On the Representability of Arbitrary Path Sets as Shortest Paths: Theory, Algorithms and Complexity

Gábor Rétvári, Róbert Szabó, József J. Bíró

High Speed Networks Laboratory, QoSIT Laboratory
Department of Telecommunications and Media Informatics
Budapest University of Technology and Economics*
H-1117, Magyar Tudósok körútja 2., Budapest, HUNGARY
{retvari,robert.szabo,biro}@tmit.bme.hu

Abstract. The question, whether an optional set of routes can be represented as shortest paths, and if yes, then how, has been a rather scarcely investigated problem up until now. In turn, an algorithm that, given an arbitrary set of traffic engineered paths, can efficiently compute OSPF link weights as to map the given paths to shortest paths may be of huge importance in today's IP networks, which still rely on legacy shortest-path-first routing protocols. This article establishes the fundamental theory and algorithms of shortest path representability, and concludes that in general it is much more difficult task to compute shortest path representable paths than to actually calculate link weights for such paths.
Keywords: traffic engineering, routing, linear programming, OSPF

1 Introduction

Most of today's Traffic Engineering (TE, [1]) proposals require the deployment of expensive routing and traffic forwarding hardware and software. On the other hand, ISPs have huge installation base of routers running legacy routing protocols like OSPF (Open Shortest Path First, [2]) or IS-IS (Intermediate-System-to-Intermediate-System). Both OSPF and IS-IS rely on shortest-path-first routing, i.e., there is an administrative weight associated with network links, and, for a given destination IP address prefix, the routing protocol uses the shortest aggregate cost path to that destination. The network operator manipulates routing by setting the administrative link weights appropriately. Usually, optional load balancing by ECMP (Equal-Cost-MultiPath) is also available, where traffic is split roughly evenly amongst multiple shortest paths, if such paths exist. Depending on the choice of the manufacturer, ECMP may implement per packet, per destination or per source-destination pair load distribution using round-robin or some hashing technique.

* This work was supported by the Ministry of Education, Hungary under the reference No. IKTA-0092/2002.

Hence, it is an easy-to-deploy and overly cost-effective solution to implement traffic engineering on top of OSPF while retaining existing routing equipment. In such an architecture, a suitable Traffic Engineer (*i*) participates in OSPF signaling to learn routing information, (*ii*) assigns paths for each session, (*iii*) computes link weights as to assure that the link weights reflect the assignment of paths (i.e., all paths, which are assigned for a particular session are shortest paths for that session) and (*iv*) distributes the selected link weights back to OSPF routers. However, this solution is almost certainly sub-optimal due to the inherent limitation of ECMP, which restricts load balancing to equal splitting.

The foundations of OSPF traffic engineering are laid down by [3] and [4]. An unpublished work [5] of the same authors shows that it is NP hard to compute link weights, as to assure that the resultant set of shortest paths fulfill some useful traffic engineering criteria. Therefore, the authors propose a local search heuristic achieving nearly optimal routing in some cases. However, the applicability of the algorithms proposed is restricted to the long-term process of network dimensioning. This is because of the running time of these algorithms, which may amount to hours even in a middle sized network. In contrast, online traffic engineering requires rapid algorithms to assure quick adaptation to topology changes or management controls. The authors also missed to identify, whether in path selection or in shortest path representation hides the real origin of exponential complexity.

To the best of our knowledge, the outstanding paper of Wang *et al.* [6] has been the only work dealing with shortest path representability up until now. According to their definition, a set of paths is *shortest path reproducible* if there exists a positive valued weight set based on which all the paths in the set are shortest paths. They establish the sufficient and necessary condition of shortest path representability and conclude that a set of paths is either loopy, and therefore is of negligible interest to traffic engineering, or it is shortest path reproducible. This precious work (and some derivatives, e.g., [7]) disproved the common belief of many researchers that shortest-path-first routing is, by nature, useless to traffic engineering.

