flow that transfers at least γ flow units from s to t over $G(V,E)$ when each link $e \in E$ has capacity $\alpha \cdot c_e$; let f be one such path flow. Then, since f is γ/K-integral, we can reduce the capacities $\{\alpha \cdot c\}_{e \in E}$ to become multiples of γ/K without affecting the capability of f to transfer the flow demand γ; hence, f can transfer at least γ flow units from s to t over $G(V,E)$ when each link $e \in E$ has a capacity $\frac{\gamma}{K} \cdot \left\lfloor \frac{\alpha \cdot c_e}{\gamma / K} \right\rfloor$. In particular, a maximum flow from s to t over $G(V,E)$ when each link $e \in E$ has a capacity $\frac{\gamma}{K} \cdot \left\lfloor \frac{\alpha \cdot c_e}{\gamma / K} \right\rfloor$ transfers at least γ flow units. Hence, the maximum flow that is computed in Step 2 transfers at least γ flow units. Thus, by construction, Procedure Test succeeds when it is invoked on the instance $<G,\{s,t\},\{c_e\},\gamma,K>$ with the restriction α on network congestion factor. ■

Theorem 4 has two important implications that enable to construct an efficient solution to the Integral Routing Problem. First, the theorem establishes that the smallest α for which Procedure Test succeeds with the input $<G,\{s,t\},\{c_e\},\gamma,K,\alpha>$ is equal to the optimal network congestion factor α^*. Therefore, if S is a finite set that includes α^* and α is the smallest network congestion factor in S such that Procedure Test succeeds for the input $<G,\{s,t\},\{c_e\},\gamma,K,\alpha>$, then $\alpha = \alpha^*$. This fact, together with the fact that the set $\overline{\alpha}$ includes α^* (as per Lemma 2), imply that, for every instance $<G,\{s,t\},\{c_e\},\gamma,K>$ of Problem Integral Routing, the optimal network congestion factor α^* is the smallest $\alpha \in \overline{\alpha}$, such that Procedure Test succeeds, for the input $<G,\{s,t\},\{c_e\},\gamma,K,\alpha>$. Moreover, since in case of a success Procedure Test returns the corresponding link flow, finding the smallest $\alpha \in \overline{\alpha}$, such that Procedure Test succeeds, identifies a link flow with a network congestion factor of α^*.

The second implication of Theorem 4 enables to employ a *binary search* when we seek the smallest $\alpha \in \overline{\alpha}$ such that Procedure Test succeeds. Indeed, it follows from Theorem 4 that, when Procedure Test succeeds for $\alpha_1 \in \overline{\alpha}$, it succeeds for all $\alpha \in \overline{\alpha}$, $\alpha \geq \alpha_1$; and when it fails for $\alpha_2 \in \overline{\alpha}$, it fails for all $\alpha \in \overline{\alpha}$, $\alpha \leq \alpha_2$; thus, if Procedure Test succeeds for $\alpha_1 \in \overline{\alpha}$ (alternatively, fails for $\alpha_2 \in \overline{\alpha}$) it is possible to eliminate from further consideration all the elements of $\overline{\alpha}$ that are larger than α_1 (correspondingly, smaller than α_2).

Remark 4: Note that performing a binary search over $\overline{\alpha}$ requires *sorting* all the elements of $\overline{\alpha}$, which consumes $O\left(\left|\overline{\alpha}\right| \cdot \log\left|\overline{\alpha}\right|\right) \leq O\left(M^2 \cdot \log N\right)$ operations [10].

Thus, we conclude that the employment of a binary search so as to find the smallest $\alpha \in \overline{\alpha}$ for which Procedure Test succeeds, establishes a link flow that has the minimum network congestion factor. The optimal solution is then achieved by decomposing the resulting link flow into a path flow via the Flow Decomposition Algorithm [1]. *Algorithm Integral Routing*, presented in Fig. 10, specifies these steps.

