Dynamic Routing in QoS-aware Traffic Engineered Networks[⋆]

Stefano Avallone[1], Fernando Kuipers[2], Giorgio Ventre[1], and Piet Van Mieghem[2]

[1] COMICS Lab, Dipartimento di Informatica e Sistemistica, Università di Napoli Federico II, Via Claudio 21, 80125 Napoli, Italy
{stavallo,giorgio}@unina.it

[2] Network Architectures and Services, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, P.O. Box 5031, 2600 GA Delft, The Netherlands
{F.A.Kuipers,P.VanMieghem}@ewi.tudelft.nl

Abstract. The problem of finding multi-constrained paths has been addressed by several QoS routing algorithms. While they generally satisfy the application requirements, they often do not consider the perspective of service providers. Service providers aim at maximizing the throughput and the number of accepted requests. These goals have been addressed by traffic engineering algorithms considering bandwidth as the sole application requirement. We propose a proper length function for an existing QoS routing algorithm (SAMCRA) that attempts to optimize network utilization while still offering QoS guarantees. This paper presents a comparison between several proposed algorithms via simulation studies. The simulations show that SAMCRA with a proper length performs similarly or even better than the best among the other algorithms and it has a fast running time.

1 Introduction

The Internet research community is making a great effort in order to define efficient network management and control functions. The driving forces behind this effort are new applications with specific performance requirements. For instance, real-time applications need delay and jitter guarantees, while a financial transaction must have low or virtually no packet loss. By offering service differentiation combined with the maximization of throughput, ISPs can increase their revenues. The challenge is therefore to define a routing algorithm which meets the users' requirements and which optimizes network resources.

Many algorithms [1] have been proposed to find the shortest path subject to multiple constraints. This problem, called MCOP (Multi-Constrained Optimal

⋆ This work has been carried out partially under the financial support of the "Ministero dell'Istruzione, dell'Università e della Ricerca (MIUR)"in the framework of the FIRB Project "Middleware for advanced services over large-scale, wired-wireless distributed systems (WEB-MINDS)".

Path), is NP-complete. Therefore, although there exist exact algorithms such as SAMCRA [2], mainly heuristics have been proposed for this problem. Such algorithms usually do not address the maximization of the throughput and the number of admitted calls. Instead, optimizing these parameters is the goal of another class of algorithms, denoted as traffic engineering algorithms. While focused on the behaviour in a dynamic scenario, most of them do not take into account additive QoS constraints and only consider bandwidth.

In this paper we present a routing scheme that aims at maximizing throughput (or minimizing blocking), while satisfying the users' QoS requirements. It is our goal to combine these two objectives as efficiently as possible. We propose to use SAMCRA [2] with a special path length definition that guarantees the QoS constraints and accounts for the traffic engineering objectives. For clarity, we name this variant SAMCRA-B.

The performance of SAMCRA-B and several other algorithms is evaluated through simulations. All the considered algorithms do not make use of any a-priori knowledge about either predicted traffic or future demands. In the literature, such algorithms are denoted as online. We assume the knowledge of quasi-static information such as the network topology and the set of ingress-egress nodes of the network. The only dynamic information is the residual bandwidth (i.e. the portion of the link capacity not yet reserved) of each link in the network.

This paper is structured as follows. In Section 2 we give a formal definition of the considered routing problem. Some solutions for routing bandwidth-guaranteed paths are discussed in Section 3. Section 4 overviews SAMCRA and the choice of a proper length function. The performance studies are shown in Section 5. Section 6 concludes our work.

2 Problem Statement

The network is modelled as a graph $G(N, E)$, where N is the set of nodes and E is the set of links. With a slight abuse of notation we will also denote by N and E, respectively, the number of nodes and the number of links. Each link $l \in E$ is assigned an $(m+1)$-dimensional QoS link weight vector $\boldsymbol{w}(l) = [w_0(l), w_1(l), \ldots w_m(l)]$, where $w_0(l)$ is the available bandwidth on link l and the other components are the values of m additive QoS measures. Additionally, the capacity of a link l is denoted by $C(l)$.