It is of extreme importance to understand that the definition of shortest path representability and the implied linear programming solution can only guarantee that the selected paths are reproduced as shortest paths. Though, it claims nothing about other paths. Therefore, a path that was originally not designated for data forwarding may be given small cost and so, be introduced into routing. The traffic engineer does not have total control over routes and may experience unwanted interference caused by the additional paths, which he or she did not even consider to use. This may very well amortize the overall performance of the network. In fact, we can show that depending on the actual topology of the network and distribution of source-destination pairs the worst case performance might degrade to an arbitrary small fraction of the optimal performance due to unintended interference. This happens regardless of using ECMP or not.

Consider Figure 1. The network consists of N identical blocks with an alternate path circumventing these blocks. All link capacities and weights equal to 1.

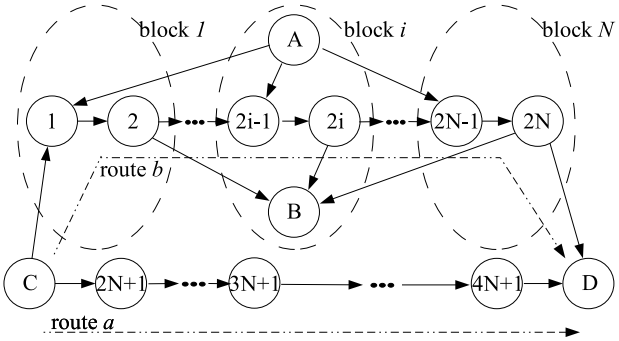

Fig. 1. Sample graph topology. All link capacities and weights equal to 1 and there are N and 1 units of demand for $A \to B$ and $C \to D$, respectively

There are two sessions (A, B) and (C, D) communicating over the network, willing to use N and 1 units of bandwidth, respectively. In an optimal setting, the traffic engineer may select the circumventing $C \to D$ path (route a in the figure) of hop count $2N+2$, and let (A, B) demands flow through the N blocks, one unit of demand per block. The optimal throughput is therefore $N + 1$. An obvious way to achieve this would be to set the weight of link $(C, 1)$ to a high value. However, traditional shortest path representation methods can not guarantee this, therefore, route b (also of length $2N+2$) is often introduced unintentionally into data forwarding[1]. In the worst case, all (C, D) communication may use route b completely blocking (A, B). This causes the performance to degrade to some $\frac{1}{N+1}$ fraction of the optimal value. If ECMP is not used, then, depending on the actual implementation of the shortest-path-first routing engine, in around half of the cases route a and in the other half route b would be selected, In average, this causes half of (C, D) traffic to interfere with (A, B) traffic. Observe that the very same situation rises with ECMP, too. Asymptotically, the total throughput degrades to the half of the optimal performance in the average case.

Yet another important thing to know about the definition of shortest path representability proposed in [6] is that it provides no means to avoid multi-path routing. Deciding whether or not multi-path routing is a beneficial feature to have is completely beyond the scope of this paper. On the one hand, multi-path routing promises load balancing and may yield higher performance and network utilization. On the other hand, multi-path routing introduces a huge amount of uncertainty compared to single-path routing, ranging from the actual implementation of the ECMP splitting algorithm to the fact that in the presence of equal-cost paths, it is totally impossible to predict, which particular path a certain connection will take. This is completely undesirable in some cases (e.g., it precludes call admission control to assess the route of a connection prior to actually instantiating it in the network). By all means, it may be advantageous

[1] Note that as of our favorite open source linear programming toolkit, GLPK, the optimal solution of ILP-SPR almost always yields this "bad" configuration.

to set link weights as to avoid multiple paths of a source-destination pair to have the same cost. To this end, [8] extends the local search heuristic proposed in [3], however, the proposed method is neither exact nor rapid.

In this paper we introduce the notions of explicit and unique shortest path representation to avoid the use of unintended paths. In Section 2 we give the basic mathematical formulation and definitions. As far as we know, this is the first time that strictly combinatorial algorithms to verify explicit shortest path representability are defined. The algorithms and some theoretical background is discussed in Section 3. Section 4 reveals the complexity of path assignment and gives simple approximations to the NP hard problem. Section 5 briefly outlines related simulation studies, and finally, Section 6 concludes our work.

