Optimal OSPF Traffic Engineering using Legacy Equal Cost Multipath Load Balancing

Krisztián Németh*, Attila Kőrösi†, Gábor Rétvári*
*High Speed Networks Laboratory, Dept. of Telecommunications and Media Informatics
Budapest University of Technology and Economics (BME), Budapest, Hungary
†MTA (Hungarian Academy of Science) – BME Information Systems Research Group
Email: {krisztian.nemeth, korosi, retvari}@tmit.bme.hu

***Abstract*—In this paper, an optimal traffic engineering technique is proposed that uses only unmodified OSPF shortest path routing with stock ECMP (Equal-Cost MultiPath) load-balancing. The main problem with OSPF ECMP is that it can only divide the traffic in equal proportion among the least-cost paths. Our proposal works around this limitation by setting up virtual links alongside existing physical ones, this way adjusting the effective splitting ratio. In this paper, we study the special case of full-mesh MPLS overlays, which already promises with important practical applications yet turns out rather untrivial to solve. We formulate the problem of how to provision the virtual links to approximate the desired traffic splitting ratio, thereby minimizing congestion, we solve this problem under various restrictions arising from present OSPF practice, and finally we present extensive numerical evaluations suggesting that our technique effectively emulates optimal traffic engineering using only off-the-shelf IP routing technology.**

I. Introduction

The Traffic Engineering refers to the art and science of performance optimization in operational networks (TE, [1]). The most important objectives are to advance users' service experience and the profitability of the valuable infrastructure, by reducing the overall congestion and improving resource utilization across the network. TE is, accordingly, fundamentally concerned with carefully managing the traffic distribution on network links, avoiding traffic hotspots, and improving service stability and robustness.

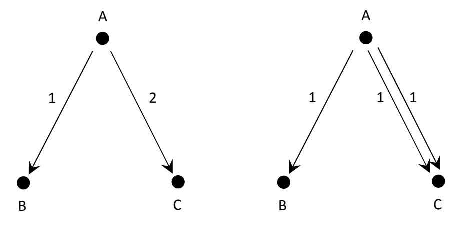
Realizing *optimal* traffic distribution, subject to the limited transit capacity of network links, is simple in theory: with a suitable traffic matrix at hand [2] just solve a simple multicommodity flow problem. This problem is polynomial time solvable if the flows are allowed to be fractional [3], while it is NP complete under integrality requirements [4]. Bringing the solution into effect, however, requires a separate connection oriented infrastructure, like MPLS RSVP-TE, dedicated solely for the purposes of TE, which often operators are reluctant to deploy [5].

On these grounds, many service providers choose "poor man's traffic engineering" and deploy traffic engineering right on top of the traditional IP routing protocols, like OSPF [6], that are already available in the network anyways. The basic idea is to adjust the administrative link costs in a way as to ensure that the shortest paths calculated by OSPF will map to exactly the ones chosen by the administrator [7]. OSPF TE can accommodate a surprisingly broad set of path selection strategies, the only requirement is that the selected paths be representable as shortest paths over properly chosen link costs [8], [9]. Unfortunately, the problem of calculating a path set that is shortest path representable *and* optimizes some traffic engineering goal at the same time is generally NP-hard [10]. Nevertheless, in this same paper the authors propose a heuristic that runs fast in the practical cases, and later several other authors followed suit [11].

The question still remains open how close OSPF TE can approximate optimal TE. Easily, there must be some performance loss, because in OSPF TE the paths selected for a user are intrinsically coupled with those of other users through the link costs. Hence, selecting an optimal path for some user may deteriorate the performance offered to another, or interfere with a third one, or might straight-out block a fourth. Another limitation stems from the fact that OSPF, at the moment, implements only a very restricted form of load balancing called Equal Cost Multipath (ECMP), where traffic between multiple equal cost shortest paths is split roughly evenly. As the optimal traffic allocation usually involves multiple paths with different flow shares, the solution obtainable by ECMP can deviate substantially from the ideal. This is so much so that, in certain networks, the quality of OSPF TE can become arbitrarily poor compared to optimal TE [10].

Interestingly, despite these limitations there is now strong *empirical* evidence that OSPF TE can perform pretty close to optimal TE [12]–[14]. It turns out that the fundamental limitation is not that the paths must be shortest path representable, but much rather that they must also be compatible with equal load splitting. In particular, it has been shown that optimal TE is *both theoretically and practically feasible over OSPF, provided that ECMP is substituted with a more sophisticated load balancing scheme* [15].

The most important contribution of our paper is to show that optimal TE is feasible *even without touching ECMP in any ways*. The idea is to *provision a carefully chosen virtual overlay in order to trick ECMP into realizing exactly the traffic splitting ratio chosen by us*. Perhaps it is easiest to understand this idea through an example (see Fig. 1(a)). Suppose the optimal traffic allocation (obtained, for instance, from solving a multicommodity flow linear program) would require that we split traffic along two equal cost shortest paths branching at router A in 1 to 2 proportion, that is, put $1/3$ part of the traffic



(a) original topology (b) topology augmented with a virtual link

Fig. 1. OSPF flow splitting with virtual links

to the link towards router B and $2/3$ part to the link to router C[1]. Easily, ECMP will not provide such a splitting ratio and rather distribute traffic in an $1/2$: $1/2$ proportion, which will most probably lead to congestion somewhere down along the paths. But if we provision a virtual link on the $A-C$ link, say, an Ethernet VLAN (see Fig. 1(b)), and expose this as an IP link to OSPF, then OSPF will happily use that link and distribute traffic in equal $1/3$: $1/3$: $1/3$ proportion amongst the three paths, realizing exactly the required traffic splitting ratio on the physical links. This is because the traffic brought to the two virtual links on the $A-C$ link add up, yielding that $2/3$ fraction of the traffic will flow through this link.

In this paper, we are not concerned with how the paths are computed or which L2 tunneling technique is used to provision the virtual links (e.g., Ethernet VLANs, IP-IP tunnels, GRE tunnels, stacked MPLS LSPs, etc.). We are only concerned with the so called *OSPF TE virtual link provisioning problem*, which asks how to allocate the right number of virtual links so that the traffic splitting ratios realized by ECMP match some required *ideal* splitting ratios the most closely.

