Available at www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

ELSEVIER

Computer Networks 42 (2003) 199–210

COMPUTER NETWORKS

www.elsevier.com/locate/comnet

On the solution of reroute sequence planning problem in MPLS networks

Balázs Gábor Józsa [a,b,*], Márton Makai [c,d]

[a] Traffic Analysis and Network Performance Laboratory, Ericsson Research Hungary, P.O. Box 107, H-1300 Budapest, Hungary
[b] High Speed Networks Laboratory, Department of Telecommunication and Telematics, Budapest University of Technology and Economics, H-1117 Magyar Tudósok Körútja 2, Budapest, Hungary
[c] Department of Operations Research, Eötvös University, H-1117 Pázmány Péter Sétány 1/C, Budapest, Hungary
[d] Communication Networks Laboratory, Eötvös University, H-1117 Pázmány Péter Sétány 1/A, Budapest, Hungary

Received 15 April 2002; received in revised form 6 November 2002; accepted 27 January 2003

Responsible Editor: J. Roberts

Abstract

This paper studies the problem of reroute sequence planning for label switched paths (LSPs) in multiprotocol label switching networks from both the theoretical and practical points of view. This issue arises when the set of LSPs is recalculated by a central path optimization tool to attain a better resource utilization in the network. In this case a sequence of LSPs has to be found for their one by one reconfiguration without service interruption, involving the constraint that the link capacities should not be violated at any time during the rerouting process. The underlying problem is related to discrepancy theory and it is NP-complete. The conditions of existence of any feasible reroute sequence are examined, and algorithms are described for solving the problem. Alternative solutions are also presented for the case when feasible solutions do not exist, finally the performance of these algorithms is investigated by empirical analysis.

Keywords: MPLS; Traffic engineering; Path optimization; Rerouting; Sequence planning

1. Introduction

Multiprotocol label switching (MPLS) is a technology developed for efficient forwarding of Internet protocol datagrams in core networks. Traffic flows in MPLS use label switched paths (LSPs) that are previously established by their source routers called label edge routers (LERs). In other words, the ingress-egress points of an LSP are LERs, while the other MPLS capable routers—that can only be transit nodes along the LSPs—are the label switching routers. One of the important benefits of MPLS is the support of various traffic engineering features that are very useful for Internet service providers (ISPs) to

* Corresponding author. Address: Traffic Analysis and Network Performance Laboratory, Ericsson Research Hungary, P.O. Box 107, H-1300 Budapest, Hungary.
E-mail addresses: balazs.jozsa@eth.ericsson.se (B.G. Józsa), marci@cs.elte.hu (M. Makai).

1389-1286/03/$ - see front matter
doi:10.1016/S1389-1286(03)00189-0

control the paths of LSPs in their networks, enabling an enhanced utilization of network resources, offering quality of service guarantees, and increasing network reliability [3,4]. MPLS can be extended by an information distribution component related to the bandwidth reservations, so that LERs can route constraint-based LSPs [13] using constrained shortest path first routing calculations for each LSP, i.e., the constrained LSP establishment procedure can be automated. Thus, an LSP can be routed on a path that differs from the default path selected by the interior gateway protocol, which is useful when the latter path contains any link whose available bandwidth is less than the bandwidth required by the given LSP.

In heavily loaded networks successive on-demand LSP establishment and deallocation actions may result in a set of LSPs where some paths are not the shortest possible ones, leading to poor resource utilization compared to the optimal state. Thus, global LSP optimization is proposed at certain time intervals (e.g., daily, weekly) to improve the network performance. A key feature of MPLS is to support the explicit routing of LSPs, which enables the ISPs to optimize the LSP placement globally with a central traffic engineering tool [14,21]. The calculated new paths should be routed *strictly* explicitly, i.e., the whole path of each LSP is completely determined. To avoid the interruption of traffic through the LSPs during the rerouting from the old to the new paths, the rerouting of an LSP involves the following steps. First, the new path of the LSP must be established while the traffic is still carried on the old path, then the traffic is switched to the new path, finally the old path is torn down. During this operation (while changing the old paths to the new ones according to a sequence) a problem may occur, namely, some of the LSPs may not be reroutable to their new paths as there might not be enough bandwidth on some links of these paths. Therefore, the *reroute sequence* should be planned before the rerouting action, with the result that the rerouting of the LSPs in the calculated sequence would be feasible, i.e., would not exceed any reservable bandwidth threshold. This sequence planning procedure can be realized as a new function of the traffic engineering tool that performs the global path optimization. In this paper the LSP reroute sequence planning (RSP) problem is investigated from the theoretical and the practical points of view.

The rest of the paper is organized as follows. In Section 2 the problem definition is given and previous work is discussed. Then in Section 3 the theoretical investigations are summarized. In Section 4 several heuristic algorithms are described for solving the RSP problem. The algorithms are analyzed in various network situations and numerical results are presented in Section 5. Finally, in Section 6 the conclusion is drawn.

2. Problem formalization

In this section the RSP problem is described in detail. After a short outline two equivalent formulations are given: the first one is a graph based description, while the other one is a vector based definition that is used in the theoretical part. Finally, previous work on RSP is discussed.

2.1. Outline of the problem

In the RSP problem the MPLS network is given with LSPs routed on their original (*old*) paths. Moreover, the optimized (*new*) paths of the LSPs are also known. The new paths are calculated by a global LSP optimizer (e.g., [14]) located in a central place of the network. The task is to reroute the LSPs from their old paths to their new paths, specifically, to find a feasible reroute sequence of the LSPs. The traffic on the LSPs should not be interrupted, so the new path of each LSP must be established before its old path is torn down. The "only" constraint is that the maximal reservable bandwidth of the links must not be exceeded at any moment of the rerouting process. In this paper we state two restrictions that make the RSP problem practically relevant. First, LSPs *cannot be split* into several paths. They must be rerouted entirely in a single step, because the split paths might result in a highly complex rerouting process. Moreover, *no temporary paths* can be used during the rerouting process, i.e., the LSPs must be

rerouted directly on their new paths in order to limit the number of rerouting steps.