A flow request is defined by a triple $(s, d, \boldsymbol{Q})$, where s is the source node, d is the destination node and $\boldsymbol{Q} = [Q_0, Q_1, \ldots Q_m]$ is a vector representing its QoS requirements. Specifically, Q_0 is the requested bandwidth while the other components are constraints on the values of the additive QoS measures along the path. Even though minimum (maximum) QoS constraints can be easily treated by omitting all links which do not satisfy the requirement, we explicitly consider available bandwidth due to its central role played in resource optimization strategies. Multiplicative QoS measures are disregarded because, if we assume independent measures over the links, we can transform them into additive QoS measures by taking the logarithm [2].

When a flow request arrives, the routing algorithm searches for a feasible path P that obeys:

$$\begin{cases} w_0(P) \overset{def}{=} \min_{l \in P} w_0(l) \geq Q_0 \\ w_i(P) \overset{def}{=} \sum_{l \in P} w_i(l) \leq Q_i, \quad \forall i = 1, \dots m \end{cases} .$$

In case no feasible path is found, the request is rejected. In presence of multiple feasible paths, the algorithm chooses the one which is thought to optimize network utilization. Typically, a path length function is defined and the feasible path with the smallest length is selected.

2.1 Discussion on QoS link weights

This subsection discusses the setting of QoS link weights ($\boldsymbol{w}(l)$) in a dynamic scenario. The guideline is the fulfillment of the QoS requirements of the flows. It is safe to state that the link weight associated with the available bandwidth should be as close as possible to the current bandwidth availability. As far as additive link weights, a path P returned by an exact algorithm is such that $\sum_{l \in P} w_i(l) \leq Q_i, \ i = 1, \dots m$. But, the QoS requirements of a flow are satisfied if the *perceived* QoS is within the constraints, i.e. $\sum_{l \in P} q_i(l) \leq Q_i, \ i = 1, \dots m$, where $q_i(l)$ is the value of the i-th QoS measure experienced crossing link l.

Assume to set the additive QoS link weights ($w_i(l)$) equal to the current experienced values (i.e. $q_i(l)$). Hence $\sum_{l \in P} q_i(l) = \sum_{l \in P} w_i(l) \leq Q_i$ and the QoS constraints are met. But, as a consequence of routing new flows on links of P, the actual QoS values $q_i(l)$ deteriorate and therefore the QoS granted to already admitted flows may not be preserved. Instead, we assume that the QoS link weights are constant and independent of the current link status ($q_i(l)$). Their value is an upper bound to the actual QoS value, in the sense that if the allocated bandwidth is less than the link capacity, then the QoS values experienced by packets crossing the link do not exceed the QoS weights. This assures that $\sum_{l \in P} q_i(l) \leq \sum_{l \in P} w_i(l) \leq Q_i, i = 1, ..., m$, i.e. the additive QoS constraints will be satisfied even after new flows are routed.

3 Existing Traffic Engineering Algorithms

Among the earliest proposed algorithms, widest-shortest path [3] (labeled as "WSP(MinHop)" throughout this paper) selects the path with the minimum hop count among all paths having sufficient residual bandwidth. If there are several such paths, the one with the maximum residual bandwidth is selected.

Most recently proposed algorithms are inspired by the work of Kar, Kodialam and Lakshman [4]. They presented an online routing algorithm (MIRA) based on the concept of minimum interference. The amount of interference on a particular source-destination pair (s, d) due to routing a flow between some other source-destination pair is defined as the decrease in the maxflow between s and d. The maxflow [5] value is an upper bound on the total amount of bandwidth that can be routed between two edge nodes. The minimum interference path between a particular source-destination pair is the path which maximizes the minimum maxflow between all other source-destination pairs. The idea is that a new request must follow a path that does not "interfere excessively" with a route that may be critical to satisfy a future demand. The problem of finding the minimum interference path is proved to be NP-hard. Therefore, Kar et al. [4] proposed to determine appropriate link costs, prune links with insufficient available bandwidth and compute the shortest path in the pruned topology.

Wang et al. [6] proposed a different definition for link costs in MIRA. We denote this variant of MIRA as "NewMIRA". The performance evaluation in [6] shows that, in a dynamic scenario, NewMIRA outperforms MIRA.

