CCIT Report # 462
January 2004

Online Multipath Routing

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

We consider the online problem of routing traffic in order to minimize network congestion in settings where demands are allowed to be splitted *along any number of paths*. Previous works in this context focused on congestion minimization schemes that limit the traffic of each request to travel along a single path. We describe a multipath routing algorithm for congestion minimization with an $O(\log N)$ competitive ratio, where N is the number of nodes in the network. We also show that this result is tight i.e., for any online multipath routing algorithm, there is a scenario in which the congestion is larger by a factor of $\Omega(\log N)$ than the (offline) optimum.

1. Introduction

In an online routing problem, demands arrive one at a time and there is no *a priori* knowledge regarding future demands. Each demand specifies the source and destination nodes and the requested bandwidth. Upon the arrival of a new request, the algorithm establishes a connection by allocating the required bandwidth along some path between the source and destination nodes. The goal of the algorithm is usually set to minimize the congestion of the network [1],[2].

The performance of online algorithms is usually evaluated in terms of the *competitive ratio* introduced by [3] and further developed by [4]. In our case, it corresponds to the supremum, over all possible input sequences, of the ratio between the congestion obtained by the online algorithm and the congestion obtained by the optimal algorithm that is based on the entire input sequence.

To the best of our knowledge, multipath routing has not been considered in the context of online computation. In order to address this issue, we employ the exponential cost functions of [1],[5] that were used thus far in order to route demands in an online fashion over *single paths*. More specifically, the schemes of [1],[5] assign to each link a cost that is exponential in its congestion; then, upon the arrival of a new demand, they compute the shortest path with respect to these exponential costs and route the demand along the resulting path. In this paper, we establish that such cost functions can be used in order to derive a competitive strategy that routes demands in an online fashion over *multipaths*. Roughly speaking, we show that identifying for each new demand a min cost flow with respect to the exponential costs of [1],[5] while restricting the resulting flow to be integral over each link, achieves a network congestion factor larger by $O(\log N)$ than the optimum. Based on this observation, we establish a polynomial online scheme for multipath routing with a competitive ratio of $O(\log N)$. Moreover, we prove that the resulting online routing scheme yields a tight, i.e., best possible, competitive ratio.

2. Model and Problem Formulation

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 $p=(v_0,v_1,\ldots v_h)$, 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 $p=(v_0,v_1,\ldots v_h)$ together with the link $(v_h,v_0)\in E$ i.e., $(v_0,v_1,\ldots v_h,v_0)$. Let $P^{(i,j)}$ denote the collection of all simple paths from the source i to the destination j and let $P \triangleq \bigcup_{(i,j)\in V\times V} P^{(i,j)}$.

Each link $e\in E$ is assigned a *capacity* $c_e\in\mathbb{Z}^+$. We consider a *link state* routing environment, where each source node has a (precise) image of the entire network.

A *commodity* is a pair of nodes $(i,j)\in V\times V$ that is assigned with a non-negative demand $\gamma^{(i,j)}$. 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*. For each node $v\in V$, 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 $I(v)=\{(w,v)|(w,v)\in E\}$.

Definition 0.1 Let $G(V,E)$ be a network. *A flow vector* $f\triangleq\{f_e^{(i,j)}\}$ is a real-valued function $f: E\times V\times V\rightarrow\mathbb{R}^+\cup\{0\}$ that satisfies the following two properties:
Flow conservation: For each commodity $(i,j)\in V\times V$ and each node other than the source i and the destination j $\sum_{e\in O(v)} f_e^{(i,j)} - \sum_{e\in I(v)} f_e^{(i,j)} = 0$.
Flow demand: For each commodity $(i,j)\in V\times V$, $\sum_{e\in O(i)} f_e^{(i,j)} - \sum_{e\in I(i)} f_e^{(i,j)} = \gamma^{(i,j)}$.

Definition 2.2: Given are a network $G(V,E)$, a flow vector $f=\{f_e^{(i,j)}\}$ and a commodity $(i,j)\in V\times V$. The *link flow of commodity* (i,j) is the collection $\{f_e^{(i,j)}|e\in E\}$.

Definition 2.3: Given are a network $G(V,E)$, a flow vector $f=\{f_e^{(i,j)}\}$, a commodity $(i,j)\in V\times V$, and a value $\sigma\in\mathbb{R}^+$. The link flow of commodity (i,j) is $\sigma-$*integral*, if for each link $e\in E$ it holds that $f_e^{(i,j)}$ is a multiple of σ.

Definition 2.4: Given are a network *G*(*V*,*E*), and, for each $(i,j)\in V\times V$ a demand $\gamma^{(i,j)}$. A *path flow* is a real-valued function $f{:}P\rightarrow \mathbb{R}^{+}\cup\{0\}$ that satisfies $\sum_{p\in P^{(i,j)}} f(p)=\gamma^{(i,j)}$ for each $(i,j)\in V\times V$.

Definition 2.5: Given are a network *G*(*V*,*E*) a capacity $c_e>0$ for each $e\in E$, and a flow vector $f=\left\{f_e^{(i,j)}\right\}$. Define $f_e \triangleq \sum_{(i,j)\in V\times V} f_e^{(i,j)}$ for each $e\in E$. The value $\frac{f_e}{c_e}$ is the *link congestion factor*.

Definition 2.6: 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 [6],[7],[8], the network congestion factor provides a good indication of congestion.

In order to evaluate the quality of algorithms in the context of on-line problems, we employ the following standard terminology [3].

Definition 2.7 A solution is *off-line* if it is based on the entire input sequence.

Definition 2.8 The *competitive ratio* of a given online algorithm is defined as the supremum over all input sequences of the performance achieved by the optimum offline algorithm and the performance achieved by this online algorithm.