2.2. Graph based definition

Now, the RSP problem is defined formally. Let the directed graph $G(V,E)$ represent the MPLS network with the n-element set of vertices V and the m-element set of directed edges $E \subseteq \{(u,v): u,v \in V, (u \neq v)\}$ corresponding to the nodes and links, respectively. The set of edges is endowed with a non-negative edge capacity function $c : E \to \mathbb{R}^+$ representing the total reservable bandwidth values of the links. A directed path P with source node u_0 and destination node u_w is defined as a sequence of w edges $\{e_1 = (u_0,u_1), e_2 = (u_1,u_2), \ldots, e_w = (u_{w-1},u_w)\}$. Let the structure of an LSP be described by $l_i = (s_i, d_i, b_i, P_i, Q_i)$, where P_i and Q_i are the old and new paths, both having source node s_i, destination node d_i, and required transmission bandwidth b_i. In order to define our problem, the k-element set of LSPs $\mathscr{L} = \{l_i: 1 \leqslant i \leqslant k\}$ is given, and the following notations are introduced: $b_{\max} := \max_{1 \leqslant i \leqslant k} b_i$, and L_i represents the reserved capacities on the edges after the ith rerouting action. It is assumed that the system of the old paths with the corresponding capacities is feasible as well as the system of the new paths, i.e., for each edge the given edge capacity $c(e)$ is not violated by the paths using that edge: $L_0(e) = \sum_{i:e\in P_i} b_i \leqslant c(e)$ and $L_k(e) = \sum_{i:e\in Q_i} b_i \leqslant c(e)$, $\forall e \in E$. It is also supposed that $P_i \neq Q_i$ $(1 \leqslant i \leqslant k)$ as LSPs with unchanged paths can be eliminated and the corresponding edge capacities can be decreased with the result that the equivalent problem without unchanged LSPs is obtained.

The goal in the RSP problem is to determine a reroute sequence (a permutation) $\pi = \{\pi_1, \pi_2, \ldots, \pi_k: \pi_i \in \{1,2,\ldots,k\}, i \neq j \Rightarrow \pi_i \neq \pi_j\}$ of LSPs that enables the LSP rerouting without exceeding the capacity constraints, i.e., $\sum_{i:e\in Q_{\pi_i}, i \leqslant t} b_{\pi_i} + \sum_{i:e\in P_{\pi_i}, i \geqslant t} b_{\pi_i} \leqslant c(e)$, $\forall e \notin P_{\pi_t} \cap Q_{\pi_t}, 1 \leqslant t \leqslant k$, and $\sum_{i:e\in Q_{\pi_i}, i \leqslant t} b_{\pi_i} + \sum_{i:e\in P_{\pi_i}, i \geqslant t} b_{\pi_i} - b_{\pi_t} \leqslant c(e)$, $\forall e \in P_{\pi_t} \cap Q_{\pi_t}, 1 \leqslant t \leqslant k$. As one can see, the capacity constraint expression depends on whether $e \in P_{\pi_t} \cap Q_{\pi_t}$, because the common edges of the old and new paths of an LSP should not be reserved twice during the rerouting [2]. To sum up, the first LSP to be rerouted is l_{π_1}, the ith one is l_{π_i}, and the last one is l_{π_k}.

2.3. Vector based definition

The above-defined RSP problem is related to *discrepancy theory* [8] in the following way. Let p_i and q_i be the incidence vectors—having m elements—of paths P_i and Q_i, respectively ($p_i(e) = 1$ if $e \in P_i$, otherwise $p_i(e) = 0$). Consequently, the rerouting of LSP l_i corresponds to the vector $v_i = b_i q_i - b_i p_i$ where each vector component represents the net change of capacity reservations. Therefore, defining the initial vector $L_0(\leqslant c)$ representing the initial capacity reservations in the network, and the set of vectors $\{v_i: 1 \leqslant i \leqslant k\}$, our task is to find a permutation $\pi = \{\pi_1, \pi_2, \ldots, \pi_k\}$ of vectors v_i so that their partial sums never exceed c for any vector component, i.e., the edge capacities are not violated at any point during the rerouting process. Formally, find a permutation $\pi = \{\pi_1, \pi_2, \ldots, \pi_k\}$ of $1,2,\ldots,k$ so that

$$L_0(e) + \sum_{i=1}^{t} v_{\pi_i}(e) \leqslant c(e) \quad \forall e \in E, \ \ 1 \leqslant t \leqslant k. \qquad (1)$$

One can easily see that this is an equivalent reformulation, and we deal with this approach in the theoretical investigations.

2.4. Previous work on reroute sequence planning

The RSP problem was introduced in [15], in which four heuristic algorithms were investigated on real-world backbone networks. Then the problem was extended to *protected* traffic flows, the issue of complexity was discussed and the heuristics were improved in [16]. It turned out in both papers that some RSP problem instances cannot be solved by these simple algorithms already at moderate (60%) network load (which means 40% spare capacity on average). These results serve as a motivation for further investigations from another point of view, namely, clarification is needed whether the problem instances had no feasible solutions or the algorithms were not able to find any. For this reason

theoretical investigations were performed and the problem instance generation was modified in such a way that ensures the existence of feasible solutions (see Section 5.1).