Banerjee and Sidhu [7] proposed two algorithms: TE-B, which takes into account only a bandwidth requirement, and TE-DB, which considers also a delay constraint. The authors introduced three objectives for traffic engineering: (a) reducing the blocking of flows, (b) minimizing network cost and (c) distributing network load. This formulation has three objective functions (plus the delay constraint in the case of TE-DB) and is proved to be NP-complete [7]. Banerjee and Sidhu presented another formulation in which objective functions (a) and (b) are transformed into constraints. Both TE-B and TE-DB use TAMCRA [8], the predecessor of SAMCRA [2], to find a set of k paths satisfying the set of constraints and then select the one with the shortest length according to (c).

Iliadis and Bauer [9] introduced a new class of routing algorithms, called SMIRA (simple minimum-interference routing algorithms). These algorithms evaluate the interference on a source-destination pair by means of a k-shortest-path-like computation instead of a maxflow computation. The set of k paths between a source-destination pair (s, d) is determined by first computing the widest-shortest path [3] between s and d. Then, all the links along this path with a residual bandwidth equal to the bottleneck bandwidth of the path are pruned. The second path is the widest-shortest path in the pruned topology. This procedure is repeated until either k paths are found or no more paths are available. The cost of links belonging to the set of k paths is increased proportionally to the weight of the path and the ratio of bottleneck bandwidth to residual bandwidth. Iliadis and Bauer [9] proposed two algorithms, MI-BLA and MI-PA. The simulations in [9] show that MI-PA outperforms MI-BLA.

4 SAMCRA

To guarantee QoS constraints and optimize network resource usage, we decided to use SAMCRA [2] with a new path length function and to study its behaviour in a dynamic scenario. First, we briefly review the basic concepts on which SAMCRA relies. The length of a path P proposed in [2] is a non-linear function of the m additive QoS measures it considers :

$$L(P) = \max_{1 \leq i \leq m} \frac{w_i(P)}{Q_i} \tag{1}$$

so that path P satisfies the constraints when $L(P) \leq 1$. An important corollary of a non-linear path length is that the subsections of shortest paths in multiple dimensions are not necessarily shortest paths. This necessitates a k-shortest path approach, which is essentially Dijkstra's algorithm that does not stop when the destination is reached, but continues until the destination has been reached k times. Not all sub-paths are stored, but an efficient distinction based on non-dominance is made: a (sub)-path P_1 is dominated by a (sub)-path P_2 if $w_i(P_2) \leq w_i(P_1)$ for $i = 1, ..., m$, with an inequality for at least one link weight component i. SAMCRA only considers non-dominated (sub)-paths.

We refer for more details on the above concepts, possible improvements and an implementation of the algorithm to [2]. Here, we explain the above concepts through a simple example. Like Dijkstra's algorithm, SAMCRA starts from the source node and explores the neighboring nodes while moving toward the destination node. Unlike Dijkstra, SAMCRA may have to store more than only the shortest sub-path for each visited node. To explain this point, consider the simple network of Figure 1. The vector $\boldsymbol{w}(l) = [w_0(l), \ldots w_3(l)]$ of QoS link weights is shown around each link. SAMCRA does not consider the available bandwidth constraint, but it suffices to prune from the network graph all the links with insufficient available bandwidth and run the algorithm on the reduced graph. Suppose SAMCRA has to route a flow from A to G subject to the QoS constraint vector $Q = [5, 14, 11, 22]$. Three sub-paths are available from the source

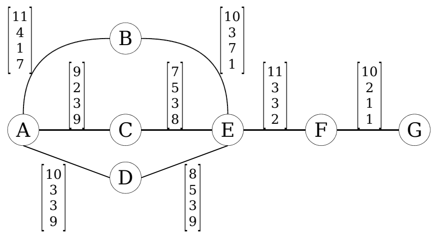

A
B
C
D
E
F
G
$\begin{bmatrix}11\\4\\1\\7\end{bmatrix}$
$\begin{bmatrix}9\\2\\3\\9\end{bmatrix}$
$\begin{bmatrix}10\\3\\7\\1\end{bmatrix}$
$\begin{bmatrix}7\\5\\3\\8\end{bmatrix}$
$\begin{bmatrix}11\\3\\3\\2\end{bmatrix}$
$\begin{bmatrix}10\\2\\1\\1\end{bmatrix}$
$\begin{bmatrix}10\\3\\3\\9\end{bmatrix}$
$\begin{bmatrix}8\\5\\3\\9\end{bmatrix}$