2 Mathematical Formulation

Let $G(V, E)$ be a directed graph, formed by the set of nodes N ($|N| = n$) and the set of edges E ($|E| = m$). Let K denote the set of source-destination pairs (s_k, d_k), which are referred to as *sessions* for short. Let $\mathcal{P}^{s_k \to d_k}$ be the set of all paths that connect a particular source-destination pair (s_k, d_k). Our task then is to explicitly represent a given subset $\mathcal{P}^k \subseteq \mathcal{P}^{s_k \to d_k}$ as shortest paths. A path $P \in \mathcal{P}^k$ of length L_P is defined by its consecutive edges: $P := \{(v_i, v_{i+1}) \in E : i = 1 \dots L_P - 1, v_1 = s_k, v_{L_P} = d_k\}$. We assume that there is a positive valued weight $w_{ij} \in \mathcal{Z}^+$ associated with each edge (i, j), and we let the aggregate cost of a path P be $W(P) = \sum_{(i,j)\in P} w_{ij}$. Furthermore, let p_k denote the number of paths for session k (i.e., $p_k = |\mathcal{P}^k|$), let $\mathcal{P}$ be the set of all designated paths (i.e., $\mathcal{P} = \bigcup_{k\in K} \mathcal{P}^k$) and let the aggregate cost of a path P over link weights w_{ij} be $W(P) = \sum_{(i,j)\in P} w_{ij}$.

Now we introduce the notion of path-graphs, which will be heavily used throughout this paper. Let n_{ij}^k be the number of paths of session k traversing link (i, j) and $n_{ij} = \sum_{k\in K} n_{ij}^k$ be the number of all paths using that link. Then, the path-graph $G_{\mathcal{P}}$ induced by a path set $\mathcal{P}$ is a special network, which includes all edges of all paths of $\mathcal{P}$ and the capacity of the edges equals to the number of paths in $\mathcal{P}$ using that link. We also let the demand t_k for session k be $t_k = p_k$. Formally, a path-graph $G_{\mathcal{P}}$ is a network on $G(V, E)$, such that the capacity of a link (i, j) is given by $u_{ij} = n_{ij}$ and all zero capacity links and zero degree nodes are removed from the network. Observe that a path set unambiguously determines the corresponding path-graph, though, the reverse is not necessarily true. This is because a path-graph may contain additional paths, which are formed by the concatenation of some sub-paths of the original path set $\mathcal{P}$. This parallels with the property of shortest paths that if $a \to b$ and $b \to c$ are shortest paths, then $a \to b \to c$ is also a shortest path. Note that in this case, we consider these additional paths to belong to $\mathcal{P}$, too. In other words, a set of paths is said to include a particular path if it includes all edges of that path.

According to [6], a path set $\mathcal{P}$ is shortest path reproducible, if there exists a positive weight setting $\mathcal{W} = \{w_{ij}\} : w_{ij} > 0$, such that all paths in $\mathcal{P}$ are shortest paths over $\mathcal{W}$:

Definition 1 (SPR). *A path set $\mathcal{P}$ is shortest path representable, if there exists a positive weight setting $\mathcal{W}$, such that for all $P' \in \mathcal{P}^k$ it holds that*

$$\forall P \in \mathcal{P}^{s_k \to d_k} \setminus \mathcal{P}^k : W(P') \leq W(P) \tag{1}$$

for every session $k \in K$.

In this context, verification of shortest path representability of a particular path set $\mathcal{P}$ and the actual link weights implementing $\mathcal{P}$ can be given by solving the following integer linear program (ILP-SPR) over the path-graph:

$$\max \textstyle\sum_{k \in K} p_k \pi^k_{d_k} - \sum_{(i,j) \in E} n_{ij} w_{ij} \tag{2}$$

$$\text{s.t.} \quad \pi^k_j - \pi^k_i \leq w_{ij} \qquad \forall (i,j) \in E, \forall k \in K \tag{3}$$

$$w_{ij} \geq 1 \qquad \forall (i,j) \in E \tag{4}$$

Observe that this definition tells nothing about paths outside of $\mathcal{P}^k$. Such paths are either shortest paths or not. In other words, the representation is not explicit. As shown in the previous section, this may lead to undesirable interference amortizing the overall performance of the network. To eliminate this shortcoming we introduce the notion of *explicit shortest path representation* in the following way:

Definition 2 (eSPR). *A path set $\mathcal{P}$ is explicitly shortest path representable, if there exists a positive weight setting $\mathcal{W}$, such that for all $P' \in \mathcal{P}^k$ it holds that*

$$\forall P \in \mathcal{P}^{s_k \to d_k} \setminus \mathcal{P}^k : W(P') < W(P) \tag{5}$$

for every session $k \in K$ (note the strict inequality!).