We proceed to present the problem that considered in this paper. We are given a network and requests that arrive in an online fashion, each with a specified flow demand, source-destination pairs, and the maximum number of paths over which the given demand is allowed to be shipped. The goal is to route each request such that the network congestion factor is minimized. This is formulated as follows.

Problem OMR (On-Line Multipath Routing) Given are a network $G(V,E)$, and a capacity $c_e>0$ for each link $e\in E$. Requests arrive as triples $R(k)\triangleq\left\langle (s,t)_k, \mathrm{K}_k, \gamma_k\right\rangle$ in an on-line fashion. The *k*-th request $R(k)$ is satisfied by transferring γ_k flow units from *s* to *t* over at most K_k paths. For each index *k*, upon the arrival of the *k*-th request $R(k)$, find an assignment that minimizes the network congestion factor while satisfying $R(k)$.

We call the triple $\left\langle G(V,E),\{c_e\},\{R(k)\}\right\rangle$ an *instance*.

3. A Competitive Ratio of $O(\log N)$ for Problem OMR

In this section, we establish a polynomial scheme for Problem OMR with a competitive ratio of $O(\log N)$. Our scheme is based on solving an auxiliary problem that introduces an additional integrality restriction to Problem OMR.

Problem I-OMR (Integral OMR) Given are a network $G(V,E)$, and a capacity $c_e > 0$ for each link $e \in E$. Requests arrive as triples $R(k) = \langle (s,t)_k, \mathrm{K}_k, \gamma_k \rangle$ in an on-line fashion. The *k*-th request $R(k)$ is satisfied by identifying a $\frac{\gamma_k}{\mathrm{K}_k}$ – integral path flow that transfers γ_k flow units from *s* to *t*. For each index *k*, upon the arrival of the *k*-th request $R(k)$, find an assignment that minimizes the network congestion factor while satisfying $R(k)$.

Note that problem I-OMR implicitly limits the number of paths for the *k*-th request to at most K_k paths, by restricting the flow to be $\frac{\gamma_k}{\mathrm{K}_k}$ – integral. Thus, Problem I-OMR differs from Problem OMR only in the restriction to have a solution (path flow) that is integral in $\frac{\gamma_k}{\mathrm{K}_k}$ for each request $R(k)$.

3.1 Solving Problem I-OMR

Problem I-OMR is solved using Procedure I-OMR that is specified in Fig.1. The procedure is given a network $G(V,E)$, a network congestion factor α and a request $R(k)$. The procedure assigns to each link $e \in E$ a cost that is exponential in $\frac{x_e}{\alpha \cdot c_e}$ where x_e is the flow that traverses through link e. Then, the procedure computes a $\frac{\gamma_k}{\mathrm{K}_k}$ – integral min-cost flow with respect to these exponential costs. We prove that, if the given network congestion factor α is at least the network congestion factor of the optimal offline solution (that considers *all k* requests ahead), then this min-cost flow computation provides a path flow with a network congestion factor of at most $\alpha \cdot \log(2 \cdot m)$. Thus, we establish that, if we *know in advance* the optimal offline network congestion factor, then our procedure derives a solution for Problem I-OMR with a competitive ratio of $\log(2 \cdot M) = \log(2 \cdot N^2) = O(\log N)$. In order to provide the specification of that procedure, consider first the following notations.

Given is an instance $\langle G(V,E), \{c_e\}, \{R(k)\} \rangle$ of Problem I-OMR. Let $\langle \{f_e^*(1)\}, \{f_e^*(2)\}, \ldots, \{f_e^*(k)\} \rangle$ be the link flows assigned for requests

$R(1),\dots,R(k)$ by the optimal offline algorithm, and let $\langle\{f_e(1)\},\{f_e(2)\},\dots,\{f_e(k)\}\rangle$ be the link flows assigned by Procedure I-OMR that is specified in Fig. 1. In addition, for each $1\le n\le k$ and each $e\in E$, let $s_e(n)\triangleq\sum_{m=1}^{n}f_e(m)$ and let $s_e^*(n)\triangleq\sum_{m=1}^{n}f_e^*(m)$. We denote by $\{s_e(k)\}$ the link flow that results from the assignment of the first *k* requests by Procedure I-OMR; similarly, we denote by $\{s_e^*(k)\}$ the optimal link flow that consider the first *k* requests.

Definition 3.1 Given an instance $\langle G(V,E),\{c_e\},\{R(k)\}\rangle$ of Problem I-OMR, a network congestion factor $\alpha>0$ and a link flow $\{x_e(k)\}$ for request $R(k)$, denote for each $e\in E$ the exponential cost $\left(\frac{3}{2}\right)^{\frac{s_e(k-1)+x_e(k)}{\alpha\cdot c_e}}-\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}$ as $p_e(x_e(k))$, i.e.,

$$p_e(x_e(k))\triangleq\left(\frac{3}{2}\right)^{\frac{s_e(k-1)+x_e(k)}{\alpha\cdot c_e}}-\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}.$$

Finally, we define the following problem that Procedure I-OMR solves upon the arrival of each new request. We note that the problem is a special case of the *convex cost flow problem* [9].

Problem CCF (Convex Cost Flow Problem) Given are a network $G(V,E)$, a capacity $c_e>0$ for each link $e\in E$, a link flow $\{s_e(k-1)\}$ that consider the first *k*-1 requests and a new request $R(k)=\langle(s,t)_k,\mathrm{K}_k,\gamma_k\rangle$. Find a $\frac{\gamma_k}{K_k}-$ integral link flow $\{x_e(k)\}$ that transfer γ_k flow units from *s* to *t* such that the total cost $\sum_{e\in E}p_e(x_e(k))$ is minimized.