3. Theoretical investigations

In [16] it was proven that the RSP problem is NP-complete even in a ring of two edges. However, if the free capacities are large enough in the network, the existence of feasible reroute sequence can be asserted.

While talking about the theoretical results, for the sake of simplicity it is assumed that $L_0 = L_k$, i.e., the initial and the final capacity reservations are equal (in other words $\sum_{i=1}^{k} v_i = 0$), and they are denoted by L. This is not an important restriction because the results can be transformed to the general case, but using this assumption the formulas are significantly simpler.

If the initial capacity vector $c(e)$ satisfies $c(e) \geqslant 2L(e)$ for all $e \in E$, the rerouting can be performed in an arbitrary sequence without violating the edge capacities. The proof is very simple because expression (1) trivially holds as both path systems can be inserted simultaneously into the network while respecting the capacity constraints: $\sum_{i=1}^{k} b_i p_i(e) + \sum_{i=1}^{k} b_i q_i(e) = L(e) + L(e) \leqslant c(e)$. However, this bound is quite loose to use it in practice, because generally there are some edges whose reserved capacities exceed the half of their total reservable capacities (e.g., at 60% network load).

As L is necessary and $2L$ is trivially enough, we have to look for a theoretical bound between them. Although our theoretical investigations resulted in several theorems, understanding them needs sophisticated mathematical knowledge. The theorems are presented in Appendix A. To summarize our theoretical results, we found that L is asymptotically nearer in the sense that a lower bound $\approx L + C\sqrt{L}$ for the capacities will be sufficient for the rerouting (where C is a constant depending on the number of edges).

Approximation algorithms were constructed based on our theorems, but the preliminary test results showed that they had no practical relevance as their performances were near to the random order (RO) (see Section 5.2). They are omitted from this paper. This result was expected since the problem instances (see in Section 5.1) were so "tight" that they did not fit the derived capacity bounds.

4. Algorithms

In this section heuristic algorithms are described for solving the RSP problem. These were introduced in [15], and enhanced in [16]. In our previous works these algorithms were compared to the reference algorithm random sorting (RS), which builds up the reroute sequence by choosing the LSPs one by one randomly. The candidate LSPs in this selection are those LSPs that can be rerouted without violation if there are reroutable ones, otherwise all actually non-rerouted LSPs are considered. This means that RS differs from the completely RO as it selects from the reroutable LSPs. In the previous works RS was not significantly worse than the other heuristics. RO is therefore investigated in this paper as a second reference.

4.1. Outline of the algorithms

The base of the heuristic algorithms is an iteration in which one LSP is selected and then rerouted. An algorithm finishes when all LSPs have been selected, therefore the reroute sequence is built up one by one greedily. The key element is the selection of the LSP to be actually rerouted. The heuristic algorithms differ in the selection of the subsequent LSP. It is a common feature of these approaches (as of RS) that the selection is based on reroutable LSPs if there are any, otherwise such an LSP is selected whose rerouting violates some of the edge capacities but the violation is aimed to be kept at a certain minimum. The algorithms assign a *greedy utility* value o_i to each candidate LSP l_i—if l_i is already rerouted $o_i = -\infty$—in each iteration, and the chosen LSP is the one that has the actually greatest greedy utility

value. The assignment of the greedy utility value o for the candidate LSPs is done as follows.

4.2. Minimal violation (MV)

The method MV is the simplest greedy method (apart from RS). It calculates for each non-rerouted LSP l_i the greatest capacity violation on its new edges (that are distinct from every old edge) if it is rerouted: $o_i = -\max_{e \in Q_i \setminus P_i} \{b_i + R(e) - c(e)\}$ where $R(e)$ is the actual reserved capacity on edge e. If some edges would be violated in case of rerouting LSP l_i, $o_i < 0$, otherwise $o_i \geqslant 0$. The idea behind this ranking is to decrease the minimal free capacity on the edges in the slightest degree possible or in case of inevitable capacity excess, effecting minimal amount of violation on the edge capacities.

4.3. Maximal freeing (MF)

The approach MF uses a capacity value $A(e)$ to be routed for each edge e, i.e., the summed bandwidth of such LSPs that has to be allocated to the given edge in the subsequent rerouting steps. Formally, $A(e) = \sum_{j: e \in Q_j \setminus P_j} b_j$ for all j where l_j is not rerouted yet. Here value o_i represents the total amount of capacity that is to be freed on the old edges of l_i for the subsequent LSP reroutings: $o_i = \sum_{e \in P_i \setminus Q_i} \{A(e) - (c(e) - R(e))\}$. The idea of this rule is to prefer those LSPs at the selection which contain edges (in their old paths) that are present in many new paths of non-rerouted LSPs and have relatively few actual free capacities.

4.4. The most reroutable (MR)

In this method o_i is calculated to represent the number of LSPs that can be rerouted without capacity violation after the successful rerouting of LSP l_i. Using this approach after selecting an LSP, in the next step the algorithm can select from the maximal number of reroutable LSPs. In this case more than one LSP may have the same greatest value o_i, consequently the utility value is modified in the following way: the value of o_i is decreased by the summed ratio of the number of non-reroutable edges and the total number of edges to be rerouted for all non-rerouted LSPs after l_i has been rerouted. The important benefit of this approach is that it looks forward one rerouting step. On the other hand, it has relatively larger computational complexity compared to the previous algorithms, thus its application is more time consuming.