What makes the difference is that eSPR explicitly prohibits a path outside of the desired path set $\mathcal{P}^k$ to become shortest path. Observe that any eSPR path set also fulfills Definition 1.

In certain situations, it is important to avoid having multiple different parallel routes for any sessions. Therefore, it is a plausible idea to define some sorts of *uniqueness* of routing by definitely precluding the existence of multiple equal-cost shortest paths between any source-destination pairs:

Definition 3 (uSPR). *A path set $\mathcal{P}$ is uniquely reproducible as shortest paths, if there exists a positive weight setting $\mathcal{W}$, such that for every session $k \in K$ there is exactly one path P'_k, for which it holds that*

$$\forall P \in \mathcal{P}^{s_k \to d_k} \setminus \{P'_k\} : W(P'_k) < W(P) \ . \tag{6}$$

Wang *et al.* propose a linear program to verify and perform shortest path representation [6], though, they do not seem to recognize that the resultant representation will be neither explicit nor easily implementable in network devices. In the sequel, we focus on the more difficult problem of eSPR instead of SPR and introduce some simple sufficient conditions to both explicit and unique shortest path representation.

3 Explicit and Unique Shortest Path Representation

In order to provide further insight into the relation of SPR and eSPR, first, we borrow some basic results of network flow theory and linear programming [9] [10]. Paths in a path set $\mathcal{P}$ define shortest paths if and only if there exist node potentials $\pi_n : n \in V$ and positive link weights $w_{ij} : (i,j) \in E$, such that:

$$\forall P \in \mathcal{P}, \forall (i,j) \in P : \pi_j - \pi_i = w_{ij} \tag{7}$$

$$\forall P \notin \mathcal{P}, \exists (i,j) \in P : \pi_j - \pi_i < w_{ij} \tag{8}$$

The cost of an $s \rightarrow d$ shortest path is given as $W(P) = \sum_{(i,j)\in P} w_{ij} = \pi_d - \pi_s$.

An intriguing question to investigate is to assess, under which conditions a shortest path representable path set is also explicitly shortest path representable. As it turns out, the two concepts of shortest path representability are identical under some surprisingly mild assumptions. This is formulated in the following important result:

Theorem 1. *Let $\mathcal{P}$ be a path set, such that for all $k \in K$ there exists an $s_k \rightarrow d_k$ path in the path-graph $G_{\mathcal{P}}$ of $\mathcal{P}$. Then, $\mathcal{P}$ is shortest path representable if and only if it is explicitly shortest path representable.*

Proof. *eSPR $\Rightarrow$ SPR* is obvious in light of the fact that an explicit shortest path representation immediately conforms to the SPR definition.

SPR $\Rightarrow$ eSPR: we give a constructive proof by presenting an algorithm, which turns a SPR into an eSPR in polynomial time. Given a path set $\mathcal{P}$ let the path-graph induced by $\mathcal{P}$ be $G_{\mathcal{P}}(V_{\mathcal{P}}, E_{\mathcal{P}})$. From Definition 1 we have that over a proper SPR weight set $\mathcal{W}$ and node potentials π_n, all $s_k \rightarrow d_k$ paths in $G_{\mathcal{P}}$ are shortest paths. What we need to assure is that no paths outside of $G_{\mathcal{P}}$ are also shortest paths. Consider the following simple modification of $\mathcal{W}$: For some edge $(i,j) \notin E_{\mathcal{P}}$ let $w_{ij} = W_{MAX}$, where W_{MAX} is defined as:

$$W_{MAX} = \max_{k\in K}(\pi_{d_k} - \pi_{s_k}) + 1 \tag{9}$$

Now, we make the following observations:

- Let $n \in V_{\mathcal{P}}$ be a node in the path-graph. By assumption, we have that for any session $k \in K$ there exists at least one $s_k \rightarrow n$ path, which lies completely inside $G_{\mathcal{P}}$. Since this path is a shortest path and its length is not affected by the weight of any $(i,j) \notin E_{\mathcal{P}}$, the node potentials $\pi_n : n \in V_{\mathcal{P}}$ are invariant with respect to the above modification of the link weights. The same applies to the value of W_{MAX}, itself.
- Furthermore, (9) is constructed as to assure that W_{MAX}, and as such, any path containing at least one edge of weight W_{MAX} is longer than the longest one of all $s_k \rightarrow d_k$ shortest paths.

