A New Bandwidth Guaranteed Routing Approach for Online Calculation of LSPs for MPLS Traffic Engineering

Karl Hendling, Brikena Statovci-Halimi, Gerald Franzl, and Artan Halimi

Vienna University of Technology, Institute of Communication Networks,
Favoritenstrasse 9/388, A-1040 Vienna, Austria
{karl.hendling, brikena.statovci, gerald.franzl,
artan.halimi}@tuwien.ac.at
http://www.ikn.tuwien.ac.at

Abstract. This paper presents a fast on-line routing algorithm for dynamic routing of label switched paths (LSPs) with bandwidth guarantees in MPLS networks, which handles requests that arrive one at a time without exploiting a priori knowledge of the traffic characteristics. Trying to avoid exacting calculations for each on-demand LSP request (e.g., maximum flow computation), we introduce a new link weight function for path selection. The link weights are calculated as a function of residual network and link capacity, hence we call the approach *Residual Network and Link Capacity* (RNLC) routing algorithm.
In terms of computer simulations we compare the performance of this new routing algorithm with four other on-line routing algorithms in two different network scenarios. Simulation results exhibit better performance of RNLC even if compared to more complex algorithms. We highlight that the new algorithm is fast and scalable due to its considerably low complexity.

1 Introduction

Traffic engineering is one of the main reasons for implementing multiprotocol label switching (MPLS) in IP backbone networks. This capability of MPLS is based on the fact that it efficiently enables explicitly routed paths, called label switched paths (LSPs), to be created between ingress and egress nodes. As a result, traffic flows can be controlled and engineered through the network. For an explicit LSP the route is determined at the ingress node. Once an explicit route is determined, a signaling protocol such as CR-LDP or RSVP-TE [1] is used to establish the LSP to the egress node.

The main goal of Internet traffic engineering is to efficiently optimize the performance of operational networks [1,2,3,4] in order to avoid the well-known shortcomings of the typical destination-based IP routing. Traffic engineering attempts to reduce or even avoid congestion hot spots and to improve the resource utilization across the backbone IP network. This may primarily be done by evenly distributing the incoming traffic over the available links in order to obtain balanced traffic scenarios. One core concept of traffic engineering, which is actually often identified as traffic engineering itself, is route optimization. Especially in networks with rather unbalanced traffic distributions, it can be applied to enhance the overall network quality. Another advantage is the support of specific quality of service (QoS) levels agreed with Service Level Agreements (SLAs) for services that need certain QoS requirements (e.g., a specific packet loss ratio, delay/jitter).

Current routing algorithms are mostly based on shortest path schemes, thus leading to unbalanced load distribution inside the networks and mostly do not consider QoS requirements. An efficient path selection procedure should enable the selection of a *feasible path* while achieving efficient resource utilization. A *feasible path* is one that has sufficient residual resources to satisfy the QoS constraints of a connection [5]. While a feasible path can be selected by a shortest path algorithm, if constrained by one metric only, additional optimality constraints need to be imposed to achieve efficient resource utilization. Several path selection schemes have been proposed and evaluated in the literature including *widest-shortest path* [6], *shortest-widest path* [7,8,9], and *utilization-optimized* algorithms. The *minimum interference routing algorithm* (MIRA) introduced in [10,11] takes into consideration future demands for its routing decision. Another routing algorithm inspired by MIRA was proposed in [12], which we call WSC (Wang-Su-Chen). MIRA and WSC both are *non-greedy* on-line routing algorithms, independent of the actual traffic profile. They achieve a better resource utilization in the network, however introducing high computation time. This can be of great importance when considering backbone networks where the ingress nodes operate at high loads, and have limited computing power.

Trying to avoid the maximum flow [13,14] computation, which introduces additional computation time, we present a new link weight function, which combines the following three criteria: saving of residual link bandwidth, optimal usage of network capacity, and minimization of path lengths. We calculate the link weights as a function of residual network capacity, link capacity, and a constant. Therefore, we call the algorithm *Residual Network and Link Capacity* (RNLC) routing algorithm. Simulation results exhibit better performance than MIRA and WSC for the studied scenarios.

Furthermore, we study the relationship between the performance of a routing algorithm and the network scenario. We show that the network scenario has a great impact on the performance of an algorithm. For comparison purposes we evaluate four algorithms: *minimum-hop algorithm* (MHA), *shortest-widest path* (SWP) algorithm, *minimum interference routing algorithm* (MIRA), and *Wang-Su-Chen* routing algoirhtm (WSC). The evaluated performance is compared with the RNLC routing algorithm in terms of different performance parameters and network scenarios.