4.5. Enhancements

The enhancement of the above-described greedy methods is based on the following. The normal RSP problem—rerouting LSPs from paths P to Q—is equivalent to the reverse problem when we want to reroute each LSP i from path Q_i to P_i, and the initial and final capacity reservations are inverted. The reason for this is the following: if there is a feasible sequence for the reverse problem, the reverse sequence is an appropriate solution for the normal problem and this is true inversely. Now, three possible improvements of the above-described greedy algorithms are presented:

(1) The simplest improvement is trying to solve both normal and reverse problems. First, the easier problem is considered, which is generally rerouting from that starting state where the total free capacity is less—this is typical in real situations because the goal of path optimization is to increase the total free capacity in the network.
(2) In the second variant of the algorithms the sequence is built up from both ends, i.e., LSPs to be rerouted are selected for the normal and reverse problems alternately.
(3) The most complex modification is using backtracking in the algorithms. The algorithm starts and if it gets stuck, i.e., there is no LSP to be rerouted without capacity violations, it deletes a part of the latest inserted LSPs from the sequence (and reroutes the LSPs to their old paths) and continues with the reverse problem. If it gets stuck again it steps back again and changes direction. To avoid infinite loops: at every back-step phase fewer LSPs are deleted from the sequence than the number of LSPs rerouted in the previous phase. In the current implementation the ratio of deleted LSPs from the sequence and

inserted LSPs in the last phase is set to 95% (that value was determined by a fine-tuning process).

4.6. Computational complexity

The computational time of the above-described algorithms consists of two main parts:

- checking whether the particular LSP is reroutable, and
- calculation of greedy utility value.

The first check is very simple: the edges of the new path should be examined to see whether all of them have enough free capacity. On the other hand, the complexity of the utility value calculations of the methods are quite different. While RS assigns a value in constant time, MV and MF perform operations "number of new edges" times. MR is the most complex (since it looks forward one rerouting step) and all new edges are checked for all non-rerouted LSPs. The overall computational time is the time of reroutability check and utility value calculation multiplied by $(k \cdot (k-1)/2) - 1$, because the base of the number of candidate LSPs decreases by one in each step. The order of magnitude of time consumption on our test networks having several tens of nodes is a few seconds using MR (on a computer having 450 MHz processor and 1 GByte memory). However, if the third enhancement is used, the running time of the algorithms (mostly of MR) can be enormous theoretically in worst cases, while the first and second enhancements hardly influence it. At the current back-step ratio setting (95%) the running time of MR during the preliminary tests was a few minutes.

4.7. Alternative solutions

As shown in our previous works on real network situations, it happens that feasible solutions cannot be found by the above-described algorithm. In these cases there are two possible approaches: (i) saying that there is no solution and the LSPs cannot be reconfigured to their new, improved paths, or (ii) trying to reconfigure LSPs while allowing some interruption/degradation of traffic during the rerouting process. Here, the latter case is followed, i.e., some capacity violations are allowed in the reroute sequence calculation. However, these violations appear only in the calculations, and they are eliminated before performing the rerouting action. Several approaches are possible. Although these approaches are not applied in this paper, two examples are proposed:

- The *interrupting* approach allows some LSPs to be interrupted during the rerouting process, as follows. First, the reroute sequence is calculated while allowing some bandwidth thresholds to be exceeded. Then some LSPs are selected so that the summed bandwidth values on their old links cover all the bandwidth excess. After these calculations the selected LSPs are deallocated temporarily. The remaining LSPs are then rerouted to their new paths in the calculated sequence. Finally, the previously temporarily deallocated LSPs are established on their new paths. Note that in this case the traffic through the temporarily deallocated LSPs is interrupted during the rerouting process—this is the price of not violating any bandwidth threshold—therefore the number of such LSPs should be kept at a minimum.
- The *shrinking* approach decreases the reserved bandwidth of the LSPs during the rerouting process, which results in temporary service degradation but in fortunate cases this amount of degradation is so small that the network users do not perceive it. The first step is common with the above-mentioned approach: the reroute sequence is calculated while enabling capacity violation. Then the bandwidth reservations of LSPs are decreased so that all violations would be eliminated. In the simplest case the bandwidth values are decreased uniformly: the original values are multiplied by $1/(1+x)$, where x is the maximal violation (defined and investigated in Section 5.2). At the final step, after the rerouting process the bandwidth values are restored.

Note that by *feasible solution* we mean a solution that does not have capacity violation. The rerouting in the calculated sequence can thus be

performed without any service interruption or degradation. However, non-feasible solutions having some capacity violations can also be possible practical solutions, because in the above-described ways the rerouting can be performed in the calculated sequence without real capacity violations.

5. Numerical results

In order to investigate the algorithms, simulations were performed on a large number of problem instances. First, the problem instance generation that guarantees a feasible solution is described. We then define the metrics that are the base of the comparison of the algorithms. Finally, the numerical results of the simulation scenarios are shown.

5.1. Problem instance generation

To evaluate the performance of the presented algorithms, a large number of test problem instances are needed. An instance consists of the graph topology including the edge capacities, and an initial and a final path set (P and Q, respectively). In our previous work [15,16] investigations were performed on real network topologies with realistic input path sets but it was unknown whether the problem had a feasible solution (i.e., no capacity violation). For this reason, artificial examples were generated in the current test scenarios, which provably have feasible solutions, enabling reliable tests from the theoretical point of view.

The graphs were generated by a random graph generator described in detail in [14] having three parameters: the number of nodes n, the average nodal degree g, and the ratio of LERs l that represents the ratio of the number of nodes that can be sources and destinations of paths to the total number of nodes. LSPs were then generated (using shortest paths) between each pair of LER nodes where the integer transmission capacity value of an LSP was chosen randomly from a predefined interval $[1, b_{\max}]$ with one parameter: $b_{\max}$. Initially, the edge capacities of the graph were set exactly to the traffic traversing the edges: $c(e) := \sum_{i:e\in P_i} b_i$. At this point, the graph topology and a *shortest* path set were constructed and these paths served as final paths Q representing the optimized LSPs.