Algorithm Integral Routing $\left(G,\{s,t\},\{c_e\},\gamma,K\right)$

1. Sort the elements of the set $\overline{\alpha}$.
2. Employ a binary search over $\overline{\alpha}$ in order to find the smallest $\alpha \in \overline{\alpha}$ such that $\textbf{Test}\left(G,\{s,t\},\{c_e\},\gamma,K,\alpha\right) \neq \textbf{FAIL}$.
3. Let $\alpha_{\min}$ be the network congestion factor that was the output of the search process (step 2). Denote by $\{f_e\}$ the link flow that was output to indicate that Procedure Test has succeeded for $\alpha_{\min}$.
4. Execute the *Flow Decomposition Algorithm* [1] over the link flow $\{f_e\}$, in order obtain a path flow f.
5. **Return** f.

Fig. 10: Algorithm Integral Routing

Our discussion above is summarized by the following theorem, which establishes that Algorithm Integral Routing solves Problem Integral Routing.

Theorem 5: Given an instance $<G,\{s,t\},\{c_e\},\gamma,K>$ of Problem Integral Routing, Algorithm Integral Routing returns a γ/K-integral path flow that transfers at least γ flow units from s to t, such that the network congestion factor is minimized.

Proof: Consider Algorithm Integral Routing (Fig. 10). By construction, the value $\alpha_{\min}$, identified in step (3) of the algorithm satisfies $\textbf{Test}\left(G,\{s,t\},\{c_e\},\gamma,K,\alpha_{\min}\right) \neq \text{FAIL}$. Hence, by the construction of Procedure Test, the link flow that Procedure Test returns for the success with $\alpha_{\min}$ transfers γ flow units from s to t; moreover, since we have shown in the proof of Theorem 4 that this link flow is γ/K-integral, the Flow Decomposition Algorithm of step 4 decomposes it into a γ/K-integral path flow.

It remains to be shown that $\alpha_{\min}$ is the optimal network congestion factor for the given instance. However, this is obvious, since $\alpha_{\min}$ is defined to be the smallest $\alpha \in \overline{\alpha}$ such that $\textbf{Test}\left(G,\{s,t\},\{c_e\},\gamma,K,\alpha\right) \neq \textbf{FAIL}$. Therefore, it follows

from Theorem 4 that $\alpha_{\min}$ equals to the smallest $\alpha \in \overline{\alpha}$ such that $\alpha \geq \alpha^*$, where α^* is the optimal network congestion factor of the given instance. However, since we established in Lemma 2 that $\alpha^* \in \overline{\alpha}$, it follows that $\alpha_{\min} = \alpha^*$. ■

We now establish the computational complexity of Algorithm Integral Routing. As was shown in Remark 4, step 1 of the algorithm consumes $O(M^2 \cdot \log N)$ operations. Step 2 consumes $O\left(T(N,M) \cdot \log\left|\overline{\alpha}\right|\right) = O\left(T(N,M) \cdot \log N\right)$, where $T(N,M)$ is the computational complexity of the *Push Relabel Algorithm* that Procedure Test employs for an N-node M-link network. Step 3 consumes O(1) operations. The Flow Decomposition Algorithm executed in step 4 consumes $O(N \cdot (M+N))$ operations [1]. Finally, since the representation of a path flow f consists of $O(M \cdot N \cdot \log N)$ bits[1], step 5 consumes $O(M \cdot N \cdot \log N)$ operations. Thus we conclude that, since $G(V,E)$ is connected (i.e., $M \geq N-1$), the overall complexity of Algorithm Integral Routing is $O(M^2 \cdot \log N + T(N,M) \cdot \log N)$. We note that the Push Relabel Algorithm has an implementation with a running time of $O(M \cdot N \cdot \log N)$ [13]. With this implementation, the complexity of Algorithm Integral Routing is $O(M \cdot \log N \cdot (M + N \cdot \log N))$.

4.3. A 2-Approximation Scheme for Problem KPR

Finally, we are ready to establish a solution for Problem KPR. To that end, we show that the solution of the Integral Routing Problem can be used in order to establish a *constant approximation scheme* for Problem KPR. The approximation scheme is based on the following key observation, which links between the optimal solution of Problem Integral Routing and the optimal solution of Problem KPR.

Theorem 6: Given are a network $G(V,E)$ and a demand of γ flow units that has to be routed from s to t. If f_1 is a γ/K-integral path flow that has the minimum network congestion factor and f_2 is a path flow that minimizes its network congestion factor while routing along at most K paths, then the network congestion factor of f_1 is at most twice the network congestion factor of f_2.