The paper is organized as follows. Section 2 presents related work, and Section 3 introduces the problem definition and statement we consider in this work. Section 4 provides a detailed introduction to the new RNLC routing algorithm, and Section 5 gives a detailed complexity analysis of all studied algorithms. Section 6 points out the limitations of studied algorithms and, Section 7 introduces the studied simulation scenarios. Section 8 discusses the performance results and, Section 9 delivers some concluding remarks.

2 Related Work

The most commonly used algorithm for routing LSPs is the *minimum-hop algorithm* (MHA), where a feasible path with the least number of hops (links) connecting an ingress-egress pair is chosen. MHA gives highest priority to minimize resource occupation, however this can create bottlenecks for future flows, consequently leading to an under-utilized network.

Another routing proposal is the *shortest-widest path* (SWP) algorithm [7,8, 9], which considers two criteria. The first one is to pick the path(s) with the maximum reservable bandwidth amongst all feasible paths. If more than one such path exists, the one with the minimum-hop count is chosen. SWP gives highest priority to balance the network load across all links, however due to preferring detours less LSPs can be established.

In [10,11], the *minimum interference routing algorithm* (MIRA) has been proposed. Bandwidth guaranteed LSPs are explicitly routed such that minimum interference occurs between all possible connections (i.e., defined ingress-egress pairs) in order to be able to accommodate future LSP set-up requests, as well as LSP re-routing requests caused by link failures. Most importantly, this heuristic algorithm exploits information in terms of pre-defined ingress-egress communication pairs unlike other proposed *greedy* schemes. As mentioned above, MIRA performs its routing decision based on the interference level according to demands from other ingress-egress pairs. Interference is quantified by the notion of *critical links*. Critical links are links with the property that whenever an LSP is routed over these links the maximum flow values of one or more ingress-egress pairs decrease. The LSP should avoid the critical links as far as possible, which is achieved by generating a weighted graph where the weights assigned to the links are proportional to their criticality. The authors propose three different weighting schemes. For performance studies we apply the scheme where weight portions are inversely proportional to the maximum values, i.e., $\alpha_{sd} = 1/\theta_{sd}$, where θ_{sd} is the maximum flow value for the ingress-egress pair (s, d). This weighting implies that the critical arcs for the ingress-egress pairs with lower maximum flow values will be weighted heavier than the ones for which the maximum flow value is higher. MIRA gives its priority to minimize the criticality of LSPs and thereby minimizes the number of rejected future requests. The main complexity of MIRA per LSP request is given by $O((p-1)n^2\sqrt{m})$ ($p-1$ times maximum flow computation [13,14]) and $O(m^2)$ (critical links calculation), where p denotes the number of ingress-egress pairs, n denotes the number of nodes, and m denotes the number of links. However, MIRA focuses exclusively on the interference effect on single ingress-egress pairs, and is not able to estimate the bottleneck created on links that are critical for *clusters* of nodes.

Inspired by MIRA, Wang et al. [12] have proposed and studied another routing algorithm for bandwidth guaranteed LSPs. Similar to MIRA, WSC is an on-line algorithm, and is independent of traffic profiles. WSC is able to overcome some of MIRA's drawbacks (pointed out in [12,15,16]) by taking into account the overall bandwidth blocking effects of routing an LSP request. The main complexity of WSC per LSP request is given by $O((p-1)n^2\sqrt{m})$ ($p-1$ times maximum flow computation [13,14]).

Another approach called *Profile Based Routing* (PBR) is presented in [15, 16]. PBR is different from MIRA and WSC and assumes both known ingress-egress pairs and a traffic profile between them. A traffic profile is derived from measurements or service level agreements (SLAs) as a rough predictor for future traffic distribution. PBR uses the traffic profile in the pre-processing step (one multi-commodity flow computation), to determine certain bandwidth allocations on the links of the network. The on-line phase of the routing algorithm then routes LSPs using a shortest path like algorithm exploiting the additional information from the pre-processing phase, i.e., occupying resources according to the pre-allocated bandwidth.