Unlike general convex cost flow problems that have no finite-time algorithms [10], convex cost flow problems (as Problem CCF) that impose integrality restriction on link flows can admit a polynomial solution. In particular, Problem CCF can be applied by the *capacity scaling algorithm* for convex cost flows [9] that has a polynomial running time. Fig. 1 specifies Procedure I-OMR. This procedure solves an instance of Problem CCF by employing the *capacity scaling algorithm* for convex cost flows [9].

Procedure I-OMR $\left(G(V,E),\{c_e\},\{s_e(k-1)\},R(k),\alpha\right)$

Parameters:

$G(V,E)$ – network

$\{c_e\}$ – capacities

$\{s_e(k-1)\}$ – the link flow of all previous k -1 requests

$R(k)$ - the k-th of establishing a $\frac{\gamma_k}{K_k}$-integral link flow of γ_k flow units from s_k to t_k

α – the restriction on network congestion factor

1. Use the *capacity scaling algorithm* [9] in order to solve the instance $\left\langle G(V,E),\{p_e(x_e(k))\},(s,t)_k,\gamma_k\right\rangle$ of Problem CCF. Let $f_e(k)$ be the resulting link flow.
2. If there exists $e\in E$ such that $\frac{s_e(k-1)+f_e(k)}{c_e}>\alpha\cdot\log_{\frac{3}{2}}(2\cdot m)$.

Return Fail

Else

Return link flow $\{x_e(k)\}$.

Fig. 1 Procedure I-OMR

Definition 3.2 Given an instance of Problem I-OMR, α_k^* is the network congestion factor of the optimal offline solution that considers the first k requests i.e., $\alpha_k^*\triangleq\max_{e\in E}\left\{\frac{s_e^*(k)}{c_e}\right\}$.

Theorem 1 Given are an instance $\left\langle G(V,E),\{c_e\},\{R(k)\}\right\rangle$ of Problem I-OMR and the corresponding input $\left\langle G(V,E),\{c_e\},\{s_e(k-1)\},R(k),\alpha\right\rangle$ for Procedure I-OMR. If $\alpha_k^*\le\alpha$, then Procedure I-OMR never fails. Thus, the returned path flow has a network congestion factor of at most $\alpha\cdot\log_{\frac{3}{2}}(2\cdot m)$.

Proof Consider the following potential function $\Phi(k)=\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k)}{\alpha\cdot c_e}}\left(2-\frac{\alpha_k^*}{\alpha}\right)$. We will first prove that $\Phi(k)-\Phi(k-1)\le 0$.

$$
\begin{aligned}
\Phi(k)-\Phi(k-1) &= \sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k)}{\alpha\cdot c_e}}\left(2-\frac{\alpha_k^*}{\alpha}\right)-\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\left(2-\frac{\alpha_{k-1}^*}{\alpha}\right)= \\
&= \sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k)}{\alpha\cdot c_e}}\cdot\left[2-\frac{1}{\alpha}\left(\frac{s_e^*(k-1)+f_e^*(k)}{c_e}\right)\right]-\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\cdot\left(2-\frac{\alpha_{k-1}^*}{\alpha}\right)= \\
&= \sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k)}{\alpha\cdot c_e}}\cdot\left[2-\frac{1}{\alpha}\left(\alpha_{k-1}^*+\frac{f_e^*(k)}{c_e}\right)\right]-\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\cdot\left(2-\frac{\alpha_{k-1}^*}{\alpha}\right)= \\
&= \sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k)}{\alpha\cdot c_e}}\cdot\left[2-\frac{\alpha_{k-1}^*}{\alpha}-\frac{f_e^*(k)}{\alpha\cdot c_e}\right]-\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\cdot\left(2-\frac{\alpha_{k-1}^*}{\alpha}\right)= \\
&= \left(2-\frac{\alpha_{k-1}^*}{\alpha}\right)\cdot\sum_{e\in E}\left(\left(\frac{3}{2}\right)^{\frac{s_e(k)}{\alpha\cdot c_e}}-\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\right)-\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k)}{\alpha\cdot c_e}}\cdot\frac{f_e^*(k)}{\alpha\cdot c_e}\le \\
&\le 2\cdot\sum_{e\in E}\left(\left(\frac{3}{2}\right)^{\frac{s_e(k)}{\alpha\cdot c_e}}-\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\right)-\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\cdot\frac{f_e^*(k)}{\alpha\cdot c_e}= \\
&= 2\cdot\sum_{e\in E}\left(\left(\frac{3}{2}\right)^{\frac{s_e(k-1)+f_e(k)}{\alpha\cdot c_e}}-\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\right)-\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\cdot\frac{f_e^*(k)}{\alpha\cdot c_e}.
\end{aligned}
$$

Since the capacity scaling algorithm, which solves Problem CCF in step (1), identifies a $\frac{\gamma_k}{\mathrm{K}_k}-$link flow $\{f_e(k)\}$ that *minimizes* the total cost $\sum_{e\in E} p_e(\cdot)$, it follows that $\sum_{e\in E} p_e(f_e(k)) \le \sum_{e\in E} p_e(f_e^*(k))$. Thus, by definition, $\sum_{e\in E}\left(\left(\frac{3}{2}\right)^{\frac{s_e(k-1)+f_e(k)}{\alpha\cdot c_e}}-\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\right)\le\sum_{e\in E}\left(\left(\frac{3}{2}\right)^{\frac{s_e(k-1)+f_e^*(k)}{\alpha\cdot c_e}}-\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\right)$. Therefore, we employ this inequality as follows.

