www.czasopisma.pan.pl PAN www.journals.pan.pl
INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2011, VOL. 57, NO. 2, PP. 169–175
Manuscript received April 10, 2011; revised May 2011. DOI: 10.2478/v10177-011-0024-9

Performance Evaluation of Preemption Algorithms in MPLS Networks

Sylwester Kaczmarek and Krzysztof Nowak

***Abstract*—Preemption is a traffic engineering technique in Multiprotocol Switching Networks that enables creation of high priority paths when there is not enough free bandwidth left on the route. Challenging part of any preemption method is to select the best set of paths for removal. Several heuristic methods are available but no wider comparison had been published before. In this paper, we discuss the dilemmas in implementing preemption methods and present the simulation study of well known existing algorithms. Based on the results, we provide recommendations for deployment of preemption for the two most common evaluation criteria: number of preemptions and preempted bandwidth.**

***Keywords*—MPLS, preemption, algorithms, traffic engineering.**

I. Introduction

MULTIPROTOCOL Label Switching (MPLS) networks are now widespread and became the de facto standard for core tele-communications networks. The added value that MPLS offers service providers is faster deployment of services like Layer 3 VPN and Layer 2 VPN. Thanks to integrated bandwidth management and fast recovery, the MPLS network is easier to manage and gives operator better control over used resources.

MPLS is already a mature technology, which beginnings date back to the early nineties. At that time the Internet network started to attract millions of users worldwide and the number of connected hosts has grown exponentially. That caused increase of IP routing tables in the core routers to the size which was not predicted before. The first generation routers were based mainly on software logic and could not cope much longer with predicted increase of routing tables due to limited switching capacity. Thus began searching for new solutions aimed at simplifying IP routing and that eventually led to definition of the MPLS architecture [1]. As we can see modern routers do manage with huge routing tables and the primary reason of introducing MPLS has become irrelevant. However, the introduction of MPLS revealed great potential of this technology and brought into the IP world a new term: traffic engineering.

MPLS has evolved from several technologies, including Cisco's proprietary "tag switching", where tagging of IP packets was introduced. There were other companies which contributed in MPLS development with their own experiences in IP switching using some local identifiers attached to IP packets. The concept was based on ATM switching and Frame Relay switching. Indeed, the first MPLS enabled routers were based mainly on ATM switching capable devices [2], [3].

The first MPLS standard has been published as MPLS Architecture in RFC 3031 [1] in 2001 as the result of several years of work of the Internet Engineering Task Force (IETF). This document defines the principles of the technology, in-cluding the functional architecture of the network, the role and requirements for the routers, principles of switching and paths building. The earlier published RFC 2702 [4] defined the requirements for traffic engineering in MPLS. The document does not present any practical method of traffic engineering. What it contains, however, is a formulation of problems in implementing traffic engineering in IP routing.

In RFC 2702, the term of the traffic trunk is introduced, which is implemented in MPLS as label switched path (LSP). The LSPs together with label-based routing are among the fundamental differences between IP and MPLS networks. In principle, the packet forwarding in MPLS is based on the label field rather than on the IP address. Before you can send data in MPLS network, a contiguous path must be created through the network. Only then the routers can properly deliver the packet. If no path exists then the packet must be dropped.

When an IP packet enters the MPLS network, it is classi-fied into one of the forward equivalence classes (FEC). The selection criteria are the destination IP address and optionally other fields, e.g. destination TCP port, Type of Service (ToS) or DiffServ (DS) field value. Having given the FEC number the packet is assigned a label and sent to the output port following the Next Hop Label Forwarding Entry (NHLFE). From that point on the attached label is used to switch the packet rather than IP address. The consecutive routers read the label and based on matching entry in the Incoming Label Map table change the label value and send the packet to the output port. After the packet reaches the last MPLS router its label is removed and the packet is sent to the destination as ordinary IP packet.