3 Problem Definition and Statement

We model the network as a graph $G = (V, E)$, where V ($|V| = n$) denotes the set of nodes (routers), and E ($|E| = m$) denotes the set of links. A subset of nodes is assumed to be ingress-egress nodes, between which LSPs can be set-up. However, it is not necessary that there is a potential LSP between every ingress-egress pair. We assume that all ingress-egress pairs are known in advance and denoted by a set P ($|P| = p$). Each LSP set-up request arrives at an ingress node, which in turn determines an explicit bandwidth satisfying route. To determine the route, each ingress node needs to know the entire topology of the network and the current link states. The residual link capacity of link l is denoted as $R(l)$ and the residual network capacity as $N_c = \sum_{\forall\, l \in E} R(l)$, i.e., the sum over all $R(l)$ (with a complexity $O(m)$). Therefore, we assume that the entire topology is either known administratively or that a link state routing protocol is operational, and that its link state database is accessible. The routing protocol database keeps track of all residual link capacities, and we assume that all initial link capacities are known and thereby the initial network capacity also. Failures of LSPs due to link faults are detected from signaling protocol (e.g., CR-LDP or RSVP-TE) information by the edge nodes. The link state database is updated by the routing protocols, and edge nodes can then request a re-routing of the LSPs.

A request for an LSP set-up r_i is defined by a triple (s_i, d_i, b_i), where $(s_i, d_i) \in P$, s_i is the ingress node, d_i is the egress node, and b_i represents the amount of bandwidth required by the LSP. All QoS requirements for the flow have been folded into the bandwidth b_i. Furthermore, we assume that requests for LSPs arrive on-line, one at a time, and there is no knowledge of the characteristics of future demands. The objective is to find a path for LSP request r_i in the network from s_i to d_i along which each link has a residual capacity of at least b_i, otherwise the request r_i is rejected. In this work, we only focus on the establishment of bandwidth guaranteed paths.

4 Residual Network and Link Capacity Routing Algorithm

This section presents the new routing approach, based on *residual network and link capacity*, for short RNLC routing algorithm. RNLC provides a new link weight function which combines three criteria: saving of residual link bandwidth, optimal usage of network capacity, and minimization of path lengths.

The weight function is given by

$$w(l) = \frac{N_c}{R(l)} + C, \tag{1}$$

where N_c $(= \sum_{\forall\, l \in E} R(l))$ is the current residual network capacity and $R(l)$ is the current residual (i.e., unreserved) link capacity on link l at the arrival event of request r_i. The constant C determines the dynamic behavior of the generated link weights. Fig. 1 shows the dependence of this weight calculation scheme on $R(l)$ and N_c.

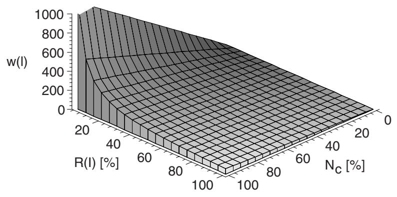

Fig. 1. Link weights as a function of residual network and link capacity.

If the residual link capacity approaches zero, the weight approaches infinite, thus eliminating all links with insufficient residual capacity. The dependence of the weight function on residual network capacity needs some detailed consideration: In case of low network load (i.e., high residual network capacity), links with less residual capacity (higher load) are assigned considerably higher weights than less loaded links. Consequently, paths over lightly loaded links are preferred and heavily loaded links are avoided. This keeps as many links as possible available for future requests, i.e., intends to avoid congestion. In case of high network load (i.e., low residual network capacity), all link weights are approximately the same as long as there is sufficient residual capacity on the links. Therefore, the minimum-hop path is preferred, and routing is performed subject to minimum resource occupation, leaving a maximum of resources available for additional requests. Reflecting, if the same weighting as the one used for low loads is applied, detours would be preferred, which save some residual link capacity (bandwidth) on individual links, but due to their higher number of hops, these paths would occupy more network capacity, and consequently reduce the available resources for future requests.

With the additive constant C the dynamic of the scheme can be controlled. If C is chosen very big, the scheme behaves like minimum-hop routing, still having the advantage of eliminating links with vanishing residual capacity. The smaller C is set, the stronger the link weights reflect the distribution of the load on the links, i.e., smaller C increases the variance of the link weights. This constant C should be chosen according to topology, meshing degree and traffic distribution — an evaluation on C is presented in Section 8.