As a first step in discovering this new intriguing problem area, in this paper we solve it for the special case of full-mesh MPLS overlays. This setting is still highly relevant in practice [13] yet challenging enough mathematically. Again, see a sample scenario in Fig. 2. In this simple transit network, there are three edge routers A, B, and C, and a full mesh MPLS overlay is realized between them containing two paths per router pair. This MPLS overlay, in turn, is seen as an IP topology deployed on top, which runs plain OSPF as a routing protocol. Easily, if the ideal traffic splitting ratios are like the ones given in the figure, then this traffic allocation is impossible to implement with ECMP. With the proposed technique, however, we can set up 4 virtual links (one between $A-B$ and three between $A-C$) to obtain exactly the required splitting (see later). Our solution is deployable right away with no HW/SW modification to the existing IP infrastructure, with a one time minor management intervention of setting up the virtual overlay. As such, ours is the first *theoretically and practically viable TE scheme using stock IP network gear*.

In this paper, we study the OSPF TE virtual link provisioning problem in the above setting that we call the *single commodity case*. We give a precise mathematical formulation

[1]This example is taken from the infamous "fish topology", a textbook example to demonstrate the limitations of OSPF TE [16]

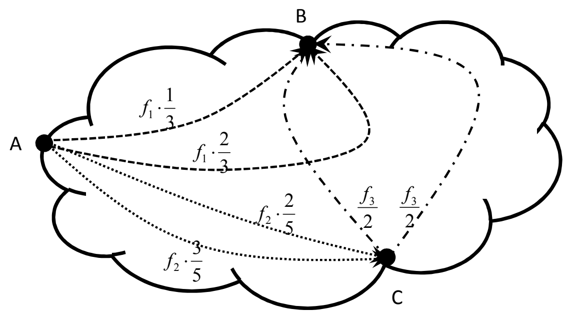


Fig. 2. MPLS TE full mesh overlay with given flow splitting ratios.

of the problem and we give various algorithms to solve it first in an ideal, non-restricted setting, and then under various practical limitations arising from present OSPF practice. We also give extensive numerical evaluation of the proposed algorithms.

The rest of this paper is organized as follows. In Section II we provide a thorough mathematical model and a problem statement. In Section III and Section IV we present optimal and approximate algorithms to solve the problem under various realistic assumptions, in Section V we present the results of our numerical studies, and finally Section VI concludes the paper and sketches future research directions.

II. Problem statement

This paper is concerned with the OSPF TE virtual link provisioning problem in the following single commodity case. Suppose we are given a pair of routers S and T and a set of $p_1, p_2, \ldots, p_k$ paths provisioned between them (as for example the $A-B$ pair in Fig. 2). Furthermore, assume that OSPF link costs have been set (e.g., using the techniques in [8]–[12]) so that p_1, p_2, …, p_k are exactly shortest paths between S and T. Alternatively, we may only be interested in the first-hop link along p_1, p_2, …, p_k (as is the case for Fig. 1); from the perspective of this paper these settings are equivalent. Suppose further that the traffic allocation that is to be realized, that is, the amount of subflows to be placed to each path is given as g_1, g_2, …, g_k, where $G = \sum_{i=1}^{k} g_i$ is the total volume of the traffic to be carried. (See Table I for a list of notations.)

Then, our objective is to route G amount of traffic over the p_1, p_2, …, p_k paths using OSPF ECMP, such that the actual $\hat{g}_1$, $\hat{g}_2$, …, $\hat{g}_k$ subflow values that emerge are as close as possible to the required g_1, g_2, …, g_k subflow volumes. Here, "closeness" between the i-the subflows is defined as

$$U_i = \frac{\hat{g}_i - g_i}{g_i} \ ,$$

and the global error metric is the maximum of the per-flow errors:

$$U = \max_{i=1\ldots k} \{U_i\} \ . \tag{1}$$

Within the context of this paper, we intend to reach the above objective by setting up virtual links or paths so that ECMP is tricked into realizing g_i by $\hat{g}_i$, $i = 1 \ldots k$. Thus, we

TABLE I
LIST OF NOTATIONS

number of next hops/paths used	k
desired traffic volume per next hop/path	$g_1, g_2, \ldots, g_k$
total traffic volume	$G = \sum_{i=1}^{k} g_i$
actual traffic volume per next hop/path	$\hat{g}_1, \hat{g}_2, \ldots, \hat{g}_k$
number of allocated links	$e_1, e_2, \ldots, e_k$
total number of allocated links	$E = \sum_{i=1}^{k} e_i$
upper bound on the total number of allocated links	Q

provision for every link/path i another $e_i - 1$ pieces of virtual links/paths, thus having altogether e_i parallel links/paths. For simplicity, we shall henceforth not differentiate between whether the problem is defined in terms of links or paths, neither we distinguish the physical substrate that carries the virtual links and the virtual links themselves, and we shall say that we have altogether e_i parallel links.

OSPF ECMP now sees e_i links and the volume of traffic sent to each of the physical links is given as

$$\hat{g}_i = G \cdot \frac{e_i}{\sum_{j=1}^{k} e_j} = \frac{Ge_i}{E}, \qquad i = 1 \ldots k \tag{2}$$

with the notation

$$E = \sum_{j=1}^{k} e_j \ .$$

The goal is to have $\hat{g}_i$ as close as possible to g_i for all $i = 1 \ldots k$ in terms of the error metric (1). This brings us to the following problem statement:

Definition 1. *OSPF TE virtual link provisioning problem, single commodity case (OSPF-TE-Virt):* given a set of target flows $g_1, g_2, \ldots, g_k$, find integers $e_1, e_2, \ldots, e_k$, so that the resultant ECMP flows $\hat{g}_i = \frac{Ge_i}{E}$, $i = 1 \ldots k$ minimize the flow error

$$U = \max_{i=1\ldots k} \left\{ \frac{\hat{g}_i - g_i}{g_i} \right\} = \max_{i=1\ldots k} \left\{ \frac{Ge_i}{Eg_i} - 1 \right\} \ .$$

In this formulation, *OSPF-TE-Virt* is a combinatorial optimization problem. It is sufficiently broad to capture all *single-commodity* formulations of flow problems, ranging from arc oriented formulations like single source–single sink (as of Fig. 1) and multiple source–single sink problems, to the single source-destination pair setting of MPLS full mesh overlays (Fig. 2). However, it is not suitable for describing inherently multi-terminal cases, where different commodities might pose different, and often conflicting, traffic splitting requirements that would be difficult to describe within the above framework, let alone solve. This multi-commodity case is, therefore, left herein for further study.

Some fundamental practical issues must still be clarified.