Then, one can set the weight of all the links outside of $E_{\mathcal{P}}$ to W_{MAX} to obtain an explicit shortest path representation. This can be done in $O(m)$ time. To prove

the correctness of this algorithm, let P_1 be any optional $s_k \rightarrow d_k$ path for some session $k \in K$, such that $P_1 \notin \mathcal{P}$. Hence, P_1 traverses some edge $(i,j) \notin E_{\mathcal{P}}$ (otherwise, it would lie completely in $G_{\mathcal{P}}$ and therefore, by definition, it would belong to $\mathcal{P}$). So $w_{ij} = W_{MAX}$. Furthermore, let P_2 be a $s_k \rightarrow d_k$ path inside $G_{\mathcal{P}}$, i.e., $\forall (u,v) \in P_2 : (u,v) \in E_{\mathcal{P}}$. Note that, by assumption, one can always find such P_2 path. Additionally, the algorithm leaves the node potentials, and as such, the length of any path in $G_{\mathcal{P}}$ intact. Thus, for the length of paths P_1 and P_2 we have that $W(P_2) < W_{MAX} \leq W(P_1)$. Since the strict inequality holds for any P_1 outside of $G_{\mathcal{P}}$ and any P_2 inside $G_{\mathcal{P}}$ we conclude that the modified weight set implements explicit shortest path representation (cf. (7)-(8)). This completes the second part of the proof. □

The significance of the above theorem is two-fold, First, as shall be shown by the simulation results in Section 5, explicit shortest path representation generally improves the performance of OSPF traffic engineering as it avoids unnecessary and adverse interference caused by unintentional paths. Moreover, proving SPR is usually simpler than proving eSPR. Hence, Theorem 1 assures that the resultant weight set can be transformed in polynomial time into one, which implements explicit shortest path representation. Henceforward, we restrict our discussions to path sets, which satisfy the assumption of Theorem 1, i.e., which contain at least one path for each session.

The linear program ILP-SPR does not provide easy way to determine, whether or not a particular set of paths is shortest path representable. It requires the rapid solution of a potentially large scale linear program, which may very well fall beyond the capabilities of today's network devices. Therefore, in the sequel, we show some easy-to-check conditions to test for eSPR and uSPR, respectively.

Lemma 1 (Sufficient condition for uSPR). *A path set $\mathcal{P}$ is uniquely representable as shortest paths if the path graph $G_{\mathcal{P}}$ induced by $\mathcal{P}$ is a directed forest.*

Proof. Given that $G_{\mathcal{P}}$ is connected with respect to source-destination pairs (s_k, d_k) and it forms a directed forest, by definition, there is only one path between any two nodes. Hence, arbitrary positive setting of the link weights will conform to (8) as long as the weight setting W_{MAX} defined in (9) is respected for the links, which do not reside in $G_{\mathcal{P}}$. This can be done in $O(m)$ time. □

Lemma 2 (Sufficient condition for eSPR 1.). *A path set $\mathcal{P}$ is explicitly representable as shortest paths if the path graph $G_{\mathcal{P}}$ induced by $\mathcal{P}$ is acyclic (i.e., it does not contain any directed circles).*

Proof. We construct a simple shortest path representation of $\mathcal{P}$, which then, according to Theorem 1 can be easily converted to an explicit shortest path representation. From graph theory, we know that every acyclic graph $G(V,E)$ possesses one or more *topological ordering*. A topological ordering is a labeling $order(n)$ of the nodes $n \in V$, such that every edge joins a lower-labeled node to a higher-labeled node, i.e., $\forall (i,j) \in E : order(i) < order(j)$. It is fairly easy to see

that any topological ordering of the path graph $G_{\mathcal{P}}$ induced by $\mathcal{P}$ defines suitable node potentials, and hence, link weights in the form: $w_{ij} = order(j) - order(i)$. A weight set w_{ij}, defined in this way is, by definition, integer and positive valued. Thus, proper eSPR link weights can be computed in $O(m)$ time in this case. □