In MPLS you can reserve bandwidth for paths. Though it's not mandatory, it is required by many traffic engineering methods. When creating the path, the network administrator can declare the bandwidth to be reserved. The reservation process itself is usually performed by using the Resource Reservation Protocol (RSVP) which has been developed for Inte-grated Services (IntServ) architecture in IP networks and later adopted for MPLS as RSVP-TE [5]. The protocol ensures that after a successful completion of a request a contiguous path is created and the specified bandwidth is reserved for it on every router along the path.

This work has been supported partially by the Polish National Central for Research and Development under project PBZ-MNiSW-02/II/2007.

S. Kaczmarek is with Gdansk University of Technology, Faculty ETI (kasyl@eti.pg.gda.pl).

K. Nowak is with Nokia Siemens Networks (krzysztof.nowak@nsn.com).

The remainder of the paper is organized as follows. In the second section we explain the role of preemption in a path creation procedure and discuss the decisions which need to be made to implement preemption. In the next two sections we introduce the existing algorithms and present the methodology and conditions of simulations. In the fifth section we present the simulation results and evaluate them using the most common performance factors. Finally we present conclusions and directions for future works.

II. Preemption Methods

Paths in MPLS networks have, among other attributes, two priority values: the setup priority and the holding priority. Both of them are used during the creation of the path by one of the MPLS traffic engineering methods called preemption. The goal of preemption is to ensure that high priority paths will be admitted even if there is not enough bandwidth available due to reservation made previously for other paths. It simply causes removing some paths which have the holding priority lower than the new path's setup priority. We can describe the setup priority as the ability to preempt (remove) other paths and the holding priority as the ability to defend from being preempted.

The path priorities can range from zero (highest priority) to seven (lowest priority). For example, a path of the setup priority s=2 can preempt paths of the holding priority h=3 or more (up to 7). There is one important rule for the priorities. The setup priority cannot be stronger (or numerically lower) than the holding priority. Otherwise we could generate infinite chain of mutual preemptions, when path A would preempt path B, and in turn B would preempt A, and so on. Thus a path of s=3 must be assigned $h \leq 3$. On the opposite side, to prevent preemptions from happen you create a new path of h=0 and s=7. This ensures that the path will neither preempt any path nor will it ever be preempted by any other path, regardless of the priorities of the existing and future paths.

In practical implementations preemption is performed automatically and is closely connected to the routing functionality of the MPLS router. The procedure runs in several steps.

1) The route for the new path is determined using any available bandwidth-aware routing protocol, e.g. OSPF-TE. If the requested bandwidth is available on every link along the route, then preemption is not needed and the path is simply created. If the bandwidth is not available, then the procedure continues with the next step.
2) A subset of existing lower priority paths is selected to be preempted (removed). These candidate paths would release necessary bandwidth on every link where it is necessary. This can fail if no (more) paths exist which can be used to release enough bandwidth. If succeeded, the procedure continues with the act of preemption.
3) The actual preemption is performed. The candidates are removed, the bandwidth previously used is released and new path is created on the selected route.
4) If possible, the preempted paths are created again on alternative routes. This may potentially generate additional preemptions, if the paths are not of the weakest setup priority.

The described procedure seems simple, but it hides problems which need to be solved and decisions which need to be made. At least the first two steps deserve more detailed analysis.

The process of selecting the route can be done in at least two ways: 1) by trying to find any route without preemption and if it fails then to rerun the preemption-aware routing, or 2) by running preemption-aware routing always (without rerun). In the first case the preemption can be avoided if only the feasible route with enough bandwidth exists. In the second case the best possible route for high priority path is selected at the cost of more often preemptions.

The selection of candidates to preempt is the most complicated part of the procedure. However, at the first glance it may seem quite straightforward.

1) Select the links of the selected route, where free bandwidth is lower than the requested bandwidth. On these links the calculations are performed.
2) For every selected link find the paths of lower holding priority than the setup priority of the new path.
3) From preselected paths (potential candidates) choose the smallest set of candidates that provides enough resources after preemption. The sum of bandwidths of the candidates must be equal or greater than the amount of insufficient bandwidth on the links.