First, it is of question how many virtual links one can provision at an IP router, that is, how large E can become. We shall see that the larger the E the smaller the error, to the point that for the case of $E = G$ the problem becomes trivial. In practice, however, *the number of virtual links one can provision for a particular destination entry in the routing table is limited by the OSPF implementation*, in line with the OSPF RFC [6]. For example, in some Cisco IOS implementations this limit is adjustable but the maximum allowed setting is $E \leq 6$ [17], while this maximum is 16 in some Cisco NX-OS Softwares, Ericsson and Juniper routers [18]–[20]. On that ground, we shall also study the version of *OSPF-TE-Virt* when we are given an upper limit Q on E and are curious as to how to choose e_i with $\sum_i e_i = E \leq Q$ so as to minimize the error.

It is also a crucial question *whether e_i can become zero or not*. In many cases, having $e_i = 0$ for some links is beneficial as it reduces the overall error. We demonstrate this with a simple example: let $k = 3$, $g_1 = 100$, $g_2 = 100$, $g_3 = 1$, and let $Q = 4$. If we allow e_i to be 0, then the optimal solution of *OSPF-TE-Virt* is $e_1 = 1$, $e_2 = 1$, $e_3 = 0$, and the error is $U = U_1 = U_2 = 0.005$. If, however, $e_i = 0$ is not allowed, then the best we can achieve is $e_1 = 2$, $e_2 = 1$, $e_3 = 1$, with error $U = U_3 = 49.25$, which is clearly worse than in the previous case. In practical terms, allowing $e_i = 0$ means that a link is administratively disabled in the router. Whether this can be done or not is a network administration decision, which is well beyond the scope of this paper, so in the sequel we shall treat these cases separately and give respective solutions.

III. Optimal Link Allocation: Error Bounds

In this and the next section, we seek solutions to the *OSPF-TE-Virt* problem. First, we search for bounds on the error under different circumstances.

A. Unlimited Number of Links

We start with the case where the number of virtual links that can be allocated on the router is unlimited. First, we show that in this case if the desired traffic volume is given with rational numbers, then an optimal solution can always be reached. Formally:

Theorem 1. *Let $Q = \infty$ and let $g_i \in \mathbb{Q}^+$. Then, $\exists e_i : (i = 1 \ldots k)$, so that $U = 0$.*

Proof. The proof is constructive: we show an actual setting for e_i. As $g_i \in \mathbb{Q}^+$ we can write:

$$g_i = \frac{a_i}{b_i} \qquad (i = 1 \ldots k) \ ,$$

such that for all i the greatest common divisor GCD$(a_i, b_i) = 1$. Furthermore let $N =$ LCM$(\{b_i\})$ (LCM: least common multiple). Let

$$g'_i = \frac{a_i N}{b_i} \qquad (i = 1 \ldots k, \ g'_i \in \mathbb{Z}^+) \ ,$$

and let $M =$ GCD$(\{g'_i\})$, and so

$$g''_i = \frac{g'_i}{M} = \frac{g_i N}{M} \qquad (i = 1 \ldots k) \ .$$

It is easy to see that $g''_i \in \mathbb{Z}^+$ and that

$$\text{GCD}(\{g''_i\}) = 1 \ . \tag{3}$$

Now let $G'' = \sum_{j=1}^{k} g''_j$, and simply let

$$e_i = g''_i = \frac{g_i N}{M} \qquad (i = 1 \ldots k) \ . \tag{4}$$

Then for each $i = 1 \dots k$ the error is:

$$U_i = \frac{Ge_i}{g_i E} - 1 = \frac{Ge_i}{g_i \sum_{j=1}^k e_j} - 1 = \frac{G g_i \frac{N}{M}}{g_i \sum_{j=1}^k g_j \frac{N}{M}} - 1 = 0 \ ,$$

so the problem can be solved optimally. □

Next, we also show that the solution obtained above is in fact minimal, in the sense that no different virtual link allocation exists that attains zero error with fewer virtual links.

Theorem 2. *Select $e_i : (i = 1 \dots k)$ as of (4). Then, $\nexists \{e_i'\}$ for which $U = 0$ such that $E' = \sum_{j=1}^k e_j' < E$.*

Proof. Suppose the opposite, i.e., there are such $\{e_i'\}$. We can also suppose that GCD($\{e_i'\}$) = 1 (otherwise one can divide the e_i's with the GCD). We also know from (3) and (4) that

$$\text{GCD}(\{e_i\}) = 1 \ . \quad (5)$$

With e_i''s the actual traffic volumes are for each $i = 1 \dots k$

$$\hat{g}' = \frac{G'' e_i'}{E'} = \frac{E e_i'}{E'} \ .$$

Knowing that $U = 0$ and using (4):

$$\frac{E e_i'}{E'} = g_i'' = e_i \qquad (i = 1 \dots k) \ ,$$

so

$$\frac{e_i'}{e_i} = \frac{E'}{E} \qquad (i = 1 \dots k) \ .$$

Let

$$\frac{s}{t} = \frac{E'}{E}, \ s < t; \ s, t \in \mathbb{Z}^+; \ \text{GCD}(s,t) = 1 \ .$$

As a consequence, $t > 1$. From the latter two equations:

$$e_i' = \frac{e_i s}{t} \qquad (i = 1 \dots k) \ .$$

From $e_i', e_i, s, t \in \mathbb{Z}^+$ and GCD$(s,t) = 1$ follows that $t | e_i$ $(i = 1 \dots k)$, which contradicts (5). □

We note that if the desired traffic volumes are given by reals ($g_i \in \mathbb{R}^+$), then the optimal solution can be approximated with arbitrary precision using the above technique, meaning that the error U can be smaller than any $\epsilon > 0$. The proof is based on the fact that there is a rational number arbitrary close to any given irrational number, but it is omitted due to lack of space.

B. Limited Number of Links

If the total number of links used by the OSPF ECMP is limited ($E \leq Q < \infty$), the problem becomes more interesting. For the sake of simplicity, let us suppose that $g_i \in \mathbb{Z}^+$, and GCD($\{g_i\}$) = 1. Using Theorem 1 and Theorem 2, we conclude that optimal solution ($U = 0$) can be reached in this case if and only if $Q \geq G$. So let us now focus on the case when $Q < G$ and search for e_i that minimizes U.

First, we give a simple universal upper bound on the error U for this setting.

Lemma 3. $U \leq G - 1$.

Proof. Using (2) we get

$$U_i = \frac{Ge_i}{Eg_i} - 1 = \frac{G}{g_i} \cdot \frac{e_i}{E} - 1$$

and since $g_i \geq 1$ (as $g_i \in \mathbb{Z}^+$) and $e_i \leq E$

$$U_i = \frac{G}{g_i} \cdot \frac{e_i}{E} - 1 \leq G - 1 \ . \qquad □$$

Let us now study the case when links are not allowed to be disabled, that is, $e_i > 0$ $(i = 1 \dots k)$. We found that for large G's, the error can become arbitrarily large in this case.