This weight calculation scheme with its complexity $O(m)$ is fast, and routing can be done according to the Dijkstra or Bellman-Ford algorithm, both well known and common. The only drawback compared to shortest path and widest bottleneck bandwidth scheme is that the link weights can not be calculated autonomously by adjacent nodes, because the residual network capacity N_c is required. As previously mentioned, N_c is calculated as the sum over all $R(l)$ and therefore a network-wide distribution (or view) of $R(l)$ is required. Compared to MIRA and WSC this scheme is by far less complex, as no maximum flow computation is required, and both rely on accurate knowledge of link states as well.

Studying the fairness of the weighting scheme, we consider the mean link weight $\overline{w}(l)$ and find it being constant, i.e., equal to the number of links plus C $(m + C)$,

$$\overline{w}(l) = \frac{N_c}{\overline{R}(l)} + C = m + C, \qquad (2)$$

where $\overline{R}(l)$ $(= \frac{N_c}{m})$ is the average residual link capacity. This overall stability yields fairness, and the dynamic can be controlled efficiently via the constant C. A similar scheme might be defined by using percentages instead of absolute figures for residual capacities. In that case C needs to be chosen accordingly smaller, e.g., $C = 1$ would quite severely leverage the dynamic. To achieve the same behavior as with the above shown scheme, C needs to be scaled by $\frac{1}{m}$.

High level view of RNLC routing algorithm:

Input: Graph $G = (V, E)$, the set of residual link capacities on all links, and the request r_i (s_i, d_i, b_i).
Output: A path from s to d, such that for each link l along this path $R(l) \geq b$.

1. Compute the weight $w(l)$ for each link l in the graph $G = (V, E)$ according to equation 1.
2. Eliminate all links l, which have residual bandwidth less than b leading to a reduced graph.
3. Compute shortest path in the reduced graph by means of Dijkstra's algorithm using the corresponding $w(l)$ on the links.
4. Route the demand b from s to d along this shortest path and update the residual link capacities.

5 Complexity Analysis

In this section we analyze the complexity of the studied routing algorithms. MHA needs no special requirements and the overall complexity is the shortest path selection (e.g., Dijkstra with $O(n^2)$ or Bellman-Ford with $O(n^3)$). Similar to MHA, SWP also needs no special requirements. The difference to MHA is that SWP applies a modified Dijkstra algorithm to select the shortest path with an overall complexity $O(n^2)$. RNLC provides a slightly higher complexity than MHA and SWP, additionally to the shortest path selection, $w(l)$ and N_c are required. Each of these values can be calculated with a complexity $O(m)$.

MIRA needs to perform $p - 1$ (number of competing ingress-egress pairs) maximum flow computations. Each of these maximum flow computations takes $O(n^2\sqrt{m})$ time. Further, MIRA needs to enumerate the links belonging to minimum cuts with a complexity $O(m^2)$. Afterwards the link weight $w(l)$ for all links is calculated with an overall complexity $O(m)$. Finally the shortest path selection with $O(n^2)$ is performed. WSC performs the same steps as MIRA with the exception that the critical link calculation is skipped.

A detail record is given in Tab. 1, which shows the overall complexity of each routing algorithm when performing the path selection.

Table 1. Complexity of each routing algorithm per LSP request.

routing algorithm	preparation phase	$w(l)$ calculation	shortest path selection Dijkstra
MHA	—	—	$O(n^2)$
SWP	—	—	$O(n^2)$
RNLC	$O(m)$	$O(m)$	$O(n^2)$
MIRA	$O((p-1)n^2\sqrt{m}) + O(m^2)$	$O(m)$	$O(n^2)$
WSC	$O((p-1)n^2\sqrt{m})$	$O(m)$	$O(n^2)$

6 Example Scenario: Limitations of Studied Algorithms

Fig. 2 shows a network scenario termed as *collector-distributor* with its properties. This scenario exhibits a case where MIRA and WSC do not perform as well as RNLC and SWP.