In cases when preemption is required on more than one link, the selected candidates from every link are included in the result set of candidates for preemption. However, this requires that the calculations on the consecutive links take into consideration the bandwidth collected on previous links. For example, if a candidate path P1 is selected on the link A, then on the link B, where P1 also exists, P1's bandwidth should be added to the free bandwidth before searching for new candidate. In this case it can happen that P1 only is enough to satisfy the preemption request on both links and no additional preemptions on link B is required. For those cases two different approaches to preemption can be used. The first one is the local preemption, which only uses information about the paths allocated on the link being currently under analysis. The second one is the global approach which uses topology and route information to make better selection of the candidates. In the latter case the algorithm may prefer the paths which exist on every link where preemption is needed and reduce the number of candidates in this way.

The big problem with preemption is how to choose the best set of candidates. In practice many different choices may be possible. As an example let us analyze a simple scenario. On a single link there is preemption needed to release the bandwidth of 20Mbit/s. The potential candidates are two paths of 10Mbit/s and one of 30Mbit/s. The question is what choice is better: to choose one big path offering more bandwidth than necessary or two smaller paths that give exactly the expected bandwidth? There is no perfect answer which fits every case. It depends on what count more for the network provider and how are the costs of preemptions calculated. If the cost of each preemption is high, then the minimizing the number of candidates will be the priority and the best choice would be the single preemption of 30Mbit path. Otherwise, if the

TABLE I
PARAMETERS OF THE TOPOLOGIES USED IN SIMULATIONS

Net id.	Name	Nodes#	Links# (unidir.)	Density	Avg. route length
1	Poland	12	36	3.00	2.23
2	Atlanta	15	44	2.93	2.15
3	France	25	90	3.60	2.30
4	USA	26	84	3.23	2.89
5	Europe I	37	114	3.08	2.84
6	Germany II	50	176	3.52	3.01
7	Europe II	28	82	2.93	2.63
8	Telecom Austria	24	102	4.25	2.16
9	New York	16	98	6.13	1.83
10	D-Yuan	11	84	7.64	1.25
11	Germany II	10	90	9.00	1.00

TABLE II
CONDITIONS OF SIMULATIONS

Name	Avg. path bandwidth [Mbit/sec.]	Avg. path creation intensity [1/sec.]	Avg. path lifetime [sec.]	Avg. paths per link
Poland	6.5	130	1	24
Atlanta	40	170	1	25
France	99	330	1	25
USA	2.5	250	1	26
Europe I	316	320	1	24
Germany II	1.5	500	1	26
Europe II	0.8	260	1	25
Telecom Austria	20.000	400	1	25
New York	40	450	1	25
D-Yuan	0.04	560	1	25
Germany II	3.200	750	1	25

preempted bandwidth needs to be minimized then two 10Mbit paths would be chosen. There can also be other measures which the network administrator defines that would be taken into account as well.

In the previous example the choice is simple, but in practice it is not an easy task due to large number of paths in the network. Though we know the optimal algorithms to minimize the number of preemption or to minimize the preempted bandwidth for any set of potential candidates, the execution time is not acceptable. The optimal methods for the two metrics require checking all possible combinations of the potential candidates. Mathematical analysis of the problem shows that the execution times of the optimal methods are of $O(2^n)$ and the problem itself is NP-complete [6], what simply makes it useless. As the optimal methods are unrealizable, we have to look for suboptimal ones. There are indeed at least several algorithms proposed by researchers and we will present them in the next section.

To summarize, here are the problems and decisions to be made when implementing preemptions.

1) Decision: to use preemption-aware routing always or only as a result of failed attempt.
2) Decision: to use global (topology-aware) or local algorithm.
3) Decision: which metrics should be used, e.g. minimizing the number of preemptions, minimizing the preempted band-width, etc.
4) Problem: only the suboptimal methods are acceptable.

The reader should be aware that the above list is only an example as it is not a complete list of problems nor it exhausts all possible variants.