$$
\begin{aligned}
\Phi(k)-\Phi(k-1) &\le 2\cdot\sum_{e\in E}\left(\left(\frac{3}{2}\right)^{\frac{s_e(k-1)+f_e(k)}{\alpha\cdot c_e}}-\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\right)-\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\cdot\frac{f_e^*(k)}{\alpha\cdot c_e}\le \\
&\le 2\cdot\sum_{e\in E}\left(\left(\frac{3}{2}\right)^{\frac{s_e(k-1)+f_e^*(k)}{\alpha\cdot c_e}}-\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\right)-\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\cdot\frac{f_e^*(k)}{\alpha\cdot c_e}= \\
&= 2\cdot\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\cdot\left(\left(\frac{3}{2}\right)^{\frac{f_e^*(k)}{\alpha\cdot c_e}}-1-\frac{f_e^*(k)}{2\cdot\alpha\cdot c_e}\right).
\end{aligned}
$$

In order to show that $2\cdot\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k-1)}{\alpha\cdot c_e}}\cdot\left(\left(\frac{3}{2}\right)^{\frac{f_e^*(k)}{\alpha\cdot c_e}}-1-\frac{f_e^*(k)}{2\cdot\alpha\cdot c_e}\right)\le 0$, we prove that, for each $e\in E$, $\left(\frac{3}{2}\right)^{\frac{f_e^*(k)}{\alpha\cdot c_e}}-1-\frac{f_e^*(k)}{2\cdot\alpha\cdot c_e}\le 0$. Observe that, since it is given that $\alpha_k^*\le\alpha$, it follows that $\frac{f_e^*(k)}{c_e}\le\frac{s_e^*(k-1)+f_e^*(k)}{c_e}=\frac{s_e^*(k)}{c_e}\le\alpha_k^*\le\alpha$ for each $e\in E$. Therefore, it holds that $\frac{f_e^*(k)}{\alpha\cdot c_e}\in[0,1]$ for each $e\in E$. Thus, we only have to prove that

(i) $\left(\frac{3}{2}\right)^x-1-\frac{x}{2}\le 0$ for every x in the range $[0,1]$.

Reference [1] states that if $a,\gamma>1$ and $a=1+{}^{1}\!/_{\gamma}$ then it follows that $\gamma\left(a^x-1\right)\le x$ for each $x\in[0,1]$. Thus, since both constants $\gamma=2, a=\frac{3}{2}$ are larger than 1 and satisfy $a=1+{}^{1}\!/_{\gamma}$, it follows that $2\left(\left(\frac{3}{2}\right)^x-1\right)\le x$ for each $x\in[0,1]$. Obviously, this validates (i), thus we have established that $\Phi(k)-\Phi(k-1)\le 0$.

We use this property in order to prove the Theorem, i.e., show that if $\alpha_k^*\le\alpha$, then Procedure I-OMRA never fails.

As it is given that $\alpha_k^*\le\alpha$, it follows that $\Phi(k)=\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k)}{\alpha\cdot c_e}}\left(2-\frac{\alpha_k^*}{\alpha}\right)\ge\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k)}{\alpha\cdot c_e}}(2-1)=\sum_{e\in E}\left(\frac{3}{2}\right)^{\frac{s_e(k)}{\alpha\cdot c_e}}\ge\left(\frac{3}{2}\right)^{\max_{e\in E}\left\{\frac{s_e(k)}{\alpha\cdot c_e}\right\}}$. Since $\Phi(k)\le\Phi(0)=2\cdot M$, and since $\Phi(k)\ge\left(\frac{3}{2}\right)^{\max_{e\in E}\left\{\frac{s_e(k)}{\alpha\cdot c_e}\right\}}$ we conclude that $\left(\frac{3}{2}\right)^{\max_{e\in E}\left\{\frac{s_e(k)}{\alpha\cdot c_e}\right\}}\le 2\cdot M$. Therefore, taking the log from both sides we get,

$$\log_{\frac{3}{2}}\left(\left(\frac{3}{2}\right)^{\max_{e\in E}\left\{\frac{s_e(k)}{\alpha\cdot c_e}\right\}}\right)\le\log_{\frac{3}{2}}(2\cdot M)\ \Rightarrow\ \max_{e\in E}\left\{\frac{s_e(k)}{\alpha\cdot c_e}\right\}\le\log_{\frac{3}{2}}(2\cdot M)$$

$$\Rightarrow\max_{e\in E}\left\{\frac{s_e(k)}{c_e}\right\}\le\alpha\cdot\log_{\frac{3}{2}}(2\cdot M).$$

Hence, by construction, Procedure I-OMRA does not fail, thus establishing the Theorem. ■

Since $\alpha \cdot \log_{\frac{3}{2}}(2 \cdot M) \le \alpha \cdot \log_{\frac{3}{2}}(2 \cdot N^2) = O(\alpha \cdot \log N)$, it follows that, given an instance of Problem I-OMR and a network congestion factor α such that $\alpha_k^* \le \alpha$, Procedure I-OMR can be used in order to output a link flow with a network congestion factor that is larger than α by a factor of $O(\log N)$. In particular, if we "guess" a network congestion factor of at most $2 \cdot \alpha_k^*$, then the procedure can be used in order to establish a scheme for Problem I-OMR with a competitive ratio of $O(\log N)$. To that end, consider the following.