Lemma 4. *If $e_i > 0$ $(i = 1 \dots k)$ is required and G is unbounded, then U can become arbitrarily high for any $Q > 1$.*

Proof. Consider the below example. Let

$$k = 2, \ g_1 = 1, \ g_2 = xQ \ \ (x > 1, x \in \mathbb{Z}) \ , \quad (6)$$

($k = 2$ is allowed as $Q > 1$).

Then $G = xQ + 1$ and the optimal allocation of links (which minimizes the error) along with the error is:

$$e_1 = 1, \ e_2 = Q - 1,$$
$$\hat{g}_1 = 1 \cdot \frac{xQ+1}{Q}, \ \hat{g}_2 = (Q-1) \cdot \frac{xQ+1}{Q},$$
$$U = U_1 = \frac{xQ+1}{Q} - 1 = x - 1 + \frac{1}{Q} \ ,$$

which can be arbitrary high, as x is unbounded. □

Interestingly, if $e_i = 0$ is allowed then the problem described in (6) can be solved with bounded error as follows. Let $e_1 = 0$ and $e_2 = 1$. Then

$$\hat{g}_1 = 0, \ \hat{g}_2 = xQ + 1,$$
$$U = U_2 = \frac{\hat{g}_2}{g_2} - 1 = \frac{xQ+1}{xQ} - 1 = \frac{1}{xQ} \ ,$$

which goes to 0 as $x \to \infty$. In fact allowing $e_i = 0$, for any g_1 and g_2 the error can never exceed 1, which is a consequence of the following lemma:

Lemma 5. *If $e_i \geq 0$ (i=1...k) is required, then for any $Q, k, \{g_i\}$: $U \leq k - 1$.*

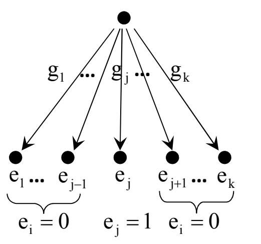
Proof. Let g_j be te maximal element of the $\{g_i\}$ set (or one of the maximal elements, if there are more than one), so $g_j \geq g_i$, for all $i = 1 \dots k$. Then let

$$e_i = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \ , \end{cases}$$

which is shown in Fig. 3. Then

$$U = U_j = \frac{\hat{g}_j}{g_j} - 1 = \frac{\sum_{i=1}^k g_i}{g_j} - 1 \leq$$
$$\leq \frac{\sum_{i=1}^k g_j}{g_j} - 1 = k - 1 \ . \qquad □$$

Observe that Lemma 5 characterizes the error in terms of k. The question remains to be decided whether or not the error



g_1 ... g_j ... g_k
e_1 ... e_{j-1} e_j e_{j+1} ... e_k
$e_i = 0$ $e_j = 1$ $e_i = 0$

Fig. 3. Link allocation for Lemma 5

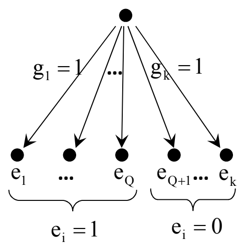


Fig. 4. Unbounded k

can indeed become unbounded with k going to infinity. The below result answers this question in the affirmative:

Lemma 6. *If $e_i \geq 0$ (i=1...k) is required and k and G are unlimited, then for any fixed Q the error U can become arbitrary high.*

Proof. Again, consider an example. Let $k > Q$ and let $g_i = 1$ for all $i = 1 \dots k$. Then the error is minimal if for Q links we set $e_i = 1$, e.g., for $i = 1 \dots Q$ we have $e_i = 1$, and we set $e_i = 0$ for the rest (see Fig. 4). Then the error is:

$$U = U_1 = \frac{\hat{g}_1}{g_1} - 1 = \frac{G \cdot \frac{e_1}{\sum_{i=1}^{k} e_i}}{1} - 1 = \frac{G}{Q} - 1 = \frac{k}{Q} - 1 \ .$$

As Q is fixed and k is unbounded, U is unbounded as well. □

IV. Optimal Solutions

We turn now to discuss how to actually solve the OSPF TE virtual link provisioning problem. Again, we suppose that $g_i \in \mathbb{Z}^+$, and $\mathrm{GCD}(\{g_i\}) = 1$.

Easily, if $Q \geq G$ then the problem is trivial: just allocate G virtual links so that there are exactly g_i links for each i (i.e., $e_i = g_i$), and the error becomes zero. However, this is not particularly practical in many cases. Thus, in this section we shall assume that $Q < G$.

First let us count the possible link allocations, denoted henceforth by N.

Lemma 7. *The number of possible virtual link allocations if $e_i \geq 0$ is $N = \binom{Q+k}{k} - 1$.*

Proof. We have to distribute the Q links among $k+1$ places: there are k places for the k links, plus one place for the unused ones. Note that we have to subtract one, as having 0 on all of the links is not allowed. Using the formula of combinations with repetitions, we got

$$N = \binom{Q + (k+1) - 1}{(k+1) - 1} - 1 = \binom{Q+k}{k} - 1 \ . \quad \square$$

Lemma 8. *The number of possible allocation if $e_i > 0$ is $N = \binom{Q}{k}$.*

Proof. This proof is similar to the previous one, but in this case we have to put at least one link for each of the k places, and the remaining $Q-k$ links have to be distributed among the $k+1$ places: k places for the k links, plus for the unused ones. In this case, however, it is allowed to put all the $Q-k$ links to the $k+1$th place (meaning that none of them is used), as still there is a link one each place. This is again combinations with repetitions with the number of possibilities being:

$$N = \binom{(Q-k) + (k+1) - 1}{(k+1) - 1} = \binom{Q}{k} \ . \quad \square$$

A. Integer Linear Program for OSPF-TE-Virt

Next, we turn to the actual algorithms to solve OSPF-TE-Virt. One way to obtain a solution would be a brute force method, but the above results about N practically eliminate the chance that such an approach could be successful. Therefore, first we provide an Integer Linear Program (ILP) to obtain a solution, and then we shall also give alternatives.