III. PREEMPTION ALGORITHMS

We analyzed and implemented in the simulation program the following heuristic algorithms.

1) GarGop (Garay-Gopal, 1992) [6]. This is the first publicly available study on preemption of the connections in ATM networks. It comprises two independent global algorithms, the first one seeks to minimize the number of preemptions whereas the second one is used for minimizing the preempted bandwidth. Both algorithms do not differentiate the priorities of the paths, but they can be adapted to MPLS networks.
2) Pey (Peyravian, 1994) [7]. The authors present a simple but effective local algorithm, which aims at minimizing both the number of preemptions and the preempted bandwidth, with no possibility to define any other measure. The paper had been published before the MPLS era, but it can be reused for MPLS.
3) OliSco (Oliveira-Scoglio, 2002) [8]. The paper contains one of the first algorithms developed especially for MPLS network, to which most of the authors of the subsequent publications on preemptions refer. It is a simple, versatile and fast local algorithm based on sorting the candidates. This algorithm was later published as the IETF informational document RFC 4829 [9].
4) BlaMeL (Blanchy-Mlon-Leduc, 2003) [10]. The paper describes a simple local mechanism in which the major criterion for selection of candidates is the holding priority of the potential candidates, followed by minimizing the number of preemptions and the preempted bandwidth.
5) KNow (Kaczmarek-Nowak, 2006) [11]. This is an effective global algorithm with flexible metric definition. It ensures minimizing both the number of preemptions and the preempted bandwidth.

In the next section we compare these algorithms based on the simulation results collected for different network topologies.

IV. METHODS OF PERFORMANCE ANALYSIS

To the best knowledge of the authors no comparison study has been published so far on these methods. To provide the results, the algorithms have been implemented into the simulation tool *msim* and a series of studies has been performed.

The *msim* program performs as follows. The simulations base on the configuration (topology and parameters) defined in a text file. The program starts with building the network graph by creating node objects and specifying the links properties, including the link bandwidth. Then several classes of service and types of sources are created. During the simulation time there are sources generated randomly and connected to a random node, what generates to the network a stream of requests for admission. The destination node for every

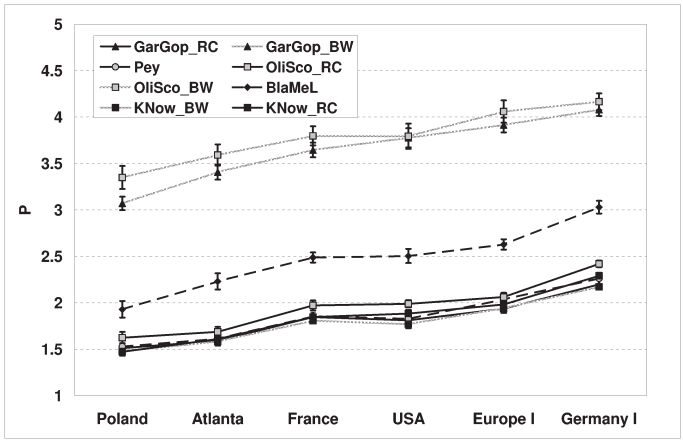

Fig. 1. The average number of preemptions (networks 1-6).

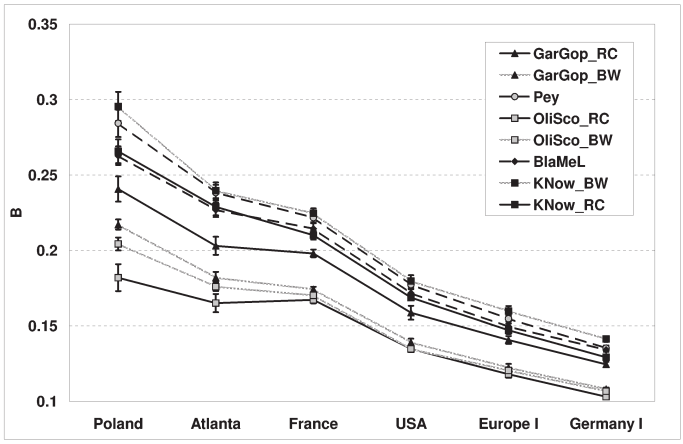

Fig. 2. The preempted bandwidth index B (networks 1-6).