Let us suppose the on-line sequence of 8 LSP requests arrives in the order $(S_3, D_3, 1)$, $(S_1, D_1, 1)$, $(S_3, D_3, 1)$, $(S_4, D_4, 1)$, $(S_3, D_3, 1)$, $(S_2, D_2, 1)$, $(S_3, D_3, 1)$, and $(S_4, D_4, 1)$. MHA always prefers the link $(7, 8)$ when traffic from (S_3, D_3) arrives, thus causing that the LSP request (S_2, D_2) is rejected. According to MIRA and WSC, traffic from (S_3, D_3) always prefers the links $(2, 3)$ and $(3, 4)$, as the link $(3, 4)$ only interferes with (S_4, D_4) traffic, while the link $(7, 8)$ interferes with traffic (S_1, D_1) and (S_2, D_2). They route 3 of the 4 demands from (S_3, D_3) over the links $(2, 3)$ and $(3, 4)$ and 1 demand over the link $(7, 8)$. In this case, the last demand from (S_4, D_4) is rejected, while in the other part of the network, the link $(7, 8)$ is under-utilized. An optimal algorithm routes 2 of the 4 demands from (S_3, D_3) over the link $(7, 8)$ and 2 demands over the links $(2, 3)$ and $(3, 4)$. For this scenario, both RNLC and SWP choose the LSPs like an optimal routing algorithm, and therefore all 8 LSP requests for all ingress-egress pairs can be served successfully.

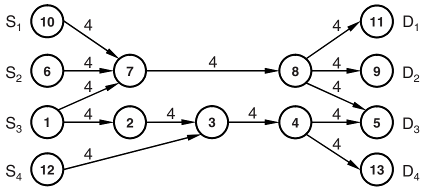

Fig. 2. The collector-distributor scenario.

7 Simulation Scenario

Without real network topologies and large amounts of traffic data, it is difficult to perform meaningful and conclusive experiments. Therefore, we follow the tradition set by other authors, and perform experiments on two handcrafted topologies depicted in Fig. 3 (taken from [16], here called the KL2+ scenario) and Fig. 4 (the more realistic ISP topology, widely used for studies on QoS routing). Fig 3 presents the network topology that has been used in [10,11] to propose MIRA, however, with two additional ingress-egress pairs (S_5, D_5) and (S_6, D_6). Due to these changes of the ingress-egress structure, the interferences change as well.

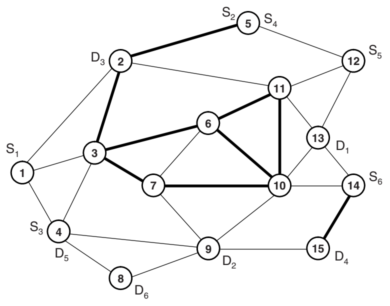

Fig. 3. The KL2+ network scenario.

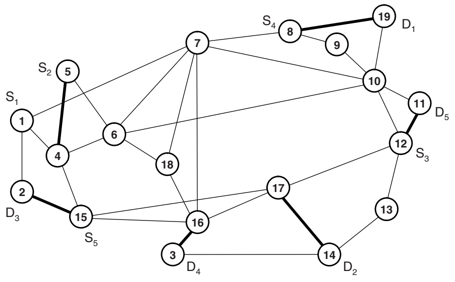

Fig. 4. The ISP network scenario.

The bandwidth of each light link and each bold link is 1,200 units and 4,800 units, respectively. These values are taken to model the capacity ratio of OC-12 and OC-48 links. Each link is bidirectional (i.e., acts like two unidirectional links of that capacity). The ingress-egress pair arrangements for LSP set-up requests are given in Fig. 3 and Fig. 4. All LSP set-up requests have been uniformly distributed among the ingress-egress pairs of the corresponding network scenario. Furthermore, LSP bandwidth demands are taken to be uniformly distributed between 1 and 4 units (only integer values are used).

8 Performance Studies

A study on the influence of the constant C is shown in Fig. 5. We can see that the achievable throughput in the KL2+ scenario reaches a maximum for $C \leq 1$. In contrast, for the ISP scenario, the influence of the constant C is marginal. The achievable throughput is almost constant within the shown scope, however, for $C = 1$ the throughput is negligibly higher. Generally, the higher C is chosen the more shorter paths become preferred, thus the more is the resource occupation minimized. The smaller C is selected, the more is load balancing preferred, thus congestion hot-spots better avoided. Based on this, we set $C = 1$ for both simulation scenarios.

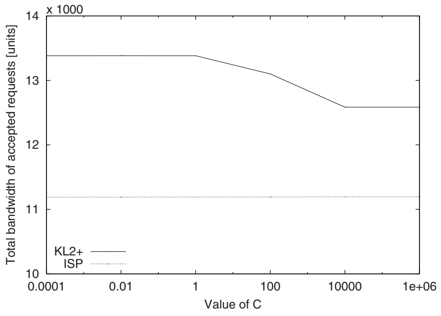

Fig. 5. The influence of C at the two network scenarios.