Proof: Suppose that f_1 and f_2 satisfy the assumptions of the Theorem. Let α_1 and α_2 denote the network congestion factor of path flows f_1 and f_2, respectively. We have to show that $\alpha_1 \leq 2 \cdot \alpha_2$.

Out of the path flow f_2, we construct a γ/K-integral path flow that ships at least γ flow units from s to t and has a network congestion factor of at most $2 \cdot \alpha_2$. Clearly, such a construction implies that the network congestion factor of every *optimal* γ/K-integral path flow that ships γ flow units from s to t is at most $2 \cdot \alpha_2$; in particular, since f_1 is one such optimal γ/K-integral path flow, such a construction establishes that $\alpha_1 \leq 2 \cdot \alpha_2$.

With this goal in mind, define the following construction. First, double the flow along each routing path that f_2 employs; obviously, the resulting path flow transfers $2 \cdot \gamma$ flow units from s to t along at most K routing paths while yielding a network congestion factor of $2 \cdot \alpha_2$. Then, *round down* the (doubled) flow along each routing path to the nearest multiple of γ/K; in this process, the flow along each path is reduced by at most γ/K flow units. Hence, since there are no more than K routing paths, the total flow from s to t is reduced by at most γ units; therefore, since before the rounding operation exactly $2 \cdot \gamma$ flow units were shipped from s to t, it follows that after the rounding is performed, the resulting path flow transfers at least γ flow units from s to t.

Thus, we have identified a γ/K-integral path flow that transfers at least γ flow units from s to t. In addition, since prior to the rounding operation the network congestion factor is $2 \cdot \alpha_2$ and the rounding can only reduce flow, the network congestion factor of the constructed path flow is at most $2 \cdot \alpha_2$. ■

Note that, given a network $G(V,E)$ and a demand γ that needs to be routed over at most K paths, *every* γ/K-integral path flow satisfies the requirement to ship the demand γ on at most K different paths. On the other hand, it has been established in Theorem 6 that the network congestion factor obtained by an optimal γ/K-integral path flow is at most twice the network congestion factor of an optimal flow that admits at most K routing paths. Thus, computing a γ/K-integral path flow that has the minimum network congestion factor satisfies the restriction on the number of routing paths and obtains a network congestion factor that is at most twice larger than the optimum. We summarize the above discussion in the following corollary, which yields an approximation scheme for Problem KPR.

Corollary 1: Given are a network $G(V,E)$, a demand γ and a restriction on the number of routing paths K. The employment of Algorithm Integral Routing for the establishment of a γ/K-integral path flow that minimizes the network congestion factor provides a 2-approximation scheme for Problem KPR with a complexity of $O(M \cdot \log N \cdot (M + N \cdot \log N))$.

5. Simulation Results

The goal of this section is to demonstrate *how much* is gained by employing optimal multipath routing algorithms for congestion minimization. To that end, we present a comparison between the congestion obtained by an optimal multipath routing algorithm to the congestion obtained by the popular heuristics ECMP and OMP. Through comprehensive simulations we show that multipath routing solutions obtained by optimal congestion reduction schemes are fundamentally more efficient. We generated two classes of random networks: *Power-Law* topologies [11] and *Waxman* topologies [25]. For each class, we generated 10,000 networks and conducted the following measurements over each network: (a) the network congestion factor produced by invoking ECMP; (b) the network congestion factor produced by invoking OMP; (c) the network congestion factor produced by an optimal assignment of traffic to shortest paths and to paths with a length that is equal to

[1] f consists O(M) routing paths, each path consists of O(N) links, and each link is represented by O(log$N$) bits. Therefore, the collection of all pairs $(p,f(p))$ such that $f(p) \neq 0$ consists of $O(MN \cdot \log N)$ bits.

$1.1 \cdot L^*$, $1.2 \cdot L^*$, $1.3 \cdot L^*$, $1.4 \cdot L^*$, $1.5 \cdot L^*$, $1.6 \cdot L^*$, $1.7 \cdot L^*$, $1.8 \cdot L^*$, $1.9 \cdot L^*$ and $2 \cdot L^*$, where L^* is the length of a shortest path. In all runs, we assumed that the link capacities are uniformly distributed in [5,150] MB/sec and the length of each link is uniformly distributed in [1, 50].