The last step in the problem instance generation was rerouting some LSPs whose number was controlled by parameter a. The following cycle was repeated a times, in which one particular LSP was rerouted. A predefined number of non-rerouted LSPs were selected randomly (in our investigations this number is set to 10). Then for each selected LSP a new path was sought that was distinct from its old path (at least in one edge) with the same source, destination, and capacity, so that the summed value of capacity violation was as low as possible. The LSP rerouted was that for which this value was the smallest. The value was equal to zero in fortunate cases and otherwise the capacities of the violated edges were increased in the slightest degree to eliminate the violations. After this cycle, for each unchanged LSP l_i path P_i was the same as Q_i, while for each other LSPs l_i the changed paths composed P_i. To sum up, this test example generation method provided us very "tight" examples—the capacity was increased in the slightest possible degree—for which a problem class was defined by $(n, g, l, b_{\max}, a)$. On the other hand, this generation also supplied a feasible reroute sequence that was the reverse sequence of the selected LSPs in the above cycle.

5.2. Investigations

In the first test scenario 1000 test instances were generated belonging to different problem classes. The parameters were taken randomly from the following intervals, according to the properties of real-world backbone networks: $n \in [10, 50]$, $g \in [3, 6]$, $l \in [0.5, 1.0]$, $b_{\max} \in [1, 1000]$, and $a \in [10, 200]$. We tried to solve the examples by all algorithms detailed in Section 4. For the comparison of the different methods the same four metrics were used as in [16]:

- *success probability*: the ratio of the number of cases without capacity violations to the total number of examined test instances,

- *maximal violation*: the greatest edge violation in percentage where the edge violation is defined by the maximal capacity excess during the rerouting action compared to the total capacity (corresponding to the maximal reservable bandwidth) of the edge,
- *edge violation*: the ratio of the number of violated edges to the total number of edges (m), and
- *capacity violation*: the ratio of the total capacity violation to the total capacity ($\sum_{e\in E} c(e)$) in the network.

Note that in the following all values of these metrics are given as percentages.

Consider Table 1, in which the results of the base algorithms (without enhancements) for each metric are shown. It is surprising that the success probability of RO is nearly zero, i.e., it could not succeed in solving any of the problem instances. On the other hand, RS—which is a modified version of RO—solved more than the half of the instances. It also turned out that MR is significantly better than the other greedy approaches in terms of any metric. This result is not surprising because it looks forward one rerouting step. Methods RS, MV, and MF gave nearly the same result but surprisingly RS was the best in terms of success probability and edge violation. MV was the best among them in terms of maximal and capacity violation, and MF was the worst at all. However, it is important to note that these results depend on the instance generation to a great extent and in another context the algorithms might perform differently.

Table 1
Results of the base algorithms

Algorithm	Success probability	Maximal violation	Edge violation	Capacity violation
RO	0.1	38.29	27.23	1.7224
RS	55.0	3.95	1.12	0.0419
MV	51.6	1.72	1.34	0.0239
MF	50.4	5.93	1.53	0.0572
MR	81.8	1.50	0.37	0.0044

Table 2
Success probability results at different enhancements

Algorithm	Enhancement				
	1	2	3	1 + 2	2 + 3
RS	55.0	22.2	76.4	59.0	77.3
MV	51.6	19.7	63.4	53.3	63.5
MF	50.4	26.2	71.1	55.0	71.4
MR	81.8	75.7	92.1	88.3	92.6

In the second test session the efficiencies of the different enhancements of the greedy methods were investigated. Table 2 shows the success probabilities of the different enhanced heuristic algorithms. Some combinations of the enhancements are also included, which means that both enhancements were applied to the problem instances. However, it was unnecessary to combine the first and third enhancements because the first one cannot solve the problem if the third one cannot solve it, due to the fact that the third enhancement is an extension of the first one. The second enhancement gave worse results than the others because for these "tight" examples it was much better to build up the reroute sequence by solving the easier problem out of the normal and reverse ones. The third enhancement improved the probability of success significantly, and there were only a few test instances that could be solved by the second enhancement but not by the third one (compare columns '3' and '2 + 3').

In the following investigations the relation between the complexity of the problem and the different parameters of the problem classes was examined. It was assumed that the networks were given, thus n, g, and l were fixed in a real situation.

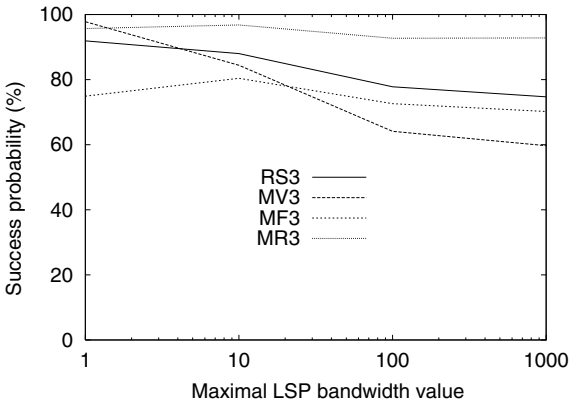

Fig. 1. Success probability results for different $b_{\max}$ values.