Note that verifying any of the above conditions takes $O(m)$ steps. This is the lower bound on the complexity of any link weight setting algorithm, since at least $O(m)$ steps are necessary to walk through all the links in the network. Also note that a set of paths may very well be shortest path representable even if its path graph contains directed circles. Therefore, the above condition is obviously not a necessary one. In order to catch a larger class of path graphs than acyclic graphs, we present yet another sufficient condition of eSPR, which is of considerably broader scope:

Lemma 3 (Sufficient condition for eSPR 2.). *Consider the single commodity flow problem (the so called mass-flow problem) derived from the original K-commodity flow problem in the following way. For every node $n \in V$, let the imbalance of n be $e(n) = \sum_{k \in K: n=s_k} p_k - \sum_{k \in K: n=d_k} p_k$. Find a minimum cost mass-flow instance that satisfies $e(n)$. This can be done in polynomial time by some combinatorial algorithm, e.g., by minimum mean-cycle cancellation in $O(n^2 m^3 \log n)$ time [9]. Then, a path set $\mathcal{P}$ is explicitly representable as shortest paths if for the aggregate cost C_{mass} of the optimal mass-flow:*

$$C_{mass} = \sum_{P \in \mathcal{P}} L_P \ , \tag{10}$$

where L_P is the length of path P.

Proof. Consider the the dual linear program instance I of ILP-SPR over $G_{\mathcal{P}}$. It is fairly easy to show that I is a minimum cost multi-commodity flow problem. From [6] we know that a path set $\mathcal{P}$ is shortest path representable if and only if the aggregate length $\sum_{P \in \mathcal{P}} L_P$ of $\mathcal{P}$ equals to the optimal objective function of I. Solve I and let the optimal objective value be C_I. It is straightforward that $C_{mass} \leq C_I$. Hence, if (10) holds, then $C_I = \sum_{P \in \mathcal{P}} L_P$ and $\mathcal{P}$ is optimal. □

In general, given a set of paths, one can either conclude that the path set is loopy or otherwise provide an explicit shortest path representation in strictly polynomial time. This implies that this is not the shortest path representation problem, but rather the determination of optimal paths, which hides the exponential complexity of OSPF traffic engineering. In fact, the next section confirms just this claim.

4 Complexity of Optimal Path Assignment

Now, we move on to investigate the complexity of optimal path selection and to show that in general, it is a hard task to compute eSPR path sets with respect to some reasonable traffic engineering optimization criteria. The reader is advised to the following discussion.

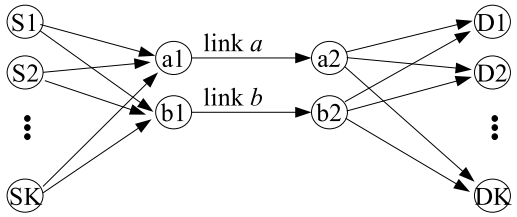

Fig. 2. Sample configuration with K sessions and demands $0 < t_k < 1$

Theorem 2. *Given a demand set $t_k : k \in K$ it is NP hard to compute a uniquely shortest path representable path set, such that all t_k demands are satisfied. In fact, it is also NP hard to even decide, whether or not the demand set can be satisfied along an uSPR path set.*

Proof. The transformation is from *2 bin packing*. Consider the sample configuration depicted in Figure 2. There are K sessions with demands $0 < t_k < 1$ and two link disjoint paths of capacity 1 from each source to each destination, one through link a and another through link b. Individual demands must be routed without any sorts of splitting to form a directed forest of a uSPR. Therefore, all $s_k \rightarrow d_k$ traffic t_k is either packed into link a or link b, alternatively. Thus, any uSPR path set in this setting also solves 2 bin packing. It is also NP hard to even decide, whether a particular demand set can be packed into the two bins or not. □

Note that it is also NP hard to determine the maximum number of sessions that can be satisfied, which would be the objective if one was to maximize the throughput of the network. The network dimensioning case is also NP hard (how many links to deploy in order to assure proper uSPR routing), since it maps to the minimum bin packing problem. Also note that the proof of Theorem 2 remains to be valid, if we let individual demands to be split between the two paths evenly, i.e., let half of the demand flow through link a and the other half through link b. Observe that such a path set is acyclic, and as such, it conforms to the condition of explicit shortest path representability given in Lemma 2.