The ILP below is formulated for the case when $e_i \geq 0$.

variables: $e_i \quad i = 1 \dots k$
$y_i \quad i = 1 \dots Q$
α

parameters: $g_i \quad i = 1 \dots k$
$G = \sum_{i=1}^{k} g_i$
r (a small number, e.g., 10^{-5})
M (a large number, e.g., 10^{5})

objective function:
$$\min \alpha + r \sum_{i=1}^{k} e_i \tag{7}$$

constraints:
$$\sum_{j=1}^{Q} y_j = 1 \tag{8}$$
$$\sum_{i=1}^{k} e_i = \sum_{j=1}^{Q} j y_j \tag{9}$$
$$\frac{e_i G}{g_i} \leq (\alpha + 1) j + M(1 - y_j), \quad i = 1 \dots k, j = 1 \dots Q \tag{10}$$
$$e_i \geq 0, e_i \in \mathbb{Z} \quad i = 1 \dots k \tag{11}$$
$$y_j \in \{0, 1\} \quad j = 1 \dots Q$$

The idea is to minimize the error U, by requiring for each link error $U_i \leq \alpha$:

$$U_i = \frac{\hat{g}_i}{g_i} - 1 = \frac{G e_i}{E g_i} - 1 \leq \alpha \qquad i = 1 \dots k \tag{12}$$

and minimizing α. The y_i variables help to find the optimal E: $y_i = 1$ if $E = i$ and $y_i = 0$ otherwise, which is enforced by constraints (8) and (9). The (10) system of constraints is only effective if $E = j$, and then results in inequality (12). The second term of the objective function (7) ensures that if there are several optimal solutions, then the solver would choose the one with the smallest number of links altogether.

When $e_i = 0$ is not allowed, a slight modification of the above ILP is enough: constraint (11) should be substituted with $e_i \geq 1$, $e_i \in \mathbb{Z}$ $i = 1 \dots k$.

B. Iterative Algorithm for OSPF-TE-Virt

While the ILP is a practical solution method in many cases, theoretically it is not guaranteed to run in polynomial time. Therefore we present an iterative algorithm that can solve the link allocation problem in pseudo-polynomial time. In the following, we suppose that $e_i = 0$ is allowed.

First we describe an algorithm that, for a given α, $\{g_i\}$ and E, checks whether or not it is possible to assign the links with $U \le \alpha$. If the assignment is feasible, then this algorithm also provides the solution:

Algorithm 1. *Error reachability:* For each $i = 1, \dots, k$, let

$$x_i = \left\lfloor \frac{(\alpha+1)g_i E}{G} \right\rfloor \ . \tag{13}$$

Solve the following set of equations to find $\{e_i\}$:

$$0 \le e_i \le x_i, \qquad e_i \in \mathbb{Z}, \quad i = 1 \dots k \tag{14}$$

$$\sum_{i=1}^{k} e_i = E \ . \tag{15}$$

It is easy to see that there is exactly one solution if $\sum_{i=1}^k x_i = E$, no solution if $\sum_{i=1}^k x_i < E$, and there are multiple solutions if $\sum_{i=1}^k x_i > E$.

Theorem 9. *The link allocation problem can be solved with $U \le \alpha$ if and only if*

$$\sum_{i=1}^{k} x_i \ge E \ . \tag{16}$$

Proof. For any solution $\{e_i\}$,

$$\alpha \ge U = \max_{j=1\dots k} \left\{ \frac{\hat{g}_j}{g_j} - 1 \right\} \ge \frac{\hat{g}_i}{g_i} - 1 = \frac{G e_i}{E g_i} - 1, \; i = 1 \dots k$$

thus

$$\frac{(\alpha+1)g_i E}{G} \ge e_i, \qquad i = 1 \dots k$$

and since $e_i \in \mathbb{Z}$,

$$\frac{(\alpha+1)g_i E}{G} \ge \left\lfloor \frac{(\alpha+1)g_i E}{G} \right\rfloor = x_i \ge e_i, \; i = 1 \dots k \ . \tag{17}$$

So if (16) holds, then we can find e_i values such that (14) and (15) are satisfied, and then due to (17) we will have a valid assignment, where $U \le \alpha$.

On the other hand, if (16) does not hold, then we cannot find e_i's such that (15) is satisfied, and $U \le \alpha$. To see this, suppose the opposite. Then (17) still must be true, and then the supposed $\sum_{i=1}^k x_i < E$ contradicts (15). □

As calculating x_i's according to (13) is simple (i.e., $O(1)$), this algorithm has a complexity of $O(k)$.

With the *Error reachability* algorithm in place, we would like to use it in a binary search framework for finding a minimal α that is satisfiable, given g_i's and E. To start the binary search, first we need an upper bound on U. For this, we shall use the upper bound $k-1$ as of Lemma 5. To stop the iteration, we also need a lower bound on $|U_i - U_j|$, $i, j = 1 \dots k$. This lower bound should consider all possible allocations of the E links, so U_i and U_j can be part of different allocations. This means that $\sum_{n=1}^k e_n = E$ does not necessarily hold, it is even possible that $e_i + e_j > E$.

Now, we show that $\frac{1}{GE}$ is such a lower bound. Let's suppose $U_i > U_j$. Then

$$\Delta U = \left(\frac{G}{E}\frac{e_i}{g_i} - 1 \right) - \left(\frac{G}{E}\frac{e_j}{g_j} - 1 \right) = \\ = \frac{G}{E}\frac{e_i g_j - e_j g_i}{g_i g_j} \ge \frac{G}{E}\frac{1}{G^2} = \frac{1}{GE} \ , \tag{18}$$

since $e_i g_j - e_j g_i$ is a positive integer.

Note for (18) that it is possible, that $U_i = U_j$. An example for this is the following: $k = 2, g_1 = 1, g_2 = 1, E = 3$. One possible allocation is $e_1 = 2, e_2 = 1$, where $U_1 = 1/3$. Another possible allocation is $e_1 = 1, e_2 = 2$, where $U_2 = 1/3$. This has no effect on the lower limit given in (18): if these U's happen to be the optimal errors for two different allocations, just like in our example, then finding the optimal α results in $\sum_{i=1}^k x_i > E$, which means that there are more than one optimal solutions.

The binary search method is summarized as follows.

Algorithm 2. *Binary search for minimal error:* Initialize $\alpha = k - 1$. Do a binary search for the minimal α for which Algorithm 1 answers affirmative, that is, finds an allocation $e_i : i \in 1 \dots k$ satisfying $U \le \alpha$. Stop if the difference for α in the last two iterations falls below $\frac{1}{GE}$.

After the algorithm terminates, it delivers the optimal $\{e_i\}$ setting, along with the corresponding error U. This can be done in $\log(kGE)$ steps, yielding an overall in $O(k \log(kGE))$ complexity.

What remained to be done is to actually find the value of E subject to the given Q that yields the smallest error. This is done by the below simple algorithm:

Algorithm 3. *Iterative* OSPF-TE-Virt *algorithm:* Run *binary search for minimal error* algorithm for each $E = 1 \dots Q$ and choose the solution with the smallest error.