In the first set of experiments we load the two network scenarios with 8,000 LSP requests. We assume that all 8,000 LSPs are long-lived, i.e., once an LSP is routed it will not be terminated. We perform 20 trials, where for each trial newly randomly chosen long-lived LSPs are generated. Afterwards, we calculate the mean values for all evaluated parameters. Fig. 6 and Fig. 7 show the sum of the remaining maximum flow (allocatable bandwidth) of all ingress-egress pairs, after routing an LSP request with the studied algorithms and updating the link capacities. For each algorithm, the maximum flow (allocatable bandwidth) decreases with the number of accepted requests until a saturation point is reached at which no more requests can be accommodated.

From Fig. 6, we can see that the allocatable bandwidth for MIRA is a little bit higher than RNLC's within the range of $[1,910; 4,550]$ requests. The insert exhibits that at higher loads ($> 4,550$ requests) RNLC provides more allocatable

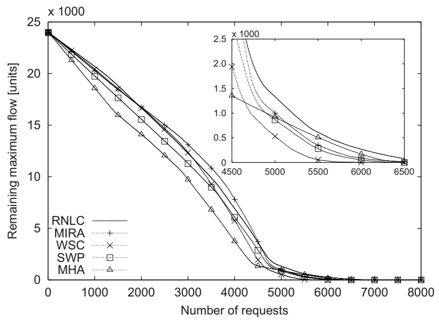

Fig. 6. Sum of maximum flow over all ingress-egress pairs in the KL2+ scenario.

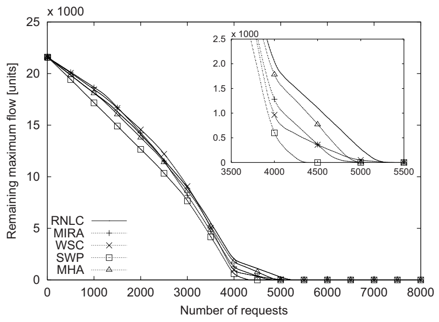

Fig. 7. Sum of maximum flow over all ingress-egress pairs in the ISP scenario.

bandwidth than any other studied algorithm, while MIRA drops behind MHA in a small interval. Fig. 7 exhibits a similar behavior as depicted in Fig. 6. At the beginning, MIRA and WSC provide more allocatable bandwidth than any other studied algorithm, but the decreasing rate (allocatable bandwidth) for MIRA and WSC is larger than RNLC's. Consequentially, RNLC provides more allocatable bandwidth at higher loads ($> 3,160$ requests) than any other studied algorithm.

Fig. 8 and Fig. 9 show the number of blocked requests (rejects) versus the total number of requests. RNLC shows in both figures (Fig. 8–9) the best performance, yielding the fewest rejects. Fig. 8 (the KL2+ scenario) exhibits that WSC causes more rejects than RNLC, MIRA and SWP. Fig. 9 (the ISP scenario) exhibits a better performance of MHA compared to the non-greedy algorithms MIRA and WSC.

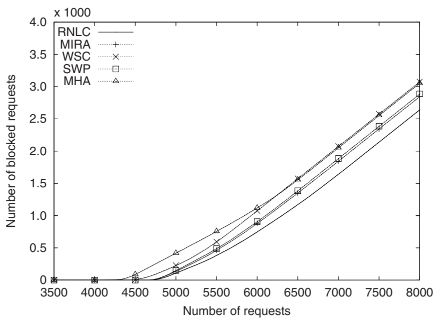

Fig. 8. Blocked requests in the KL2+ scenario.

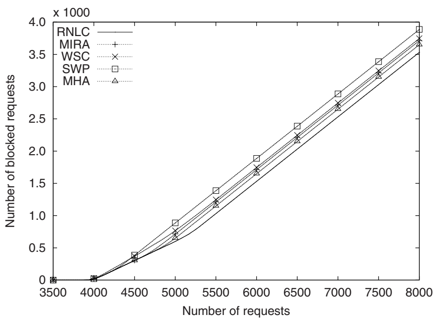

Fig. 9. Blocked requests in the ISP scenario.