Definition 3.3 Given an instance of Problem I-OMR, define the value $\min_{i \in [1,k]} \left\{ \frac{\gamma_i}{\mathrm{K}_i} \right\}$ as $\frac{\gamma}{\mathrm{K}}$ and the set $\left\{ \frac{2^i}{c_e} \cdot \frac{\gamma}{\mathrm{K}} \middle| e \in E, i \in \left[0, \log\left(\frac{K}{\gamma} \cdot \sum_{i \in [1,k]} \gamma_i \right) \right], i \in \mathbb{Z} \right\} \cup \{0\}$ as $\overline{\overline{\alpha}}$ i.e.,

$$\frac{\gamma}{\mathrm{K}} \triangleq \min_{i \in [1,k]} \left\{ \frac{\gamma_i}{\mathrm{K}_i} \right\} \text{ and } \overline{\overline{\alpha}} \triangleq \left\{ \frac{2^i}{c_e} \cdot \frac{\gamma}{\mathrm{K}} \middle| e \in E, i \in \left[0, \log\left(\frac{K}{\gamma} \cdot \sum_{i \in [1,k]} \gamma_i \right) \right], i \in \mathbb{Z} \right\} \cup \{0\}.$$

Lemma 1 Given an instance of Problem I-OMRA, there exists at least one $\alpha \in \overline{\overline{\alpha}}$, such that $\alpha_k^* \le \alpha \le 2 \cdot \alpha_k^*$.

Proof Since for the case that $\alpha_k^* = 0$ the lemma is trivially satisfied ($0 \in \overline{\overline{\alpha}}$), we will consider only the case $\alpha_k^* > 0$. To that end, consider the optimal offline solution of Problem I-OMR. By the definition of α_k^*, there exists a link $e \in E$ with a flow (1) $s_e^*(k) = \alpha_k^* \cdot c_e$. Since $\alpha_k^* > 0$ and $c_e > 0$, it follows that (2) $s_e^*(k) > 0$. In addition, since for each $i \in [1,k]$, Problem I-OMR restricts the i-th request to have a $\frac{\gamma_i}{K_i}$-integral link flow, then, for each $e \in E$, it follows that (3) $s_e^*(k) = \sum_{i=1}^{k} n_i \frac{\gamma_i}{\mathrm{K}_i}$ where $n_i \in [0, \mathrm{K}_i]$.Thus, from (2) and (3), it follows that $s_e^*(k) \ge \min_{i \in [1,k]} \left\{ \frac{\gamma_i}{\mathrm{K}_i} \right\}$, therefore, by Definition 3.3, it follows that $s_e^*(k) \ge \frac{\gamma}{\mathrm{K}}$.

On the other hand, it is easy to see that $s_e^*(k) \le \sum_{i \in [1,k]} \gamma_i$. Therefore, $\frac{\gamma}{\mathrm{K}} \le s_e^*(k) \le \sum_{i \in [1,k]} \gamma_i$. Thus, there exists some i, $0 \le i \le \log\left(\frac{K}{\gamma} \cdot \sum_{i \in [1,k]} \gamma_i \right) - 1$, such that $2^i \cdot \frac{\gamma}{\mathrm{K}} \le s_e^*(k) \le 2^{i+1} \cdot \frac{\gamma}{\mathrm{K}}$. Hence, it follows from (1) that there exists some i,

$0 \le i \le \log\left(\frac{K}{\gamma} \cdot \sum_{i\in[1,k]} \gamma_i\right) - 1$, such that $2^i \cdot \frac{\gamma}{\mathrm{K}} \le \alpha_k^* \cdot c_e \le 2^{i+1} \cdot \frac{\gamma}{\mathrm{K}}$, hence (4) $\frac{2^i}{c_e} \cdot \frac{\gamma}{\mathrm{K}} \le \alpha_k^* \le \frac{2^{i+1}}{c_e} \cdot \frac{\gamma}{\mathrm{K}}$. Define $\alpha \triangleq \frac{2^{i+1}}{c_e} \cdot \frac{\gamma}{\mathrm{K}}$, and note that $\alpha \in \overline{\overline{\alpha}}$. We will prove that $\alpha_k^* \le \alpha \le 2 \cdot \alpha_k^*$. Since it immediately follows from (4) that $\alpha \ge \alpha_k^*$, we need only show that $\alpha \le 2 \cdot \alpha_k^*$. To that end, observe that, since $\alpha = \frac{2^{i+1}}{c_e} \cdot \frac{\gamma}{\mathrm{K}}$, it follows from (4) that $\frac{\alpha_k^*}{\alpha} = \frac{\alpha_k^*}{\frac{2^{i+1}}{c_e} \cdot \frac{\gamma}{\mathrm{K}}} \ge \frac{\frac{2^i}{c_e} \cdot \frac{\gamma}{\mathrm{K}}}{\frac{2^{i+1}}{c_e} \cdot \frac{\gamma}{\mathrm{K}}} = \frac{1}{2}$. Thus, $\alpha \le 2 \cdot \alpha_k^*$, and the lemma was established. ■

Note that the number of elements in the set $\overline{\overline{\alpha}}$ equals to $M \cdot \log\left(\frac{K}{\gamma} \cdot \sum_{i\in[1,k]} \gamma_i\right) = M \cdot \left(\log\frac{K}{\gamma} + \log \sum_{i\in[1,k]} \gamma_i\right) \le M \cdot \left(|\log K| + |\log \gamma| + \log\left[k \cdot \gamma_{\max}\right]\right) =$ $= O\left(M \cdot \left(\log K + \log k + \log \gamma_{\max}\right)\right)$. We use the fact that the set $\overline{\overline{\alpha}}$ contains a polynomial number of elements in order to establish a solution for Problem I-OMR, as follows.