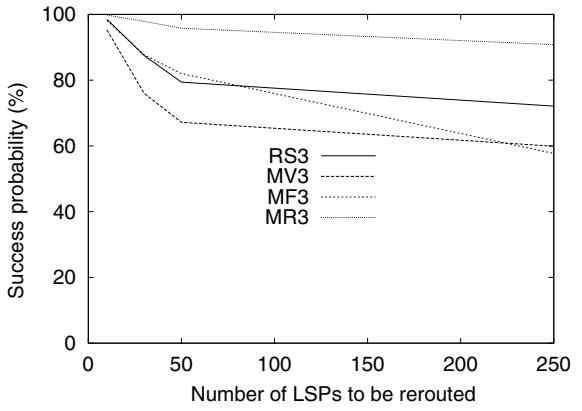

Fig. 2. Success probability results for different a values.

Therefore, the problem at different b_{max} and a values was investigated, while the other parameters were taken from the above-mentioned intervals. First, the parameter b_{max} was examined. The same input was generated as in the first test scenario except for the maximal bandwidth value of the LSPs (b_{max}), which was fixed to 1, 10, 100, and 1000, respectively. In Fig. 1 one can see that b_{max} influenced the results to a great extent, and the uniform 1 bandwidth value was not the easiest case for all methods but their success probability curves had a maximum value. However, we can say that over a particular b_{max} value the problem is harder to solve as the maximal bandwidth values were increased.

The relation between the number of LSPs to be rerouted a and the complexity of the problem can be seen clearly in Fig. 2. The success probability was in inverse proportion to a as expected. It also turned out that if a was increased, the probability of finding a feasible reroute sequence converged to a particular value that depended on the algorithm used.

6. Conclusion

This paper has addressed the investigation of the RSP problem from theoretical as well as practical points of view. Several bounds have been introduced that guarantee the existence of a feasible solution for the RSP problem. However, these bounds were proven to be so weak that we may say that the discrepancy theory based approximation algorithms have no practical relevance. Greedy heuristic algorithms have been investigated by empirical analysis on artificially generated "tight" examples that have provably feasible solutions. Every heuristic found feasible solutions in more than half of the cases, moreover the success probability of the third enhancement of MR was over 90%. Consequently, it is probable that some realistic problem instances in our previous works have no feasible solutions, and two alternative solutions—the interrupting and shrinking approaches—were introduced for these cases.

In summary, we propose to use the third enhancement of algorithm MR in practice. Furthermore, when the problem cannot be solved without violations, we suggest to recalculate the sequence by the MV algorithm and allow temporary interruption or degradation of some LSPs during the rerouting process using one of the presented alternative solutions.

Our future research aims at improving the success probability of RSP by using temporary and split paths, combining the path optimization with the RSP procedure, finally trying other general meta-heuristics (e.g., tabu search, simulated annealing).

Acknowledgements

The authors would like to acknowledge the comments and review of Zoltán Király, Gábor Magyar and Áron Szentesi, and thank all the colleagues in the Ericsson Traffic Lab for their useful advice.

Appendix A. Theoretical results

In this section our theoretical results on RSP are presented. The connection to discrepancy theory is discussed and criteria for the existence of feasible solutions are presented. Finally, an interesting result is shown concerning the applicability of random reroute sequence.

Close relatives of the RSP combinatorial problem have been studied in the mathematical literature. In discrepancy theory and in the theory of scheduling and sequencing there are also a lot of related questions but we have to emphasize our special interest in the algorithmic results and approaches. Now, three problems are presented to explore some of the related ones. Let s be a norm on $\mathbb{R}^m$.

The *s-discrepancy* of a vector-set $Z = \{z_1, z_2, \ldots, z_k\} \subset \mathbb{R}^m$ is defined to be

$$\mathrm{disc}_s(Z) = \min_{\varepsilon \in \{-1,+1\}^k} \left\| \sum_{i=1}^{k} \varepsilon_i z_i \right\|_s, \tag{A.1}$$

i.e., the problem is to divide the vector-set into two parts as equally as possible.

The notion of *compact vector summation* (CVS) is very similar to the vector formulation of RSP. In CVS the task is to compute an order of the elements of a zero-sum vector-set, so that all partial sums are small in the sum of this order. If $Z = \{z_1, z_2, \ldots, z_k\} \subset \mathbb{R}^m$, $\sum_{i=1}^{k} z_i = 0$ (called zero-sum vector-set) then define

$$\varphi_s(Z) = \min_{\pi} \max_{1 \leqslant t \leqslant k} \left\| \sum_{i=1}^{t} z_{\pi_i} \right\|_s, \tag{A.2}$$

where the minimum is taken over all permutations π of $1, 2, \ldots, k$. The function

$$\varphi_s(k) = \sup \left\{ \varphi_s(Z) \colon Z = \{z_1, z_2, \ldots z_k\}, \; \|z_i\|_s \leqslant 1, \; \sum_{i=1}^{k} z_i = 0, \; k \in \mathbb{N} \right\}$$

is known as the Steinitz-function [5,6,10,11,18–20].

Finally, the *dynamic s-discrepancy* of $Z = \{z_1, z_2, \ldots, z_k\} \subset \mathbb{R}^m$ is defined by

$$\mu_s(Z) = \min_{\varepsilon \in \{-1,+1\}^k} \max_{1 \leqslant t \leqslant k} \left\| \sum_{i=1}^{t} \varepsilon_i z_i \right\|_s. \tag{A.3}$$

The combinatorial optimization problems of computing the above functions (A.1)–(A.3) are NP-hard for several norms (e.g., for $s = \ell_\infty$ concerning the maximal violation that is defined in Section 5.2). The main difference between RSP and CVS using an ℓ_∞-like norm is that in the RSP we are interested only in the upper bound of the summing trajectory coordinate functions. Since the symmetrically bounded CVS problem is easier to examine, this problem is investigated instead of the one-side bounded RSP problem. To see the connection between the above presented problems and functions consider the following proposition.