Fig. 10 and Fig. 11 show the total bandwidth of accepted requests (throughput). For each studied algorithm, the bandwidth increases with the number of accepted requests until a saturation point is reached. In both figures, RNLC clearly shows the best performance in terms of maximum throughput. Due to the fact of the highest blocking, WSC exhibits the worst performance in Fig. 10 (the KL2+ scenario) above 6,240 LSP requests, thus yields the lowest maximum throughput.

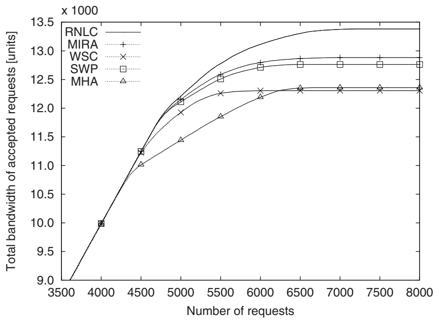

Fig. 10. Throughput of accepted requests in the KL2+ scenario.

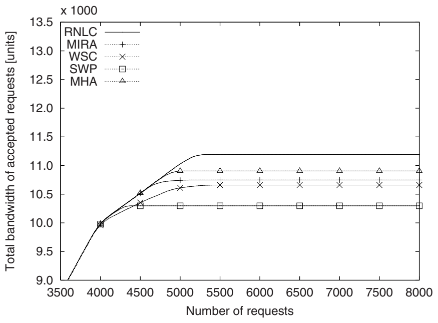

Fig. 11. Throughput of accepted requests in the ISP scenario.

In the second set of experiments, we determine the dynamic behavior of the routing algorithms. We use the same assumptions concerning the ingress-egress pair arrangements, uniform distribution of requests and bandwidth demands among all ingress-egress pairs. Each network scenario is first loaded with 4,000 randomly chosen long-lived LSPs and from this point on, all arriving LSP requests are assumed to have exponential holding time. The exponential holding time is chosen so that an average rate of 2,000 short-lived LSPs has to be accommodated additionally to the 4,000 long-lived LSPs. We run the experiment for 8,000 LSP requests (including the initial 4,000 LSP set-up requests), and perform 20 trials.

Fig. 12 and Fig. 13 show the number of blocked requests of each trial. As these two figures exhibit, RNLC causes smaller number of blocked requests than any other studied routing algorithm. This leads to improved performance and hence provides better overall network resource utilization. The differences in performance of MHA and SWP depicted in these figures show how scenario-dependent the performance of routing algorithms can be. Moreover, MIRA and WSC show instable performance (strong oscillation) while other algorithms exhibit more stable behavior with the same simulation scenarios.

9 Conclusion

In this paper, we have proposed a fast on-line routing algorithm for dynamic routing of bandwidth guaranteed LSPs. The proposed routing algorithm is different from the two commonly used *minimum-hop algorithm* (MHA) and *widest-shortest path* (WSP). This new algorithm avoids using residual capacity of congested links by means of routing traffic demands away from hot spots, even if

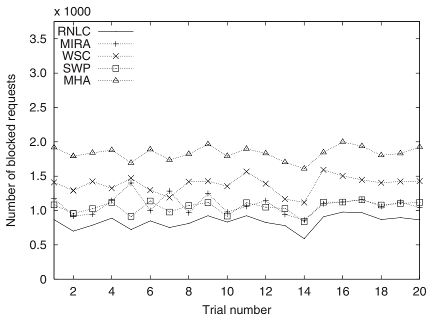

Fig. 12. Blocked requests during 20 independent trials in the KL2+ scenario.

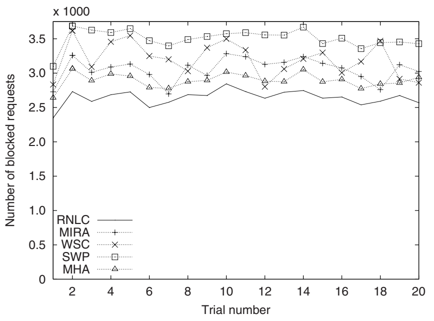

Fig. 13. Blocked requests during 20 independent trials in the ISP scenario.

the traffic then uses slightly longer paths. This increases the acceptance rate for future demands, and consequently reduces the rejection rate, which is especially important when re-routing due to a link failure is performed.