Note that this is theoretically not a polynomial time algorithm as the complexity is $O(Qk \log(kGE))$, which is not polynomial in Q (as the size of the input parameter Q is $\log(Q)$). For a fixed Q, however, as it is the case in practice, the algorithm is indeed polynomial. Moreover, as shall be described in the next section, we found this algorithm easily tractable for all practical use cases.

This algorithm is easy to tweak for the case when $e_i \ne 0$ must hold. In Algorithm 1, (14) has to be replaced by $1 \le e_i \le x_i$, $e_i \in \mathbb{Z}$, $i = 1 \dots k$. Additionally, in the iterative search we cannot use $k - 1$ as an upper bound for the error but, by Lemma 3, $G - 1$ can be used. Finally, we have to search for the best allocation in the $E = k \dots Q$ space instead of $E = 1 \dots Q$. The complexity of this version is then $O(Qk \log(G^2 E))$.

V. Simulation Experiments

We have validated our idea using simulation experiments. Our main questions were how the error U depends on the maximum number of links in practical scenarios and what the implications on the network congestion are.

A. Simulation Scenario

We have chosen to implement *Iterative* OSPF-TE-Virt *algorithm* instead of the ILP as it offers fast running time. We have implemented the algorithm in C++ using the Lemon Graph Library [21]. In order to get realistic input data relevant to the case of OSPF Traffic Engineering, that is, to obtain k and a set of g_i flow requirements to be fed to the algorithm, we have used two different traffic engineering schemes.

Maximum Multicommodity Flows (MCMF): In this approach we have selected a set of source-destination pairs and we have set up flows between them, such that the total sum of the flows is maximal. Flows between the source-destination pairs were allowed to split at the intermediate nodes. We have formulated the problem as an ILP, and solved it using the free GLPK solver [22]. This resulted in flow values for each link and for each source-destination pair, which we decomposed into subflows. The number of the subflows gave us the k parameter, while the value of the subflows was used as g_i (different for each source-destination pair).

Maximum Arc-disjoint Paths (MADP): In this approach we were looking for the maximum number of arc-disjoint paths between two random nodes of a capacitated network. For this we have used the Lemon implementation of Suurballe's algorithm [23]. k has been set to the number of the returned paths. On each of the paths we have searched for the link with the minimum capacity, corresponding to the capacity of that path, which has then been assigned as g_i.

We have used several input topologies for the evaluations. For the sake of brevity, in this paper we provide the results for two selected topologies (the rest produced quite similar results). The first one was taken from the inferred ISP data maps of the Rocketfuel dataset [24] (AS3967). We obtained approximate POP-level maps by collapsing the topologies so that nodes correspond to cities and we eliminated leaf-nodes. This network comes with inferred link costs. The resulting graph contained 21 nodes and 72 capacitated edges, with an average node degree of 3.43. The second topology was a larger one, modeling the ITC Deltacom fiber backbone network (as of 2010) with 113 nodes and 322 capacitated edges (average node degree of 2.85), taken from the Topology-Zoo project's dataset [25].

For the maximum multicommodity flow scenario we selected 8 source-destination pairs randomly according to a uniform distribution. For the maximum arc-disjoint paths case the source-destination pairs were again randomly selected according to a uniform distribution. The experiments were repeated 5000 times. This meant 5000 input sets $(k, \{g_i\})$ for the *Iterative* OSPF-TE-Virt *algorithm* for the maximum arc-disjoint paths selection software, but more for the maximum multicommodity flow, as it resulted in up to 8 input sets in one run. We have examined the $Q = 6 \dots 16$ range, which is the most relevant in current practical scenarios.

The parameters we measured were as follows. First, we observed the error U, as produced by the *Iterative* OSPF-TE-Virt *algorithm* (alg. err.). Second, we were interested in the extent of congestion that would arise should we deploy the virtual overlay as provisioned by the algorithm. For this, we fed the calculated $\hat{g}_i$ subflow volumes to the network instead of the ideal g_i's, and we measured the link overloads ($\frac{load}{capacity} - 1$) inside the network. The parameters we calculated were the average congestion (link avg. err.) and the maximum of the congestion (link max. err.), taken over all links.

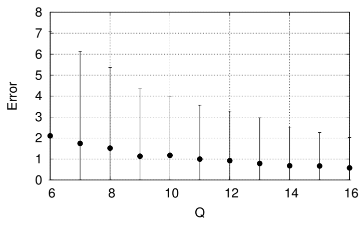
B. Results

The results are given in Fig. 5 and Fig. 6. The dots are the averages of the results, while the error bars show the 5th and 95th percentile of the data. For each plot, the first term in the caption shows the algorithm used to determine the algorithm's input, i.e., maximum multicommodity flows (MCMF), or maximum arc-disjoint paths (MADP); the second term shows if $e_i = 0$ was permitted; and finally third stands for the measured parameter, i.e., either the error given by the algorithm (alg. err.), the average congestion (link avg. err.), or the maximum congestion (link max. err.), each one plotted against the upper bound on the number of links Q. Thus, the caption "MCMF, $e_i > 0$, alg. err." of Fig.5(a), for instance, indicates that this diagram shows the average of the error U produced by the algorithm, taken over all inputs produced by MCMF, in the case when physical links/paths cannot be disabled (i.e., $e_i > 0$). The MADP results for the Deltacom topology have been excluded for brevity.

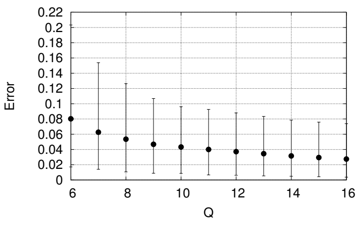
Our main observations are as follows. The results indicate that the larger the number of permitted virtual links Q the smaller the error. This is easy to understand intuitively, as the algorithm has immensely more allocations to choose from for larger Q (recall Lemma 7 and Lemma 8). It can also be seen that the error drops dramatically if we allow to disable physical links/paths (the $e_i \geq 0$ case). We also found that the results are somewhat better for the maximum arc-disjoint paths case. The reason behind this is probably that the split ratios happened to be more even in this case.

Regarding the congestion, the worst cases of link overload belong to the maximum multicommodity flow generation with the $e_i > 0$ requirement. Even in this case the overload is only $50 - 100\%$ on the most over-utilized link for all examined Qs. This may seem prohibitively high at the first sight, but in practice links are not planned to be utilized at 100%, and for a planned 50% utilization an extra $50 - 100\%$ means a total utilization of $75 - 100\%$. Again, this is the worst case in our results, and the average link overloads are orders of magnitudes lower than that. For the rest of the cases the congestion was in the range 5–15% even for very low Q.