Proposition A.1. *Let $Z = \{z_1, z_2, \ldots, z_k\} \subset \mathbb{R}^m$ with $\sum_{i=1}^{k} z_i = 0$. Then*

$$\varphi_s(Z) \leqslant \max_{\pi} \mu_s(\{z_{\pi_1}, z_{\pi_2}, \ldots, z_{\pi_k}\}), \tag{A.4}$$

where the maximum in the right-hand side is taken over all permutations π of $1, 2, \ldots, k$.

The inequality (A.4) is known as Chobanyan's transference lemma [8]. This is the most powerfully applicable tool to the RSP. Examining these discrepancy-related questions, our results are transformed for the RSP problem. Recall the assumption $L_0 = L_k = L$.

Although the efficiency of the approximation algorithms is usually measured in approximation factor, here an additive approach is more appropriate. Applying Grinberg's and Sevastyanov's CVS theorem [6,10,12], one can derive the following.

Theorem A.2. *Given a capacity function satisfying $c(e) \geqslant L(e) + m \cdot b_{\max}$ for all $e \in E$, a feasible reroute sequence can be computed in deterministic polynomial time by Grinberg's and Sevastyanov's algorithm* [10].

As it can be seen, this bound depends only on the number of edges and on the maximal transmission capacity. The disadvantage of this method is that it works well only when the capacity reservation $L(e)$ is significantly higher than the maximal transmission capacity multiplied by the number of edges, which is rather atypical in practice. This is why we look for other approaches. Chobanyan's lemma established the relation between the CVS and the dynamic discrepancy problem, and implied Proposition A.1. Based on this, it looks to be useful to apply Spencer's cosine

hyperbolic algorithm [19] to give an approximate solution to the dynamic discrepancy problem, inducing the following approximate RSP solution.

Theorem A.3. *Provided that the capacity function satisfies*

$$c(e) \geqslant L(e) + \sqrt{2k\ln(2m)} \cdot b_{\max} + \varepsilon L(e) \quad \forall e \in E, \tag{A.5}$$

a feasible reroute sequence exists, and it can be computed in $O(km\log(\varepsilon^{-1}))$ *time, where* ln *is the natural logarithm.*

Considering the practical properties of real networks, it can be supposed that $m \geqslant n$. We further assume that the number of LSPs given between each node pair is upper bounded by a constant C. Then $k \leqslant C\binom{n}{2} \leqslant C(m^2/2)$. Using only these bounds, apart from the $\sqrt{\ln(2m)}$ and constant factors, the bound is the same as in Theorem A.2. If $m \gg n$ or $k \ll C\binom{n}{2}$, the last result is better.

The other theoretical results are obtained by the simple application of the *probabilistic method*. Furthermore, due to a *martingale* based *method of pessimistic estimators* [17], the probabilistic argument gives rise to a deterministic algorithm.

Theorem A.4. *If the capacity function satisfies*

$$c(e) \geqslant L(e) + \sqrt{2\sum_{i=1}^{k} v_i(e)^2 \ln(2m)} + \varepsilon L(e) \qquad \forall e \in E, \tag{A.6}$$

a feasible reroute sequence exists, and it can be computed in $O(km\log(\varepsilon^{-1}))$ *time.*

The above result is particularly interesting for uniform and unit bandwidth values. We therefore set $b_i = 1$, $1 \leqslant i \leqslant k$. Denoting by $T(e)$ the number of LSPs for which exactly one of their paths P_i and Q_i contains the edge e (in other words the number of LSPs to be rerouted from/to edge e), $T(e) \leqslant L_0(e) + L_k(e) = 2L(e)$. Thus, the capacity function

$$\begin{aligned} c(e) &\geqslant L(e) + 2\sqrt{L(e)\ln(2m)} \\ &= L(e)\left(1 + 2\sqrt{\frac{\ln(2m)}{L(e)}}\right) \qquad \forall e \in E, \end{aligned} \tag{A.7}$$

satisfies (A.6). Consider now a fixed edge e. We emphasize that the needed relative additional capacity for the rerouting action tends to zero when $L(e)$ tends to infinity if m is fixed. Moreover, if $L(e) \gg \ln n$, then $\ln(2m)/L(e) \leqslant 2\ln n/L(e)$, which tends to zero as well. In other words, in this case, the RSP with unit bandwidth values admits an asymptotically optimal algorithm.

On the other hand, to mention a negative result: path-systems can be derived from *Hadamard matrices* [8] for which $K \leqslant L(e) \leqslant 2K$, and $\max_{1 \leqslant t \leqslant k}\{L(e) + \sum_{i=1}^{t} v_{\pi_i}(e)\} \geqslant L(e) + C\sqrt{K}$ for every permutation π, for some constant C.

The above theorem can be improved trivially by the Lovász local lemma [9] but we do not know any algorithm. Suppose that the length of every path is at most D. Then the matrix $M = (v_1, v_2, \ldots, v_k)$ (composed by the vectors v_i as its columns) has at most $2D$ non-zero entries in each column. Thus, the following theorem is obtained, which heavily exploits the row-, and the column-sparsity. It is still a challenge to derandomize it, and to see whether the classical derandomizations work or not [1,7].

Theorem A.5. *If expression*

$$c(e) \geqslant L(e) + \sqrt{2\sum_{i=1}^{k} v_i(e)^2 \ln(16DT(e))} \quad \forall e \in E, \tag{A.8}$$

holds, then there is a feasible solution for the RSP.