Fig. 1. Simple network to illustrate SAMCRA's principles

node A to the intermediate node E. The lengths of those sub-paths and of the corresponding paths from A to G are:

$$\begin{aligned}
L(P_{ABE}) &= \max\left\{\frac{4+3}{14}, \frac{1+7}{11}, \frac{7+1}{22}\right\} = 0.73 :\rightarrow: L(P_{ABEFG}) = \max\left\{\frac{12}{14}, \frac{12}{11}, \frac{11}{22}\right\} = 1.09 \\
L(P_{ACE}) &= \max\left\{\frac{2+5}{14}, \frac{3+3}{11}, \frac{9+8}{22}\right\} = 0.77 :\rightarrow: L(P_{ACEFG}) = \max\left\{\frac{12}{14}, \frac{10}{11}, \frac{20}{22}\right\} = 0.91 \\
L(P_{ADE}) &= \max\left\{\frac{3+5}{14}, \frac{3+3}{11}, \frac{9+9}{22}\right\} = 0.82 :\rightarrow: L(P_{ADEFG}) = \max\left\{\frac{13}{14}, \frac{10}{11}, \frac{21}{22}\right\} = 0.95 \ .
\end{aligned}$$

The shortest sub-path, P_{ABE}, leads to a non-feasible path ($L(P_{ABEFG}) > 1$). If the algorithm stores just the shortest sub-path in the intermediate nodes, it erroneously concludes that a feasible path does not exist (the two other paths are feasible). In order to reduce complexity (while still returning the exact solution), SAMCRA does not store all the sub-paths but discards the dominated ones. In the example, sub-path P_{ADE} is dominated by P_{ACE} since $w_i(P_{ACE}) \leq w_i(P_{ADE})$ for $i = 1, 2, 3$. Given (1) as path length function and non-negative QoS link weights, sub-path P_{ADE} can be safely discarded.

As discussed in [2] SAMCRA can be used with any path length function, but the definition (1) is a function of the QoS link weights ($w_i(l), i = 1, \ldots m$) and of the QoS constraints ($Q_i, i = 1, \ldots m$). As discussed in Section 2.1, static QoS link weights should be considered. Thus, if link weights are load-independent, the path length function (1) does not take into account network utilization. We expect that a better dynamic behavior can be achieved by letting the path length be a function of dynamic information such as the available bandwidth. We assume:

$$L(P) = \sum_{l \in P} c(l) \tag{2}$$

where $c(l)$ is a link cost that depends on dynamic information related to link l. For clarity we name this variant SAMCRA-B. It selects the shortest path according to (2) among those satisfying the QoS constraints. Using such a path length function requires to add one more parameter to the dominance check. SAMCRA-B can discard a sub-path P_1 when there exists a sub-path P_2 such that $w_i(P_2) \leq w_i(P_1)$ for $i = 1, \ldots m$ and $L(P_2) \leq L(P_1)$. To clarify why, consider again the example shown in Figure 1 and compute the lengths (according to (2)) of the three sub-paths from A to E and the corresponding paths from A to G using, for instance, $c(l) = \frac{1}{w_0(l)}$, i.e. the cost of a link is the reciprocal of the available bandwidth:

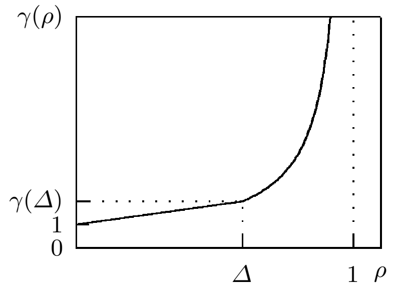
$$L(P_{ABE}) = \frac{1}{11} + \frac{1}{10} = 0.19 :\rightarrow: L(P_{ABEFG}) = \frac{1}{11} + \frac{1}{10} + \frac{1}{11} + \frac{1}{10} = 0.38$$
$$L(P_{ACE}) = \frac{1}{9} + \frac{1}{7} = 0.25 :\rightarrow: L(P_{ACEFG}) = \frac{1}{9} + \frac{1}{7} + \frac{1}{11} + \frac{1}{10} = 0.44$$
$$L(P_{ADE}) = \frac{1}{10} + \frac{1}{8} = 0.23 :\rightarrow: L(P_{ADEFG}) = \frac{1}{10} + \frac{1}{8} + \frac{1}{11} + \frac{1}{10} = 0.42$$