Corollary 1. *In general, determining a path set that is explicitly shortest path representable and optimizes OSPF ECMP routing is NP hard.*

Despite of the intractable complexity of eSPR path selection, there are certain relaxations of the full-fledged problem, which are both easy to solve and may prove to be of substantial interest in some realistic scenario. The NP complete nature of OSPF-ECMP routing is closely coupled with the requirement that either flows are unsplittable (uSPR) or can only be split evenly (eSPR). Relaxing this requirement of unsplittable flows immediately yields polynomial approximate algorithms for eSPR path selection, such as the optimal routing or the minimum cost maximum throughput linear programs [9], [10].

uSPR path selection is a more difficult problem, because one must avoid any branching of the optimal paths in this case. Therefore, some more relaxation is necessary. First, it is plausible to let $\forall k \in K : t_k = 1$, since most of today's networking architectures do not provide means for a user to specify his or her demand size. In addition, the scope of the routing information retrievable from OSPF link state information is currently limited to the actual topology of the network. This gives rise to a unit-demand-unit-capacity relaxation of the original problem. In this setting, the integer linear program below (if solvable) provides a path set $\mathcal{P}$ that can be uniquely represented as shortest paths:

$$\max \sum_{k \in K} t_k - \alpha \sum_{k \in K} \sum_{(i,j) \in E} X_{ij}^k \tag{11}$$

$$\sum_{j:(i,j) \in E} X_{ij}^k - \sum_{j:(j,i) \in E} X_{ji}^k = \begin{cases} t_k & \text{if } i = s_k \\ -t_k & \text{if } i = d_k \\ 0 & \text{otherwise} \end{cases} \quad \forall k \in K, \forall i \in V \tag{12}$$

$$\sum_{k \in K} \sum_{j:(i,j) \in E} X_{ij}^k \leq 1 \qquad \forall i \in V \tag{13}$$

$$X_{ij}^k \in [0,1],\ t_k \in [0,1] \qquad \forall k \in K, \forall (i,j) \in E \tag{14}$$

The objective function (11) maximizes the overall throughput of the network, while minimizing the aggregate flow to avoid loops (α is a suitably small constant). (12) requires flow conservation. It is assured that the resultant path set (which comes in the form $\exists k : X_{ij}^k > 0 \Rightarrow (i,j) \in \mathcal{P}$) is uSPR, since (13) lets only one unit of flow to emanate from any node. Thus, the path graph consists of isolated paths for some sessions, which together form a directed forest. Finally, (14) keeps per-session traffic t_k and the link flow X_{ij}^k integer.

5 Simulation Studies

In this section, we briefly outline some results of our simulation studies that demonstrate the benefits of rendering the representation explicit. We used the *BRITE* tool [11] with the *router-level Waxman-model* ($\alpha = 0.15, \beta = 0.2, m = 3$) to generate a sequence of increasing sized realistic random graphs. We tried to keep the load constant throughout the sequence by setting the number of sessions, request intensity, average demand size and average holding time as to assure that the generated traffic keeps track with the growing capacity of the consecutively increasing networks. We used a call level OSPF-ECMP simulator to compare the performance of distance-vector routing (minimum hop-count routing, *MINHOP*), shortest path representation of the optimal paths generated by the maximum throughput relaxation (*SPR*) and the *explicit* representation (using Theorem 1) of the very same paths (*eSPR*). The results presented below are averaged over 30 graph sequences.