source is also selected randomly with the condition that the destination cannot be the same as the source node. If there is enough bandwidth available then a new path is created for the source and the source is admitted. The lifespan of the source is randomly defined so as after the time period the source is deleted and the used bandwidth is released. In case there is not enough bandwidth available, to create the path the access node requests preemption. The preemption is performed if possible. If it is not successful (e.g. the setup priority is too low) then the source is rejected. The intensity of sources' generation must be specified in the configuration file. For research purposes, it should be on such a level that results in high utilization of link bandwidth what in turn generates frequent preemptions.

The actual parameters which we used in our simulations are presented in Table II. The goal was to adjust the parameters in a way to keep on average about 25 paths per link to make the algorithms work in similar conditions. One can argue that the lifetime of 1 second is too short for the realistic situation. Indeed, in the real MPLS networks the paths are handled as medium or long term connections. However, the value alone is irrelevant and can be seen also as one hour or one day, as long as we use the same unit for intensity of path setups. The practical meaning is this – the connection setup intensity is the average number of path requests per average lifetime of a path.

The simulations run on the path level, i.e. the sources do not generate any packets. This gives us possibility to simulate behavior of networks for quite long time periods. Moreover the packet level simulations do not give any value to performance measures of preemption methods which itself operate on the path level only.

The simulation program counts the preemption-related events and calculates the predefined set of metrics. The simulation time was divided into ten periods where the partial results are collected to calculate at the end of the simulation the average metrics values and the confidence periods. The program allows for automatic repetitions of the simulations with use of modified conditions in a single file and generates a common report including the results from every repetition.

Here is the summary of the conditions of the simulations.

1) To use realistic network topologies we used the network graphs included in the SNDlib at the Zusse Institut Berlin. The key parameters of the topologies are presented in Table I.
2) Every node is the access node. That is, every node can be a terminating point of a path.
3) The intensity of sources for every node is the same. This may not follow the real examples but is taken for simplicity and clearness of the results.

Some of the algorithms are configurable and for those we chose two variants, the first one for minimizing the number of preemptions (relocation count priority, RC) and the second for minimizing the preempted bandwidth (bandwidth priority, BW). The following list contains the evaluated algorithms and the abbreviations we used.

1) GarGop_ RC. This is the Garay-Gopal algorithm with the relocation count priority.
2) GarGop_ BW. This is the Garay-Gopal algorithm with the bandwidth priority.
3) Pey. This is the Peyravian algorithm (no variants available).
4) OliSco_ RC. This is the Oliveira-Scoglio algorithm with the relocation count priority.
5) OliSco_ BW. This is the Oliveira-Scoglio algorithm with the bandwidth priority.
6) BlaMeL. This is the Blanchy-Mélon-Leduc algorithm (no variants available).
7) KNow_ RC. This is our algorithm with the relocation count priority.
8) KNow_ BW. This is our algorithm with the bandwidth priority.

We evaluated the performance of the methods based on metrics which are used commonly by researchers working on preemption, with some modifications when necessary: the number of preemptions, the preempted bandwidth and the combined performance metrics.

1) The average number of preemptions P is the number of paths directly preempted in single procedure, counted only when preemptions happen. The smaller is the value, the fewer reconfigurations are necessary in the network.
2) The average preempted bandwidth should also be kept as