P_{ABE} is the shortest sub-path but leads to path P_{ABEFG}, which is not feasible because $w_2(P_{ABEFG}) > Q_2$. Thus, the algorithm cannot store just the shortest sub-path in each intermediate node. On the other hand, disregarding the condition on the length of the sub-paths would cause sub-path P_{ADE}, which leads to the shortest feasible path P_{ADEFG}, to be discarded in favour of P_{ACE}. Instead, by also comparing path lengths it is still possible to achieve a correct search space reduction and return the shortest path according to (2) within the constraints. Indeed, sub-path P_{ADE} is not dominated by P_{ACE} because $L(P_{ACE}) > L(P_{ADE})$ and therefore it is not discarded.

The worst-case complexity of SAMCRA is $O(kN\log(kN) + k^2mE)$ [2], where k is the maximum amount of simultaneously stored paths. In [2], it is shown that if the QoS link weights $w_i(l)$ and the QoS constraints Q_i are integers, SAMCRA has a pseudo-polynomial-time complexity. The complexity of SAMCRA-B is that of SAMCRA, apart from a larger value of k because SAMCRA-B has to check one more condition and therefore less paths can be discarded.

4.1 Link cost function

This section illustrates the link cost function we introduce to improve the dynamic behavior of SAMCRA-B with respect to SAMCRA. The link cost $c(l)$ is defined as a function γ (eq. 3) of the link utilization $\rho = \frac{C(l)-w_0(l)}{C(l)}$ (ratio of the reserved bandwidth to the total capacity). The function $\gamma(\rho)$, depicted in



$$\gamma(\rho) = \begin{cases} \dfrac{\gamma(\Delta)-1}{\Delta}\rho + 1 \text{ if } \rho \leqslant \Delta \\ \\ \gamma(\Delta)\dfrac{1-\Delta}{1-\rho} \qquad \text{otherwise} \end{cases} \tag{3}$$

Fig. 2. Link cost function

Fig. 2, depends on two design parameters, Δ and $\gamma(\Delta)$. The rationale behind such proposal is that a minimum-hop path approach is preferable when network load is low, since it prevents longer paths from consuming more resources. On the other hand, when network load is high, it is preferable to use the path with the maximum available bandwidth. Consuming all the available bandwidth on some links could cause future requests to be blocked. Such a twofold behavior may be achieved by appropriately setting each link cost depending on its resource occupation. The link cost function we propose is divided in two segments whose junction takes place in $(\Delta, \gamma(\Delta))$. For values of the link utilization ρ less than Δ, the link cost grows linearly. As soon as ρ becomes greater than Δ, the link cost tends to infinity as ρ approaches 1. With such a function, link costs are only slightly differing for low traffic loads. Low slope values of the linear segment let the routing algorithm behave similarly to the minimum-hop path routing. Instead, small load variations reflect in substantial link cost differences as soon as the load on the link increases beyond a specified threshold Δ. In this way, links having more available bandwidth are highly preferable.

The values assigned to Δ and $\gamma(\Delta)$ determine the relative importance of the two approaches. We expect that their "optimal" values are dependent on the network topology and the traffic load. As an attempt, we have used $\Delta = 0.6$ and $\gamma(\Delta) = 1.5$ for the simulations shown in the next section. From now on, we will implicitly consider (3) as the link cost function of SAMCRA-B.