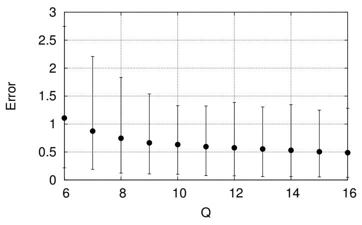
To sum up the results, an overall acceptable error level is realizable by our algorithms with just $6 - 16$ virtual links and, to the extent we are aware of, that many virtual links is allowed on most commercial IP router offerings of our days.



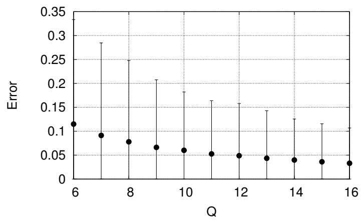
(a) MCMF, $e_i > 0$, alg. err.



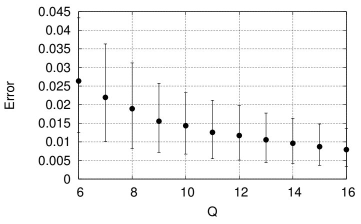
(b) MCMF, $e_i > 0$, link avg .err.



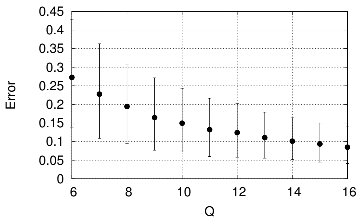
(c) MCMF, $e_i > 0$, link max. err.



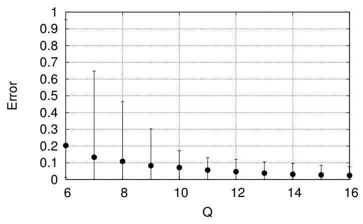
(d) MCMF, $e_i \geq 0$, alg. err.



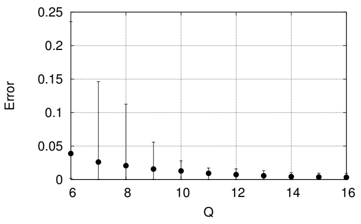
(e) MCMF, $e_i \geq 0$, link avg. err.



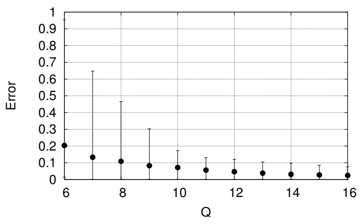
(f) MCMF, $e_i \geq 0$, link max. err.



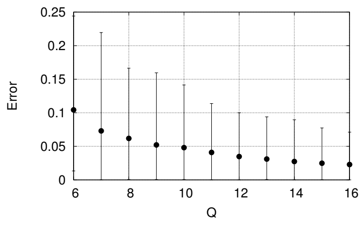
(g) MADP, $e_i > 0$, alg. err.



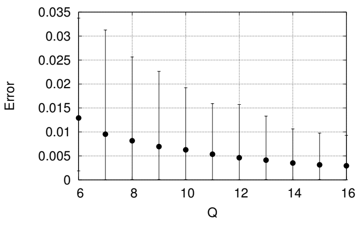
(h) MADP, $e_i > 0$, link avg. err.



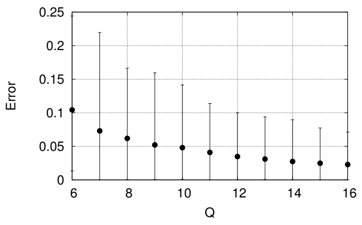
(i) MADP, $e_i > 0$, link max. err.



(j) MADP, $e_i \geq 0$, alg. err.

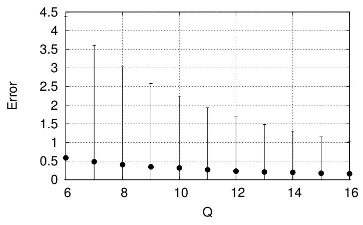


(k) MADP, $e_i \geq 0$, link avg. err.

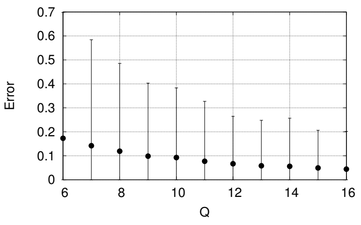


(l) MADP, $e_i \geq 0$, link max. err.

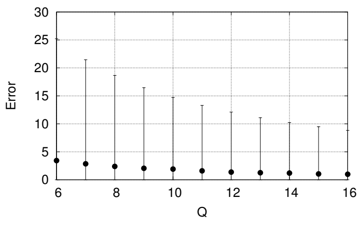
Fig. 5. Results for AS3967



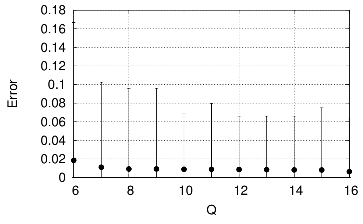
(a) MCMF, $e_i > 0$, alg. err.



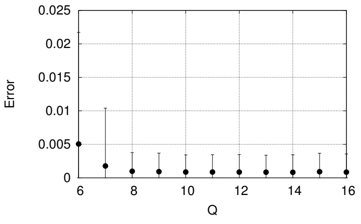
(b) [MCMF, $e_i > 0$, link avg .err.



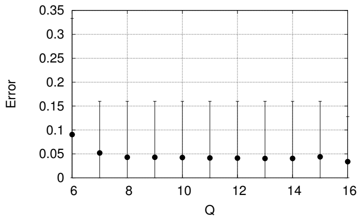
(c) MCMF, $e_i > 0$, link max. err.



(d) MCMF, $e_i \geq 0$, alg. err.



(e) MCMF, $e_i \geq 0$, link avg. err.



(f) MCMF, $e_i \geq 0$, link max. err.

Fig. 6. Results for Deltacom Network

Regarding the running time, our algorithm has proven very effective. The running time for the 5000 repetitions on an average PC was 3 sec for the maximum arc-disjoint path case and 10 sec for the maximum multicommodity flow scenario, which includes both running times of the path calculation and the *Iterative* OSPF-TE-Virt *algorithm* itself. The memory consumption was also moderate: less than 80 MB in both cases.

VI. Summary and Future Works

In this paper an potentially optimal traffic engineering solution is proposed that is built solely on legacy OSPF ECMP technology. Using our proposal high quality IP TE can be realized without any hardware or software modification, engaging in only minor administrative settings.