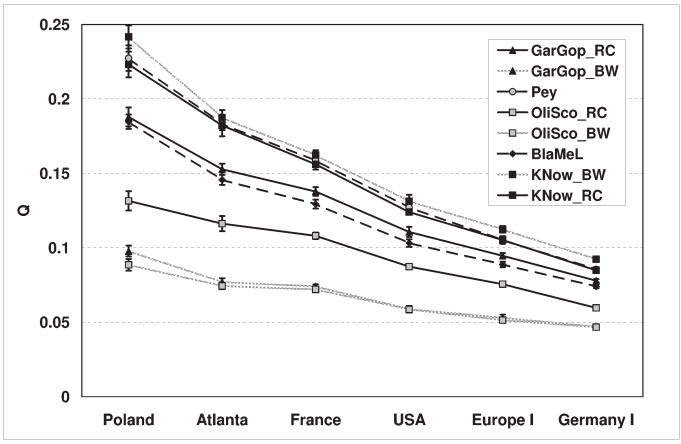

Fig. 3. The average performance metric Q (networks 1-6).

low as possible. Based on the simulation results we have found that the most practical metric is the normalized bandwidth index B, which is the average value of the division of the required bandwidth b_{req} by the preempted network bandwidth. The network bandwidth is the sum of bandwidths b of the preempted paths multiplied by the hop count m of the path (1). In contrast to the bandwidth value, the index B is greater for better algorithm.

$$B = \text{avg}\left(\frac{b_{req}}{\sum_p b(p)mp}\right) \quad (1)$$

3) The performance metric Q is a simple combination of both the number of preemptions and preempted bandwidth (2). This average value spans in range of [0,1] and is higher for more accurate methods.

$$Q = \text{avg}(\frac{B}{P}) \quad (2)$$

All the results contain confidence periods which were calcu-lated for the confidence level of 0.95.

V. Performance Results

We described the number of preemptions as one of the most important measures of the algorithms which should be kept as low as possible. As we can see in Figure 1, the average values of different algorithms differ significantly. Based on the results, we can make the following remarks:

- all the algorithms which take number of preemptions as the main priority and the Pey algorithm perform on similar level, with OliSco_ RC scoring slightly worse results,
- most of the algorithms which do not consider number of preemptions as the priority, perform much worse than the others, with the algorithms Gar-Gop_ BW and OliSco_ BW scoring the worst results here; the rule does not apply to KNow_ BW, which performs similarly to the best algorithms,
- the bigger is the network, the more preemptions happen.

TABLE III
Average Number of Links Where Preemption is Necessary for Two of the Algorithms

Name	GarGop_ RC		OliSco_ BW	
	Links#	Confidence period	Links#	Confidence period
Poland	1.21885	0.02727	1.19644	0,01424
Atlanta	1.26585	0.02650	1.20568	0.01922
France	1.33355	0.01186	1.26750	0.01152
USA	1.42445	0.02703	1.34394	0.01351
Europe I	1.54402	0.01236	1.45400	0.01152
Germany I	1.70255	0.01643	1.53730	0.01553

This is not surprising that the methods GarGop_ BW and OliSco_ BW do not score best results here, but the distance from other methods may be greater than expected. Taking this into consideration, we can see that the methods should not be used if we want to minimize the number of preemptions, as they make it double. The best choices are GarGop_ RC, Pey and both KNow methods.

The second most important measure is the preempted bandwidth which should be minimized to lower the amount of affected traffic. Note that the comparison presented in Figure 2 brings even more interesting results. Note that here the bandwidth metric B is presented, with higher values corresponding to better performance.

The obtained results do not follow the expected behavior, that the algorithms which perform well in number of preemptions should score worse results for preempted bandwidth. We rather observe that some of the methods perform well for both cases. The algorithms which perform best here are Pey and KNow_ BW are also among the best in the first category. This means that as long as we choose one of the most important priorities, e.g. number of preemptions or the preempted bandwidth, these methods can be used without adjusting them by the network administrator. This removes the burden of additional investigations necessary to choose the best algorithm for any specific case and it lets us keep a single, universal implementation at the network device.

The results showing that there are universal methods were for us the drivers to develop a common metric, which would include both the number of preemptions and the preempted bandwidth into a single performance measure. The simple common metric Q (2) is used for such purpose. The corresponding results are presented in Figure 3, where higher value means better method. The metric is defined in the way that the optimal algorithm would get the results of 1.0, provided that there are available paths that fit perfectly into the necessary bandwidth. The condition cannot be met in practice if the bandwidths of the paths are not equal. As we can see, the measured algorithms score 0.2-0.25 at best. The best algorithm here is KNow_ BW, following by Pey and KNow_ RC.