In real networks, LSPs are usually short, i.e., traverse as few routers as possible. This approach may therefore be of practical interest and is worth exploring further.

Greedy heuristic algorithms were also developed. They build up the reroute sequence by choosing an LSP in each step while minimizing different weight functions. The study of random permutation shows that the application of these heuristics is not illegitimate and we also emphasize that the exponential weight function version of these heuristic algorithms can be seen as the derandomization of the random permutation. The

following theorem explains this statement more precisely.

Theorem A.6. *If expression*

$$c(e) \geqslant L(e) + \sqrt{2T(e)\ln\left(\frac{T(e)m}{1-h}\right)} \cdot b_{\max}(e) \quad \forall e \in E, \tag{A.9}$$

holds, the random reroute sequence does not violate the capacities with greater probability than h. Especially, if $c(e) \geqslant L(e) + \sqrt{2T(e)\ln(T(e)m)} \cdot b_{\max}(e)$, $\forall e \in E$, *there exists a solution for the RSP problem (where* $b_{\max}(e)$ *denotes the maximal LSP bandwidth on edge e).*

References

[1] N. Alon, A parallel algorithmic version of the local lemma, Random Structures & Algorithms 2 (1991) 367–378.

[2] J. Ash, Y. Lee, P. Ashwood-Smith, B. Jamoussi, D. Fedyk, D. Skalecki, L. Li, LSP modification using CR-LDP, Internet Engineering Task Force, Request For Comments (Proposed Standard) 3214, January 2002.

[3] D. Awduche, MPLS and traffic engineering in IP networks, IEEE Communications Magazine 37 (12) (1999) 42–47.

[4] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, J. McManus, Requirements for traffic engineering over MPLS, Internet Engineering Task Force, Request For Comments (Proposed Standard) 2702, September 1999.

[5] I. Bárány, V.S. Grinberg, On some combinatorial questions in finite-dimensional spaces, Linear Algebra and its Applications 41 (1981) 1–9.

[6] I. Bárány, T. Fiala, Többgépes ütemezési problémák közel optimális megoldása, Szigma 15 (3) (1982) 177–191 (in Hungarian).

[7] J. Beck, An algorithmic approach to the Lovász local lemma, Random Structures & Algorithms 2 (1991) 343–365.

[8] J. Beck, V.T. Sós, Discrepancy theory, in: R. Graham, M. Grötsel, L. Lovász (Eds.), Handbook of Combinatorics, Elsevier, Amsterdam, 1995, pp. 1405–1446.

[9] P. Erdős, L. Lovász, Problems and results on 3-chromatic hypergraphs and some related questions, in: A. Hajnal, et al. (Eds.), Infinite and Finite Sets, Colloq. Math. Soc. J. Bolyai, vol. 11, North-Holland, Amsterdam, 1975, pp. 609–627.

[10] V.S. Grinberg, S.V. Sevastyanov, O velicine konstanty Steinica, Funkcionalnij Analiz i Prilozen 14/2 (1980) 56–57 (in Russian); [Functional Analysis and its Applications 14 (1980) 125–126], MR 81h:52008.

[11] D.S. Hochbaum (Ed.), Approximation Algorithms for NP-hard Problems, PWS Publishing Company, Boston, MA, 1997 (Chapter 1).

[12] W. Hoeffding, Probability inequalities for sums of bounded random variables, American Statistical Association Journal 58 (1963) 13–30.

[13] B. Jamoussi (Ed.), Constraint-based LSP setup using LDP, Internet Engineering Task Force, Request For Comments (Proposed Standard) 3212, January 2002.

[14] B.G. Józsa, Z. Király, G. Magyar, Á Szentesi, An efficient algorithm for global path optimization in MPLS networks, Optimization and Engineering 2 (3) (2001) 321–347.

[15] B.G. Józsa, G. Magyar, Reroute sequence planning for label switched paths in multiprotocol label switching networks, in: IEEE Symposium on Computers and Communications, Hammamet, Tunisia, 2001.

[16] B.G. Józsa, Reroute sequence planning for protected traffic flows in GMPLS networks, in: IEEE International Conference on Communications, New York, 2002.

[17] P. Raghavan, Probabilistic construction of deterministic algorithms: approximating packing integer programs, Journal of Computer and System Sciences 37 (1988) 130–143.

[18] S.V. Sevastyanov, On some geometric methods in scheduling theory: a survey, Discrete Applied Mathematics 55 (1994) 59–82.

[19] J. Spencer, Balancing games, Journal of Combinatorial Theory Series B 23 (1977) 68–74.

[20] J. Spencer, Balancing vectors in the max norm, Combinatorica 6 (1986) 55–65.

[21] X. Xiao, A. Hannan, B. Bailey, L.M. Ni, Traffic engineering with MPLS in the internet, IEEE Network 14 (2) (2000) 28–33.

Balázs Gábor Józsa received his M.Sc. degree in Computer Science from Budapest University of Technology and Economics (Hungary) in 2000. He is currently working on his Ph.D. at the High Speed Networks Laboratory (HSNLab) of the same university. Meanwhile, he also works as a research fellow at Ericsson Traffic Analysis and Network Performance Laboratory in Budapest, Hungary. His main interests are traffic engineering, routing and performance optimization problems in telecommunication networks.

Márton Makai got his M.Sc. degree in Applied Mathematics at the Eötvös University (Budapest, Hungary) in 2001. He is now a Ph.D. student at the Operations Research Department at the same university. He also works as a research fellow at Communication Networks Laboratory (Budapest, Hungary). His research subjects are combinatorial optimization, approximation algorithms and their applications in telecommunication networks such as routing, economic resource allocation and network design.