Our premise was the observation that the main reason for the inefficiency of OSPF for TE lies in the way it realizes load balancing: ECMP can only divide the incoming traffic into equal portions at the splitting nodes, which might cause highly suboptimal traffic distribution in certain cases [12]. Our proposed idea is to provision a virtual overlay, by setting up virtual links properly, in order to trick ECMP into achieving the desired uneven traffic split ratio. We analyzed the lower and upper bounds on the attainable error in the resultant traffic splitting ratios and we gave an optimal Integer Linear Program and a pseudo-polynomial heuristics to realize desired splitting. In practice, router implementations pose strict limitations on the number of equal cost paths, which could jeopardize the optimal solution. Therefore, we have proposed extensions to our algorithms to find the best virtual link allocation in such a constrained environment. Extensive numerical studies has been carried out to evaluate the usefulness of our proposal. The results are reassuring: in practical situations near-optimal solutions can be reached with as few as half a dozen links, which is easily realizable in today's routers.

The solution of the OSPF TE virtual link provisioning problem for the single commodity case, presented in this paper, can be applied in several real-life scenarios, including the MPLS multipath full-mesh overlay networks. While these applications are very important by themselves, we plan to continue exploring this interesting topic and solve the problem for the general multicommodity case, when there are several source-destination node pairs exchanging data concurrently, sharing the resources of a single legacy OSPF ECMP domain.

Acknowledgements

The authors would like to thank the support of the Dept. of Telecommunications and Media Informatics and the High Speed Networks Laboratory, and especially their heads, Tamás Henk and Attila Vidács.

Gábor Rétvari was supported by the MTA-BME Future Internet Research Group and the OTKA/PD-104939 grant. This work was partially supported by the European Union and the European Social Fund through project FuturICT.hu (grant no.: TAMOP-4.2.2.C-11/1/KONV-2012-0013).

References

[1] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and principles of Internet traffic engineering," RFC 3272, May 2002.
[2] S. Gunnar, M. Johansson, and T. Telkamp, "Traffic matrix estimation on a large IP backbone: a comparison on real data," in *ACM SIGCOMM conference on Internet measurement*, ser. IMC'04, 2004, pp. 149–160.
[3] M. S. Bazaraa, J. J. Jarvis, and H. D. Sherali, *Linear Programming and Network Flows*. John Wiley & Sons, January 1990.
[4] J. Evans, "The simplex method for integral multicommodity networks," *Naval Research Logistics Quarterly*, vol. 25, pp. 31–37, Mar 1978.
[5] Z. Wang, *Internet QoS: architectures and mechanisms for Quality of Service*. Academic Press, 2001.
[6] J. Moy, "OSPF Version 2," RFC 2328 (Standard), April 1998. [Online]. Available: http://www.rfc-editor.org/rfc/rfc2328.txt
[7] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," *IEEE Communications Magazine*, vol. 40, no. 10, pp. 118–124, Oct 2002.
[8] A. Farago, B. Szviatovszki, and A. Szentesi, "Inverse optimization in high-speed networks," *Discrete Applied Mathematics*, vol. 129, no. 1, pp. 83–98, June 2003.
[9] G. Rétvári, J. J. Bíró, and T. Cinkler, "On Shortest Path Representation," *IEEE/ACM Transactions on Networking*, vol. 15, no. 6, pp. 1293–1306, Dec. 2007.
[10] B. Fortz and M. T. At, "Increasing internet capacity using local search," *Computational Optimization and Applications*, vol. 29, pp. 13–48, 2004.
[11] G. Rétvári and T. Cinkler, "Practical OSPF traffic engineering," *IEEE Commununications Letters*, vol. 8, pp. 689–691, November 2004.
[12] B. Fortz and M. Thorup, "Internet Traffic Engineering by Optimizing OSPF Weights," in *Proc. IEEE INFOCOM*, 2000, pp. 519–528.
[13] Z. Wang, Y. Wang, and L. Zhang, "Internet traffic engineering without full-mesh overlaying," in *Proceedings of INFOCOM 2001*, vol. 1, April 2001, pp. 565–571.
[14] A. Sridharan, C. Diot, and R. Guérin, "Achieving near-optimal traffic engineering solutions for current OSPF/IS-IS networks," in *Proceedings of INFOCOM 2003*, vol. 2, March 2003, pp. 1167–1177.
[15] D. Xu, M. Chiang, and J. Rexford, "Link-state routing with hop-by-hop forwarding can achieve optimal traffic engineering," *IEEE/ACM Trans. Netw.*, vol. 19, no. 6, pp. 1717–1730, 2011.
[16] V. Vassiliou, "High speed multimedia and multiservice networks: Traffic enginnering (slides)," available online: http://www.cs.ucy.ac.cy/courses/EPL420/Slides/MPLS-TE.pdf.
[17] Cisco Systems, Inc., "Troubleshooting Load Balancing Over Parallel Links Using Cisco Express Forwarding," available online: http://www.cisco.com/image/gif/paws/18285/loadbal_cef.pdf, August 2005, Document ID: 18285.
[18] Cisco Systems, Inc, "Cisco Nexus 7000 Series NX-OS Unicast Routing Command Reference," available online: http://www.cisco.com/en/US/docs/switches/datacenter/sw/6_x/nx-os/unicast/command/reference/l3_cmd.pdf, August 2012, Section: maximum-paths (OSPF).
[19] Current Analysis, Inc., "Product Assessment: Ericsson - SmartEdge Series," available online: http://archive.ericsson.net/service/internet/picov/get?DocNo=13/28701-FGB101647&Lang=EN&HighestFree=Y, July 2009.
[20] Juniper Networks, Inc., "E-series and JUNOSe Documentation," available online: http://www.juniper.net/techpubs/en_US/junose9.3/information-products/topic-collections/command-reference-a-m/maximum-paths.html, 2010, Section: commands/maximum-paths.
[21] "LEMON – Library for Efficient Modeling and Optimization in Networks, version 1.2.3." [Online]. Available: http://lemon.cs.elte.hu/
[22] "GLPK: GNU Linear Programming Kit, version 4.43." [Online]. Available: http://www.gnu.org/software/glpk/
[23] J. Suurballe, "Disjoint paths in a network," *Networks*, vol. 4, no. 2, pp. 125–145, 1974.
[24] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson, "Inferring link weights using end-to-end measurements," in *ACM IMC*, 2002, pp. 231–236.
[25] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The Internet Topology Zoo," http://www.topology-zoo.org.