Taking the network size into consideration, the performance of all the evaluated methods degrades when the network grows. This is the effect of statistically longer paths in bigger networks. That in turn increases the average number of links, where preemption is necessary, as shown in Table III. The combination of longer paths and increasing number of links with preemption leads to larger losses of suboptimal methods

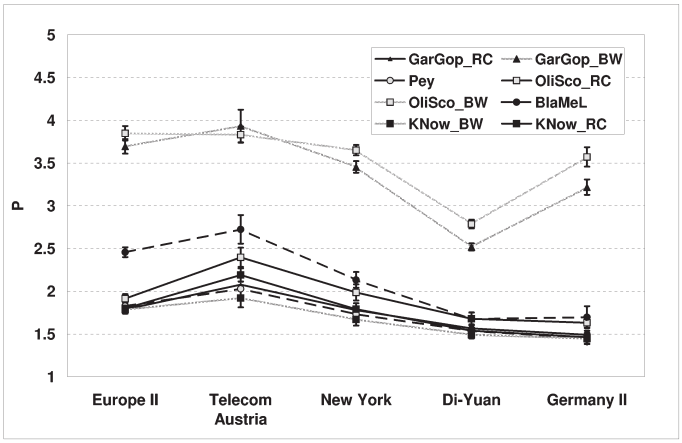

Fig. 4. The average number of preemptions (networks 7-11).

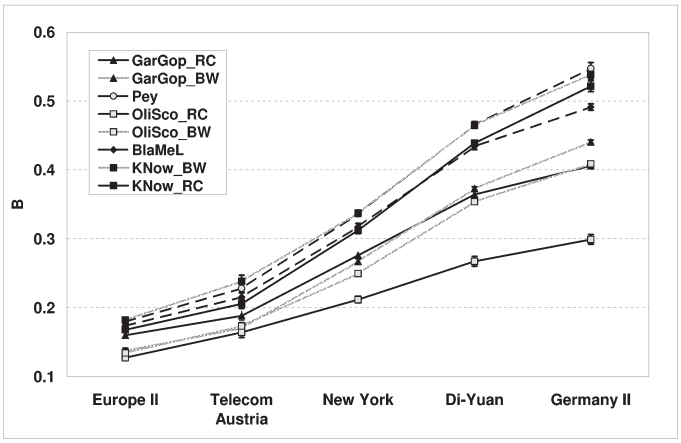

Fig. 5. The preempted bandwidth index B (networks 7-11).

what is in turn reflected in the decrease in performance. It might be surprising that the number of links with preemption is different for different algorithms.

For the second series of topologies we wanted to verify the influence of density of the network on the performance results. In the Figures 4, 5, 6 we present the results for the number of preemptions P, preempted bandwidth index B and the combined index Q, respectively. As can be seen in Figure 4, there is no strong dependency on the network density. Though we can observe a decrease of number of preemptions for denser networks, the first network (Europe II) does not follow the rule. If we compare the results with the networks properties in Table I, especially the number of links, we can say that the number of preemptions:

- is smaller for denser networks,
- larger in larger networks, in terms of number of links.

The relation of network density and the preempted bandwidth presented in Figure 5 shows high dependency. Clearly for denser networks the bandwidth performance rises. This is the effect of shorter paths, as shown in Table IV.

Knowing the results for the number of preemptions and the bandwidth index, the results of the combined metric presented in Figure 6 follow our expectations. Overall performance of the algorithms is better for denser networks, mainly thanks to shorter paths.

TABLE IV
AVERAGE ROUTE LENGTH (NETWORKS 7-11)

Network	Route length	Confidence period
Europe II	3.51567	0.03412
Telecom Austria	2.47573	0.07128
New York	1.76237	0.00891
D-Yuan	1.33201	0.01100
Germany II	1.07896	0.01006

VI. CONCLUSIONS

In this paper we explained the role of performance in the MPLS traffic engineering and we discussed the problems and dilemmas which a researcher faces when deciding on which algorithm should be used. We presented the differences in performance of well known preemption algorithms based on the simulation results using realistic network topologies.