Algorithm I-OMR Given are a network $G(V,E)$, capacities $\{c_e\}$ and a link flow $\{s_e(k-1)\}$ that consider the first *k*-1 requests. Upon the arrival of request $R(k)$ perform the following three steps:

1. perform a *binary search* over $\overline{\overline{\alpha}}$ in order to find the smallest $\alpha \in \overline{\overline{\alpha}}$ such that Procedure I-OMR succeeds for the input $\langle G(V,E), \{c_e\}, \{s_e(k-1)\}, R(k), \alpha \rangle$. Denote the resulting network congestion factor by $\alpha_{\min}$.
2. Perform $\{f_e(k)\} \leftarrow$ Procedure I-OMR $\left(G(V,E), \{c_e\}, \{s_e(k-1)\}, R(k), \alpha_{\min}\right)$. Apply the *flow decomposition algorithm* [9] on link flow $\{f_e(k)\}$ in order to obtain a path flow *f*.
3. Return *f*.

It follows from Theorem 1 and Lemma 1 that Algorithm I-OMR produces for the first *k* requests a network congestion factor that is larger by a factor of at most $2 \cdot \log_{\frac{3}{2}}(2 \cdot M) = O(\log N)$ than the optimal offline network congestion factor α_k^*. Thus, Algorithm I-OMR is an online scheme that solves Problem I-OMR with a competitive ratio of $O(\log N)$. In addition, as Algorithm I-OMR performs a binary search over $\overline{\overline{\alpha}}$, it follows that it executes Procedure I-OMR for at most

$$\log\left|\overline{\overline{\alpha}}\right| = O\left(\log\left[M\cdot\left(\log K + \log k + \log\gamma_{\max}\right)\right]\right) = O\left(\log N + \log\left[\log K + \log k + \log\gamma_{\max}\right]\right)$$

times. Hence, since the complexity of Procedure I-OMR is determined by the complexity of the capacity scaling algorithm, Algorithm I-OMR has a polynomial running time.

3.2 Solving Problem OMR

We now employ the solution to Problem I-OMR in order to derive an efficient online scheme with a competitive ratio of $O(\log N)$ for Problem OMR i.e., the general problem that has no restrictions on the integrality of the flow. The scheme is based on the following key observation.

Theorem 2 Given are an instance $\left\langle G(V,E),\{c_e\},\{R(k)\}\right\rangle$ of Problem I-OMR and an instance $\left\langle G(V,E),\{c_e\},\{R(k)\}\right\rangle$ of Problem OMR. Let α^*_{OMR} be the network congestion factor of the optimal (offline) solution to Problem OMR. Then, the network congestion factor of the optimal (offline) solution to Problem I-OMR is at most $2\cdot\alpha^*_{OMR}$.

Proof Consider the instance $\left\langle G(V,E),\{c_e\},\{\widetilde{R(k)}\}\right\rangle$ of Problem OMR where $\left\langle (s,t)_k, \mathrm{K}_k, 2\cdot\gamma_k\right\rangle \in \{\widetilde{R(k)}\}$ *iff* $\left\langle (s,t)_k, \mathrm{K}_k, \gamma_k\right\rangle \in \{R(k)\}$. Let path flow $f^*: P\to\mathbb{R}^+\cup\{0\}$ be the optimal offline solution to the instance $\left\langle G(V,E),\{c_e\},\{R(k)\}\right\rangle$ of Problem OMR. For each index *k* denote by f^*_k the path flow that considers solely request $R(k)$ in path flow f^* i.e., all the positive flows that together satisfy request $R(k)$ in f^*. Obviously, associating a path flow $g_k: P\to\mathbb{R}^+\cup\{0\}$ with each request $\widetilde{R(k)}$ such that $g_k(p) = 2\cdot f^*_k(p)$ for each $p\in P$ is a feasible solution to the instance $\left\langle G(V,E),\{c_e\},\{\widetilde{R(k)}\}\right\rangle$ of Problem OMR; moreover this solution produces a network congestion factor of $2\cdot\alpha^*_{OMR}$. For each index *k* and each path $p\in P$ round down the flow $g_k(p)$ to be a multiple of $\frac{\gamma_k}{K_k}$. In this process, the flow over each path in g_k is reduced by at most $\frac{\gamma_k}{K_k}$ flow units. Since for each *k*, request $\widetilde{R(k)}$ employs at most K_k paths, its total flow is reduced by at most γ_k flow units. Therefore, since for each *k* the original demand was $2\cdot\gamma_k$ flow units, it follows that, after the rounding, the total flow is of at least γ_k flow units.

Thus, for each k we have identified a $\frac{\gamma_k}{K_k}$ - integral path flow that transfers γ_k flow units from s_k to t_k. By definition, this is a feasible solution to the instance $\left\langle G(V,E),\{c_e\},\{R(k)\}\right\rangle$ of Problem I-OMR. Moreover, since in the rounding process we only reduce flow (and never increase it), the network congestion factor is at most $2\cdot\alpha^*_{OMR}$. ■

Given an instance $\left\langle G(V,E),\{c_e\},\{R(k)\}\right\rangle$ of Problem OMR and an instance $\left\langle G(V,E),\{c_e\},\{R(k)\}\right\rangle$ of Problem I-OMR, it follows from Theorem 2 that the optimal offline network congestion factors of both solutions differ by at most a factor of 2. Hence, since Algorithm I-OMR produces a solution with a competitive ratio of $O(\log N)$, it follows that the same algorithm can be employed in order to derive a solution for Problem OMR with a competitive ratio of $2\cdot O(\log N)=O(\log N)$. The latter is summaries as follows.

Algorithm OMR Given an instance $\left\langle G(V,E),\{c_e\},\{R(k)\}\right\rangle$ of Problem OMR, solve the instance $\left\langle G(V,E),\{c_e\},\{R(k)\}\right\rangle$ of Problem I-OMR using Algorithm I-OMR.

.

Corollary 1 Algorithm OMR is a polynomial online strategy for Problem OMR with a competitive ratio of $O(\log N)$.