We turn to specify the way we generated each class of topologies, starting with Waxman topologies. Our construction follows the lines of [25]. We first located the source and the destination at the diagonally opposite corners of a square area of unit dimension. Then, we randomly spread 198 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 $\alpha = 2$ and $\beta = 0.2$. The above approach resulted in 200 nodes and approximately 1800 links per network topology.

We turn to specify the way we generated power-law topologies. Our construction followed the lines of [11]. First, we randomly assigned a certain number of *out-degree credits* to each node, using the power-law distribution $\beta \cdot x^{-\alpha}$, where $\alpha = 0.3$ and $\beta = 3$. Then, we connected the nodes so that every node obtained the assigned out-degree. More specifically, we randomly picked a pair of nodes *u* and *v*, and assigned a directed link from *u* to *v* if *u* had some remaining out-degree credits and link $u \to v$ had not been defined already. Whenever a link $u \to v$ was placed between the corresponding nodes, we also decremented the out-degree credit of node *u*. When the selected pair of nodes was not suitable for a link, we continued to pick pairs of nodes until finding one that was suitable. The above strategy resulted in 200 nodes and approximately 1200 links per network topology.

Our results are summarized in Figures 11 and 12. Note that, for power-law topologies, if the ECMP or OMP heuristics had

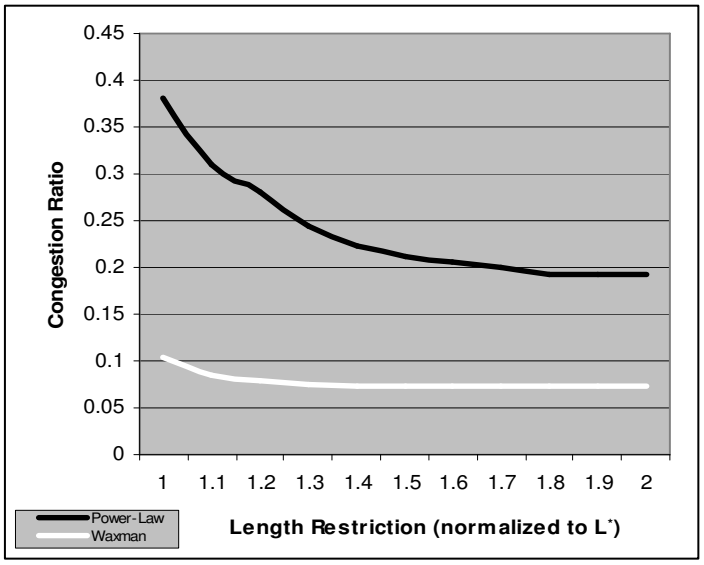

Fig. 11. The ratio between the network congestion produced by an optimal multipath routing assignment (for several length restrictions) and the network congestion produced by ECMP.

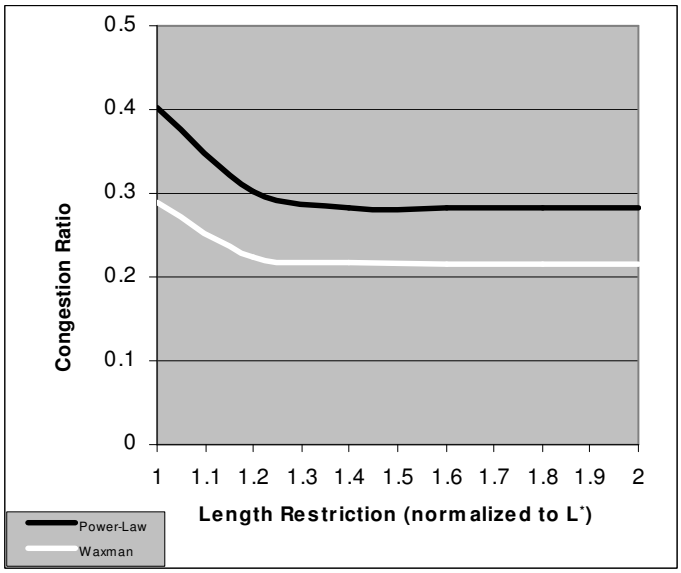

Fig. 12. The ratio between the network congestion produced by an optimal multipath routing assignment (for several length restrictions) and the network congestion produced by OMP.

an optimal mechanism to distribute traffic among the shortest paths, the network congestion factor would have been reduced by a factor of at least 2.5; moreover, for Waxman topologies, this factor of improvement is greater than 3 for OMP and greater than 9.5 for ECMP. Next, observe that optimal traffic distribution mechanisms that allow relaxing the requirement to route along shortest paths by just 20% produce a network congestion factor that, in power-law topologies, is at least 3.3 times smaller than with OMP and ECMP; moreover, for Waxman topologies, this factor of improvement is more than 4.5 for OMP and is more than 12 for ECMP. Thus, for $L \approx 1.2 \cdot L^*$, Algorithm RMP or its ε-optimal approximation can drastically reduce network congestion at the price of routing along paths that are just slightly longer than the shortest.