The analysis of the simulation results show that the difference in performance of different algorithms in the same network conditions differs significantly. Additionally, the performance degrades when the network grows or becomes sparser. One of the most important results of the simulations indicates that some algorithms can be used in universal preemption methods that perform well for both of the most common criteria: number of preemptions and preempted bandwidth. That greatly simplifies the choice of the best algorithm. We also show that using the proper preemption algorithm can decrease the number of preemptions by up to 50%, which in turn has direct positive impact on the level of service. This conclusion is of great value to everyone involved in implementing of preemptions in MPLS networks.

For the future work in the area of preemption we aim to focus on the effectiveness of preemption and its cost. We would like to find answer to the question, if and when it is reasonable to use global preemption rather than the local one. Another open issue is the overall change in path acceptance level between the networks with and without preemption.

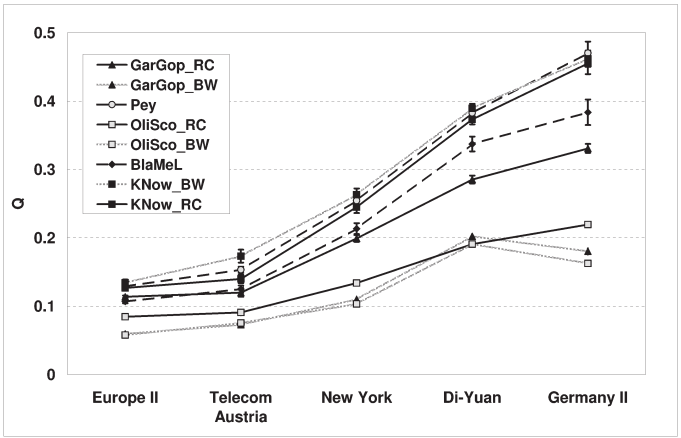

Fig. 6. The average performance metric Q (networks 7-11).

REFERENCES

[1] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," *RFC 3031*, January 2001.

[2] B. S. Davie and Y. Rekhter, *MPLS: Technology and Applications*. Morgan Kaufmann Publishers, 2000.

[3] M. Kolon and J. Doyle, *Juniper Networks Routers: The Complete Reference*. Osborne/McGraw-Hill, 2002.

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

[5] J. A. Garay and I. S. Gopal, "Call Preemption in Cmmunication Networks," in *Proceedings of INFOCOM '92*, Florence, 1992, pp. 1043–1050.

[6] M. Peyravian, "Providing Different Levels of Network Availability in High-speed Net-works," in *Proceedings of GLOBECOM '94*, 1994, pp. 941–945.

[7] J. de Oliveira, C. Scoglio, I. F. Akyildiz, and G. Uhl, "A New Preemption Policy for Diffserv-Aware Traffic Engineering to Minimize Rerouting," in *Proceedings of IEEE INFOCOM 2002*, vol. 2, June 2002, pp. 695–704.

[8] J. de Oliveira, J. P. Vasseur, L. Chen, and C. Scoglio, "Label Switched Path (LSP) Preemption Policies for MPLS Traffic Engineering," *RFC 4829*, April 2007.

[9] F. Blanchy, L. Melon, and G. Leduc, "Routing in a MPLS Network Featuring Preemption Mchanisms," in *Proceedings of ICT 2003*, 2003.

[10] S. Kaczmarek and K. Nowak, "A New Heuristic Algorithm for Effective Preemption in MPLS Networks," *Workshop on High Performance Switching and Routing*, pp. 337–342, 2006.

[11] ——, "A Simulation Tool for Traffic Engineering Methods and QoS Evaluation of MPLS Networks," in *Proceedings of SIMUTools 2009*, Rome, 2009, art. no. 54.