The *Residual Network and Link Capacity* (RNLC) routing algorithm is substantially faster due to much lower computational complexity than the recent proposals, *minimum interference routing algorithm* (MIRA) and the routing approach proposed by Wang-Su-Chen (WSC). This is a crucial point, because a high computation delay is a limitation for on-line routing algorithms. For the studied scenarios, RNLC shows better overall performance.

Moreover, we found that the chosen network scenario has a considerable influence on the performance of the routing algorithms. Especially, WSC shows an unfavorable performance when used with the KL2+ scenario, even though it overcomes some of MIRA's drawbacks by taking into account the overall bandwidth blocking effects of establishing an LSP request.

In practice, network operators may have to find a combination of off-line and on-line algorithms in order to operate their networks more efficiently. Off-line algorithms are used to re-optimize routing occasionally, while on-line algorithms (e.g., RNLC, MHA) allow for rapid response to new traffic demands or even new network conditions, for which off-line algorithms can be prohibitively slow. The scalability and fast response of the proposed routing algorithm make it applicable for large, even huge networks, if the link state broadcast is well implemented.

An aspect to extend our work is to alter C dynamically depending on the traffic demands (e.g., traffic type and size), another to consider off-line optimized $C(l)$.

References

1. Ghanwani, A., Jamoussi, B., Fedyk, D., Ashwood-Smith, P., Li, L., Feldman, N.: Traffic Engineering Standards in IP Networks Using MPLS. IEEE Communications Magazine **37** (1999) 49–53
2. Awduche, D.: MPLS and Traffic Engineering in IP Networks. IEEE Communications Magazine **37** (1999) 42–47

3. Wang, Z.: Internet QoS: Architectures and Mechanisms for Quality of Service. The Morgan Kaufmann Series in Networking. Morgan Kaufmann Publishers (2001)
4. Spraggs, S.: Traffic engineering. BT Technology Journal **18** (2000) 137–150
5. Chen, S., Nahrstedt, K.: An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions. IEEE Network, Special Issue on Transmission and Distribution of Digital Video **12** (1998) 64–79
6. Guérin, R.A., Orda, A., Williams, D.: QoS Routing Mechanisms and OSPF Extensions. Proceedings of IEEE Global Communications Conference 1997 (GLOBECOM'97), **3** (1997), 1903–1908
7. Wang, Z., Crowcroft, J.: Quality-of-Service Routing for Supporting Multimedia Applications. IEEE Journal on Selected Areas in Communications **14** (1996) 1228–1234
8. Ma, Q., Steenkiste, P.: On Path Selection for Traffic with Bandwidth Guarantees. Proceedings of IEEE International Conference on Network Protocols 1997 (ICNP'97) (1997) 191–202
9. Kamei, S., Kimura, T.: Evaluation of Routing Algorithms and Network Topologies for MPLS Traffic Engineering. Proceedings of IEEE Global Communications Conference 2001 (GLOBECOM'01) **1** (2001) 25–29
10. Kodialam, M., Lakshman, T.V.: Minimum Interference Routing with Applications to MPLS Traffic Engineering. Proceedings of IEEE Computer and Communications Societies 2000 (INFOCOM'00) **2** (2000) 884–893
11. Kar, K., Kodialam, M., Lakshman, T.V.: Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications. IEEE Journal on Selected Areas in Communications **18** (2000) 2566–2579
12. Wang, B., Su, X., Chen, C.L.P.: A New Bandwidth Guaranteed Routing Algorithm for MPLS Traffic Engineering. Proceedings of IEEE International Conference on Communications 2002 (ICC'02) **2** (2002) 1001–1005
13. Goldberg, A.V., Tarjan, R.E.: A New Approach to the Maximum-Flow Problem. Journal of the Association for Computing Machinery **35** (1988) 921–940
14. Ahuja, R.K., Magnanti, T.L., Orlin, J.B.: Networks Flows: Theory, Algorithms, and Applications. Prentice-Hall, Inc., Upper Saddle River, New Jersey 07458 (1993)
15. Suri, S., Waldvogel, M., Warkhede, P.R. In: Profile-Based Routing: A New Framework for MPLS Traffic Engineering. Volume 2156 of Lecture Notes in Computer Science. Springer Verlag, Berlin (2001) 138–157
16. Suri, S., Waldvogel, M., Bauer, D., Warkhede, P.R. In: Profile-Based Routing and Traffic Engineering. Volume 26 of Computer Communications. Elsevier Science B.V., Amsterdam (2003) 351–365