6. Conclusion

Previous multipath routing schemes for congestion avoidance focused on heuristic methods. Yet, our simulations indicate that optimal congestion reduction schemes are *significantly* more efficient. Accordingly, we investigated multipath routing as an optimization problem of minimizing network congestion, and considered two fundamental problems. Although both have been shown to be computationally intractable, they have been found to admit efficient approximation schemes. Indeed, for each problem, we have established a *polynomial time* algorithm that approximates the optimal solution by a (small) *constant* approximation factor.

A common feature that both approximations share is the discretization of the set of feasible solutions. Whereas the solution to Problem KPR is established by restricting the flow along each path to be integral in γ/K, the solution to Problem RMP is established by restricting all lengths to be integral in some common scaling factor. These discretizations enable to

reduce the space of feasible solutions and therefore obtain polynomial running time algorithms.

While this study has laid the algorithmic foundations of two fundamental multipath routing problems, there are still many challenges to overcome. One major challenge is to establish an efficient unifying scheme that combines the two problems. Furthermore, as in practice there may be a need for simpler solutions, another research challenge is the development of approximations with lower computational complexity. Finally, as discussed in [4], multipath routing offers a rich ground for research also in other contexts, such as survivability, recovery, network security and energy efficiency. We are currently working on these issues and have obtained several results regarding survivability [5].

References

[1] R. K. Ahuja, T. L. Magnanti and J. B. Orlin, "Network Flows: Theory, Algorithm, and Applications", Prentice Hall, 1993.

[2] D. Awduche, J. Malcolm, M. O'Dell, and J. McManus, "Requirements for traffic engineering over MPLS" , Internet Draft, April, 1998.

[3] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell and J. McManus, "Requirements for Traffic Engineering Over MPLS", IETF RFC 2702, September 1999.

[4] R. Banner and A. Orda, "Multipath Routing Algorithms for Congestion Minimization", CCIT Report No. 429, Department of Electrical Engineering, Technion, Haifa, Israel, 2005. Available from: http://www.ee.technion.ac.il/people/ron/Congestion.pdf

[5] R. Banner and A. Orda, "The Power of Tuning: a Novel Approach for the Efficient Design of Survivable Networks", In Proc. IEEE ICNP 2004.

[6] D. Bertsekas and R. Gallager, "Data networks", Prentice-Hall, 1992.

[7] Allan Borodin and Ran El-Yaniv, "Online Computation and Competative Analysis", Cambridge University Press, 1998.

[8] I. Cidon, R. Rom and Y. Shavitt, "Analysis of Multi-Path Routing", IEEE Transactions on Networking, v. 7, No. 6, pp. 885–896, 1999.

[9] S. Chen and K. Nahrstedt, "An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions", IEEE Network, v. 12, No. 6, pp. 64-79, December 1998.

[10] T. Cormen, C. Leiserson, and R. Rivest, "Introduction to Algorithms", Cambridge, MA: The MIT Press, 2001.

[11] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On Power-law Relationships of the Internet Topology", in Proceedings of ACM SIGCOMM, cambridge, MA, September 1999.

[12] M. R. Garey and D. S. Johnson, "Computers and Intractability", W.H. Freeman and Co., 1979.

[13] A. V. Goldberg and R. E. Tarjan, "A New Approach to The Maximum Flow Problem", Journal of ACM, v.35, No. 4, pp. 921-940, 1988.

[14] S. Iyer, S. Bhattacharyya, N. Taft, N. McKeoen, C. Diot, "A measurement Based Study of Load Balancing in an IP Backbone", Sprint ATL Technical Report, TR02-ATL-051027, May 2002.