Figure 3 depicts the average number of ECMP routes as the function of the network size. MINHOP may only accidently create ECMP paths, while the solution of the maximum throughput relaxation intentionally, though, being just a relaxation of the NP hard path selection problem, sub-optimally forms ECMP

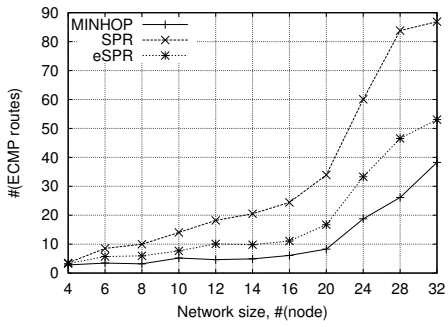

Fig. 3. Average number of ECMP routes as the function of network size

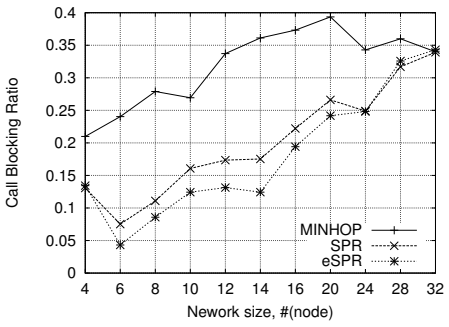

Fig. 4. Average call blocking ratio as the function of the network size

paths. However, compared to eSPR, which exclusively implements these paths, almost every second path formed by SPR is unintended. The average call blocking ratio depicted in Figure 4 insists that this nature of SPR is indeed leading to interference owing to the additional paths. The difference amounts to some 5-8% in average, which, in individual cases may be highly significant gain implied by eSPR. The figure also underlines the superiority of OSPF traffic engineering in comparison to traditional MINHOP routing.

6 Conclusions

This paper focuses on shortest path representation, a question of crucial importance in the majority of today's IP networks, which still rely on legacy shortest-path-first routing protocols. Our most important contribution to the groundbreaking work of [6] in this field comes from the recognition that it is not enough to blindly map the desired paths to shortest paths. We provided strong theoretical and practical evidence that if one can not exclude unintended paths from becoming shortest paths, he or she risks substantial amortization of the network revenue. We showed that, under reasonable assumptions, every non-explicit shortest path representation can be turned into an explicit one in polynomial time. As the explicit representation is a stronger and more useful one, we propose to use it instead of the non-explicit case. To ease this, we gave some novel sufficient conditions to test for SPR, which are, in contrast with prior work, strictly combinatorial. We also introduced uniqueness and proposed an exact method to compute uSPR link weights. Finally, we dealt with the problem of the selection of traffic engineered paths subject to eSPR or uSPR, and concluded that this problem is NP hard. As our major contribution, we concluded that actual shortest path representation is an easily tractable problem, and instead, optimal eSPR/uSPR path selection hides the real origin of exponential complexity. Therefore, we plan to take further efforts in this field.

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. J. Moy, "OSPF Version 2." RFC 2328, April 1998.
3. B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," *IEEE Communications Magazine*, vol. 40, pp. 118–124, Oct 2002.
4. B. Fortz and M. Thorup, "Optimizing OSPF/IS-IS weights in a changing world," *IEEE Journal of Selected Areas in Communications*, vol. 20, pp. 756–767, May 2002.
5. B. Fortz and M. Thorup, "Increasing internet capacity using local search," 2000. unpublished manuscript, http://www.research.att.com/~mthorup/PAPERS/or_ospf.ps.
6. Z. Wang, Y. Wang, and L. Zhang, "Internet traffic engineering without full-mesh overlaying," in *Proceedings of INFOCOM 2001*, April 2001.
7. A. Sridharan, C. Diot, and R. Guérin, "Achieving near-optimal traffic engineering solutions for current OSPF/IS-IS networks," in *Proceedings of INFOCOM 2003*, March 2003.
8. M. Thorup, "Avoiding ties in shortest path first routing," 2001. unpublished manuscript, http://www.research.att.com/~mthorup/PAPERS/ties_ospf.ps.
9. R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Englewood Cliffs, NJ, 1993.
10. M. S. Bazaraa, J. J. Jarvis, and H. D. Sherali, *Linear Programming and Network Flows*. John Wiley & Sons, January 1990.
11. A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: Universal topology generation from a user's perspective," Tech. Rep. 2001-003, 1 2001.