4. A Lower Bound of Ω(logN) for Multipath Routing

In this section, we show that there is a lower bound of $\Omega(\log N)$ for the competitive ratio of any on-line multipath routing algorithm. In [1], it was established that there exists a lower bound of $\Omega(\log N)$ for the competitive ratio of online routing algorithms that limit the users to route their traffic along a *single* path. In this section, we show that the same lower bound exists if multipath routing is allowed i.e., each user can split its traffic along any number of paths. This implies that our $O(\log N)$-competitive algorithm presented in the previous section obtains a tight, i.e., best possible, competitive ratio.

The basic idea is to modify the lower bound of [1] and [11] to correspond to multipath routing. To that end, consider the network that appears in Fig. 2. Assume that *N* is a power of 2. Moreover, assume that the network has a single source node *S* and multiple targets $\{T_{i,j}\}$. Each target $T_{i,j}$ is located in the j-th location of layer i. Assume that: target $T_{1,1}$ is connected to all the nodes $v_1\cdots v_N$; targets $T_{2,1}$ and $T_{2,2}$ are connected to the nodes $v_1\cdots v_{\frac{N}{2}}$ and $v_{\frac{N}{2}+1}\cdots v_N$, respectively; and so forth. More formally, for each $1 \le i \le \log N$, the target $T_{i,1}$ is connected to the first set of $\frac{N}{2^{i-1}}$ nodes i.e., $v_1 \cdots v_{\frac{N}{2^{i-1}}}$, target $T_{i,2}$ is connected to second set of $\frac{N}{2^{i-1}}$ nodes i.e., $v_{\frac{N}{2^{i-1}}+1} \cdots v_{2\cdot\frac{N}{2^{i-1}}}$, and in general, the target $T_{i,j}$ is connected to the j-th set of $\frac{N}{2^{i-1}}$ nodes i.e., nodes $v_{\frac{(j-1)\cdot N}{2^{i-1}}+1} \cdots v_{\frac{j\cdot N}{2^i}}$. Finally, assume that all the links in the network have a single unit of capacity.

We construct a sequence of requests from the source S to the targets $\{T_{i,j}\}$. Each request can split its flow among any number of paths. The first aims at transferring a flow demand of $\frac{N}{2}$ units from the source S to some target in layer 1, the second request aims at transferring a flow demand of $\frac{N}{4}$ units from the source S to some target in layer 2, and so forth; in general, the *k*-th request aims at transferring a flow demand of $\frac{N}{2^k}$ units from the source S to the some target in layer *k*.

It remains to specify how to select the target within each layer. To that end, consider the first request. By construction, layer 1 has a single target node, namely, $T_{1,1}$. In order to determine the target within layer 2 (i.e., the target of the second request) we consider the congestion of the links $\{s \to v_i \mid 1 \le i \le N\}$ after applying the demand of the first request. More precisely, we consider the congestion of the two disjoint sets $\mathrm{A} \subseteq E$ and $\mathrm{B} \subseteq E$ where $\mathrm{A} \triangleq \left\{ s \to v_i \,\middle|\, 1 \le i \le \frac{N}{2} \right\}$ and $\mathrm{B} \triangleq \left\{ s \to v_i \,\middle|\, \frac{N}{2}+1 \le i \le N \right\}$; note that the links in A connect S to the target $T_{2,1}$, and the links in B connect S to the target $T_{2,2}$. The target of the second request is determined according to the set that has the larger *average* congestion factor. More specifically, the target of the next request is determined so that the set with the larger average link congestion factor is used again. Explicitly, if the set A has an average link congestion factor larger than that of B, the target of the next request is $T_{2,1}$; otherwise the selected target is $T_{2,2}$.

Consider now the network congestion factor after applying the first request. Since the first request transfers $\frac{N}{2}$ flow units from S to $T_{1,1}$ via the links of $\{s \to v_i \mid 1 \le i \le N\}$, the average link congestion factor of the links in $\{s \to v_i \mid 1 \le i \le N\}$ is $\frac{\text{Total flow}}{\text{Total capacity}} = \frac{1}{2}$. Hence, after applying the first request the network congestion factor is of at least $\frac{1}{2}$. Next, consider the network congestion factor when applying the second request. The second request transfers $\frac{N}{2^2} = \frac{N}{4}$ from S to either $T_{2,1}$ or $T_{2,2}$. Since the target is selected according to the set of links that has the larger average link congestion factor, it follows that, before the second request is satisfied, the links in $\{s \to v_i | 1 \le i \le N\}$ that connect S to the selected target have already an average link congestion factor of at least $\frac{1}{2}$. Moreover, note that, by construction, the traffic of the selected target is carried by at most $\frac{N}{2}$ links from $\{s \to v_i | 1 \le i \le N\}$; hence, the second request increases their average link congestion factor by $\frac{1}{2}$. Thus, after the second request is satisfied, the network congestion factor is at least 1.

In general, it is easy to see that this strategy increases the network congestion factor by $\frac{1}{2}$ for each new request. Thus, applying this strategy for a sequence of $\log N$ requests produces a network congestion factor of at least $\frac{\log N}{2}$. Thus, in order to show that the competitive ratio is $O(\log N)$, we have to show that the optimal offline algorithm can achieve a network congestion factor of $O(1)$. We shall show that it can achieve a network congestion factor of 1.

To that end, we show that the optimal offline algorithm can maintain for the *k*-th request a dedicated set of $\frac{N}{2^k}$ links from $\{s \to v_i | 1 \le i \le N\}$ that are not employed by other requests. Indeed, by construction, if $T_{k,j}$, $T_{k+1,h}$ are two target nodes selected according to the above strategy, then it holds that $T_{k,j}$ is reachable from S via two disjoint sets, each of $\frac{N}{2^k}$ links from $\{s \to v_i | 1 \le i \le N\}$, such that only one set can be used in order to connect S to $T_{k+1,h}$. Therefore, in the offline solution, $T_{k,j}$ has a set of $\frac{N}{2^k}$ links that are not employed by other requests. Denote the latter set of links as C. Since the demand of the $k-\text{th}$ request is of $\frac{N}{2^k}$ flow units, it holds that the set C can transfer the entire flow demand of the $k-\text{th}$ request without exceeding the congestion of its links beyond 1. Therefore, the optimal offline solution can achieve a network congestion factor of 1.