[15] Y. Jia, I. Nikolaidis and P. Gburzynski, "Multiple Path QoS Routing", Proceedings of ICC'01 Finland, pp. 2583-2587, June 2001.

[16] N. Karmarkar, "A New Polynomial-Time Algorithm For Linear Programming", Combinatorica, vol. 4, pp. 373-395, 1984.

[17] J. Kleinberg, "Single-Source Unsplittable Flow," In Proc. 37th IEEE FOCS, 1996.

[18] J.Moy, "OSPF Version 2", IETF RFC 2328, April 1998.

[19] S. Nelakuditi and Zhi-Li Zhang, "On Selection of Paths for Multipath Routing", In Proc. IWQoS, Karlsruhe, Germany, 2001.

[20] V. Paxson, "End-to-End Routing Behavior in the Internet", In Proc. ACM SIGCOMM, 1996.

[21] E. Rosen, A. Viswanathan, and R. Callon. "Multiprotocol Label Switching Architecture". IETF RFC 3031, 2001.

[22] D. Thaler and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", IETF RFC 2991, 2000.

[23] C. Villamizar, "OSPF Optimised Multipath (OSPF-OMP)", Internet Draft, 1998.

[24] Y. Wang and Z. Wang, "Explicit Routing Algorithms For Internet Traffic Engineering", In Proc. ICCN'99, Boston, October 1999.

[25] B. M. Waxman, "Routing of Multipoint Connections", IEEE Journal on Selected Areas in Communications, 6:1617-1622,1988.

[26] A. E. I. Widjaja, "Mate: MPLS Adaptive Traffic Engineering", Internet Draft, August, 1998.

Appendix

In this appendix, we show that the problem of minimizing the network congestion factor is equivalent to the well-known Maximum Flow Problem. To that end, we prove the following Theorem.

Theorem A.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 $\gamma \le F$, the link flow $\{f_e\}_{e\in E}$ is a maximum flow *iff* the link flow $\left\{\frac{\gamma}{F}\cdot f_e\right\}_{e\in E}$ transfers γ flow units from s to t 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 $\left\{\frac{\gamma}{F}\cdot f_e\right\}_{e\in E}$ must also satisfy these constraints. In addition, since $\{f_e\}$ transfers F flow units from s to t (i.e., $\sum_{e\in O(s)} f_e - \sum_{e\in I(s)} f_e = F$), then it follows that $\sum_{e\in O(s)} \frac{\gamma}{F}\cdot f_e - \sum_{e\in I(s)} \frac{\gamma}{F}\cdot f_e = \frac{\gamma}{F}\cdot\left(\sum_{e\in O(s)} f_e - \sum_{e\in I(s)} f_e\right) = \frac{\gamma}{F}\cdot F = \gamma$. 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 $\left\{\widetilde{f}_e\right\}_{e\in E}$ that transfers γ flow units from s to t such that $\tilde{\alpha} = \max_{e\in E}\left\{\frac{\widetilde{f}_e}{c_e}\right\} < \frac{\gamma}{F}$. Consider the link flow $\left\{\frac{1}{\tilde{\alpha}}\cdot\widetilde{f}_e\right\}_{e\in E}$. Since $\tilde{\alpha} = \max_{e\in E}\left\{\frac{\widetilde{f}_e}{c_e}\right\}$, it holds that $\max_{e\in E}\left\{\frac{\frac{1}{\tilde{\alpha}}\cdot\widetilde{f}_e}{c_e}\right\} = \frac{1}{\tilde{\alpha}}\cdot\max_{e\in E}\left\{\frac{\widetilde{f}_e}{c_e}\right\} = 1$; hence the capacity constraints are satisfied for $\left\{\frac{1}{\tilde{\alpha}}\cdot\widetilde{f}_e\right\}_{e\in E}$. Moreover, it is given

that $\sum_{e\in O(s)}\widetilde{f}_e-\sum_{e\in I(s)}\widetilde{f}_e=\gamma$; hence, $\sum_{e\in O(s)}\frac{1}{\widetilde{\alpha}}\cdot\widetilde{f}_e-\sum_{e\in I(s)}\frac{1}{\widetilde{\alpha}}\cdot\widetilde{f}_e=$ $=\frac{1}{\widetilde{\alpha}}\cdot\left(\sum_{e\in O(s)}\widetilde{f}_e-\sum_{e\in I(s)}\widetilde{f}_e\right)=\frac{\gamma}{\widetilde{\alpha}}>\frac{\gamma}{\gamma/F}=F$. Thus, link flow $\left\{\frac{1}{\widetilde{\alpha}}\cdot\widetilde{f}_e\right\}_{e\in E}$ 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 $\left\{\frac{\gamma}{F}\cdot f_e\right\}_{e\in E}$ 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 $\left\{\frac{\gamma}{F}\cdot f_e\right\}_{e\in E}$ satisfies the flow conservation constraints it follows that $\{f_e\}_{e\in E}$ also satisfies the flow conservation constraints. Moreover, since $\left\{\frac{\gamma}{F}\cdot f_e\right\}_{e\in E}$ transfers γ flow units from s to t, $\sum_{e\in O(s)}f_e-\sum_{e\in I(s)}f_e=\frac{F}{\gamma}\cdot\left(\sum_{e\in O(s)}\frac{\gamma}{F}\cdot f_e-\sum_{e\in I(s)}\frac{\gamma}{F}\cdot f_e\right)=\frac{F}{\gamma}\cdot\gamma=F$ 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 $\left\{\frac{\gamma}{F}\cdot f_e\right\}_{e\in E}$ is at most $\frac{\gamma}{F}$. Hence, since we assume that $\left\{\frac{\gamma}{F}\cdot f_e\right\}_{e\in E}$ 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 $\left\{\widetilde{f}_e\right\}_{e\in E}$ is a maximum link flow from s to t, then $\left\{\frac{\gamma}{F}\cdot\widetilde{f}_e\right\}_{e\in E}$ 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 $\left\{\widetilde{f}_e\right\}_{e\in E}$, we first show that $\left\{\frac{\gamma}{F}\cdot\widetilde{f}_e\right\}_{e\in E}$ has a network congestion factor of at most $\frac{\gamma}{F}$. Since $\left\{\widetilde{f}_e\right\}_{e\in E}$ must satisfy the capacity constraints, it holds that $\widetilde{f}_e\le c_e$ for each $e\in E$; hence, $\frac{\widetilde{f}_e}{c_e}\le 1$ for each $e\in E$ and, in particular, $\max_{e\in E}\left\{\frac{\widetilde{f}_e}{c_e}\right\}\le 1$. Thus, the network congestion factor of $\left\{\frac{\gamma}{F}\cdot\widetilde{f}_e\right\}_{e\in E}$ is $\max_{e\in E}\left\{\frac{\frac{\gamma}{F}\widetilde{f}_e}{c_e}\right\}=\frac{\gamma}{F}\cdot\max_{e\in E}\left\{\frac{\widetilde{f}_e}{c_e}\right\}\le\frac{\gamma}{F}$. We turn to show that $\left\{\frac{\gamma}{F}\cdot\widetilde{f}_e\right\}_{e\in E}$ transfers γ flow units from s to t. Since $\left\{\widetilde{f}_e\right\}_{e\in E}$ transfers F flow units from s to t, it holds that $\sum_{e\in O(s)}\frac{\gamma}{F}\cdot\widetilde{f}_e-\sum_{e\in I(s)}\frac{\gamma}{F}\cdot\widetilde{f}_e=\frac{\gamma}{F}\cdot\left(\sum_{e\in O(s)}\widetilde{f}_e-\sum_{e\in I(s)}\widetilde{f}_e\right)=\frac{\gamma}{F}\cdot F=\gamma$ i.e., $\left\{\frac{\gamma}{F}\cdot\widetilde{f}_e\right\}_{e\in E}$ ships γ flow units from s to t. Finally, it is easy to see that since the conservation constraints are satisfied for $\left\{\widetilde{f}_e\right\}_{e\in E}$ they are also satisfied for $\left\{\frac{\gamma}{F}\cdot\widetilde{f}_e\right\}_{e\in E}$; moreover, since we assume that $\gamma\le F$ and we have shown that the network congestion factor of $\left\{\frac{\gamma}{F}\cdot\widetilde{f}_e\right\}_{e\in E}$ is at most $\frac{\gamma}{F}$, it holds that $\left\{\frac{\gamma}{F}\cdot\widetilde{f}_e\right\}_{e\in E}$ satisfies the capacity constraints. ■