Therefore, we have shown that for any online algorithm of multipath routing, there is a scenario in which a network with at most $\tilde{N} \triangleq N \cdot \log N$ nodes achieves a network congestion factor larger by $\frac{\log N}{2}$ than the optimum. Since $\log \tilde{N} = \log(N \cdot \log N) = \log N + \log\log N = O(\log N)$, it follows that, for any online algorithm for multipath routing, there is a lower bound of $\Omega(\log N)$ for the best possible competitive ratio.

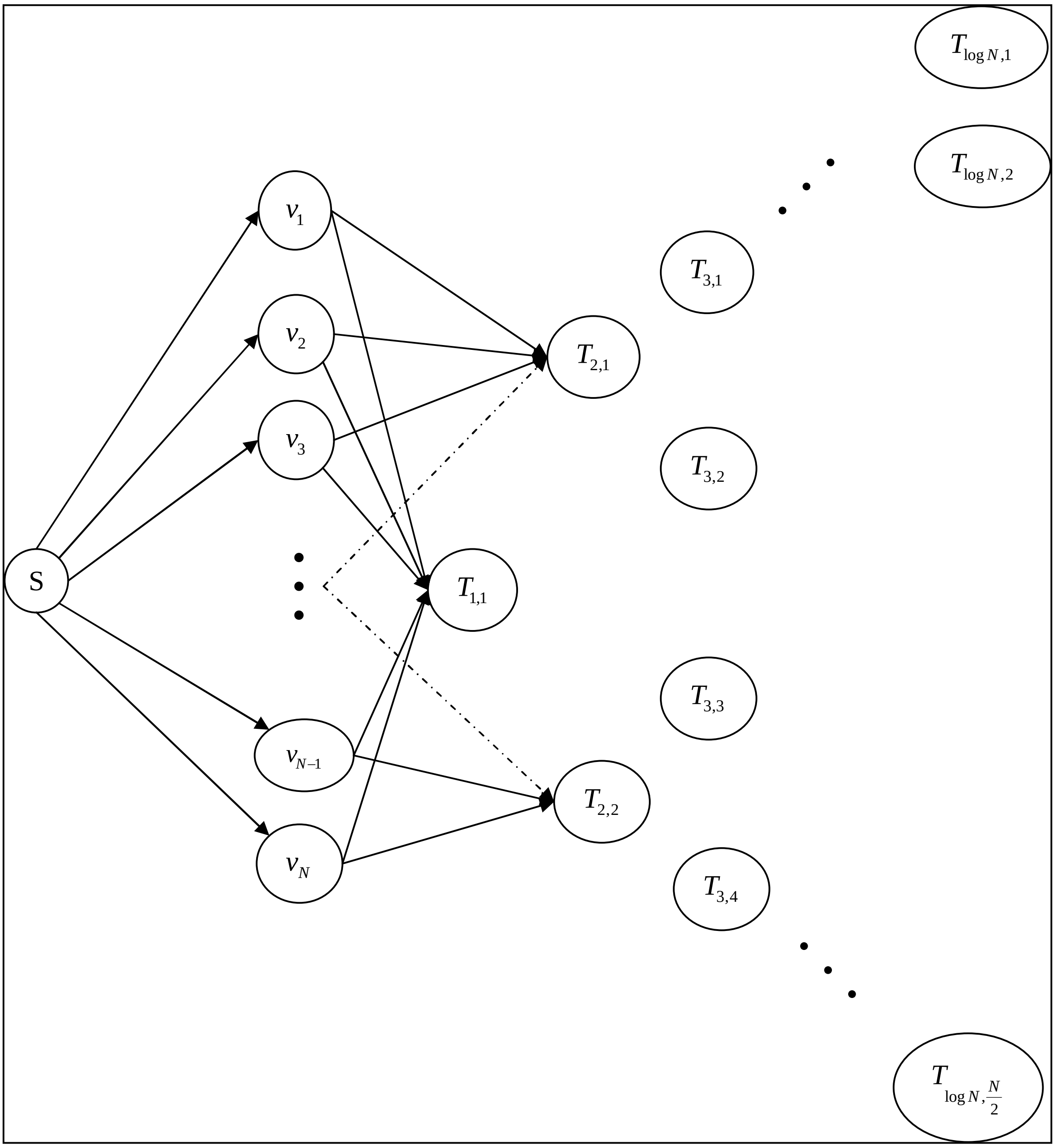

Fig. 2: A Lower Bound of Ω(logN) for Multipath Routing

References

[1] 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.

[2] Baruch Awerbuch, Yossi Azar, Serge Plotkin and Orli Waarts, "Competitive Routing of Virtual Circuits with Unknown Duration", In Proceedings of the 5th ACM-SIAM Symposium on Discrete Algorithms, pages 321-327, 1994.

[3] 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.

[4] A. Borodin, N. Linial and M. Saks, " An optimal online algorithm for metrical task system", Journal of ACM 39, Pages 745-763.

[5] 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.

[6] 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.

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

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

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

[10] 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.

[11] Y. Azar, J. Naor and R. Rom. "The Competitiveness of on-line Assignment", In proc. 3rd ACM-SIAM Symposium on Discrete Algorithms, pages 203-210, 1992.