Table 1. Random variables used to specify network flows

Name	Description
`IntArriv`	Inter-arrival time between two successive flow requests
`Source`	Source node
`Dest`	Destination node
`FlowDur`	Flow duration
`Bwd`	Requested bandwidth
`QoS`$_i$	i-th additive QoS constraint

5 Performance Studies

The performance studies of this section aim to compare the algorithms based on the minimum interference concept (New MIRA, TE-DB, MI-PA) to SAMCRA and SAMCRA-B and to evaluate the possible gain achieved by using the new path length function (3). The experiments were carried out on several topologies generated by BRITE [10]. We used two router-level models, Barabasi-Albert and Waxman. All the topologies have 100 nodes and a different number of links per new node. For each topology, 10 nodes are randomly chosen to act as edge routers, the entry and exit points for the network traffic. The capacity of the links is uniformly distributed between 100 and 1024 units. We considered two additive QoS constraints ($m = 2$), the first uniformly distributed between 3 and 8 units, while the second ($w_2(l)$) uniformly distributed between 4 and 9 units. All links are symmetric, with respect to both capacity and QoS link weights.

We have developed a flow-level simulator[3] to analyze and compare the performance of different routing algorithms in a dynamic scenario. Our simulator makes use of several random variables to specify the characteristics of network flows (Table 1). For all the presented simulations, source and destination nodes are chosen uniformly among the set of edge nodes. We studied the performance of the routing algorithms in the generated topologies under different loads. Each test was repeated 20 times with different seeds for the random variables. For each of these 20 iterations, the algorithms under evaluation faced the same set of flow requests. Each iteration involved the generation of 120000 flows. The first 20000 were not considered in our analysis, as they represent a warm-up period needed by the network in order to reach a steady state regime.

For each iteration we computed the call blocking rate (CBR) and the bandwidth blocking rate (BBR) achieved by each algorithm:

$$CBR = \frac{\text{number of rejected flows}}{\text{total number of flows}}, BBR = \frac{\sum_{\text{rejected flows}} \text{requested bandwidth}}{\sum_{\text{all the flows}} \text{requested bandwidth}} .$$

We also computed the throughput after the processing of each new flow request as the sum of the bandwidth requested by the flows crossing the network at that time. In order to get a smooth throughput curve, we first computed the mean over each window of 5000 throughput samples for each iteration and then the average of the corresponding values obtained from the 20 iterations. Finally, we measured the average processor time spent by each algorithm to select a path.

[3] NetSim++, developed at Delft University of Technology

Table 2. Scenario 1: flow and topology parameters

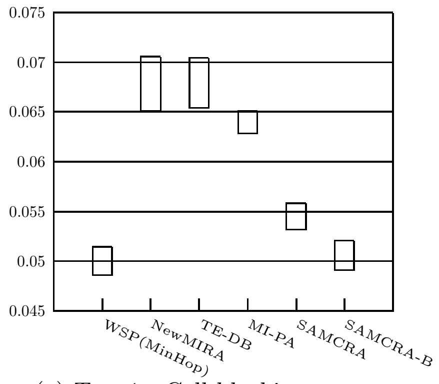
Test	1	2	3
`IntArriv`	Exp(1/0.15)	Exp(1/0.11)	Exp(1/0.5)
`FlowDur`	Exp(1/250)	Exp(1/320)	Exp(1/175)
`Bwd`	U(1,10)	U(1,10)	$\begin{cases} \text{U}(1,10) & \text{with } P = 0.75 \\ \text{U}(80,100) & \text{with } P = 0.25 \end{cases}$
$\texttt{QoS}_1$	U(792,800)		
$\texttt{QoS}_2$	U(891,900)		
Topology model	Barabasi	Waxman	Barabasi
Links	294	200	197

5.1 Large QoS constraints

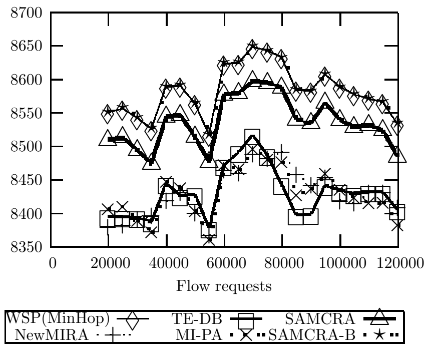
The purpose of this subsection is to compare all the algorithms from the viewpoint of resource optimization. Since some of them select a path disregarding additive QoS constraints, we chose the QoS constraints large such that all algorithms can return a path that obeys these constraints. We have carried out a number of simulations using several topologies and different loads. In this subsection we illustrate three different tests that are representative of the different cases we observed. Table 2 shows how we set the random variables that specify a flow and the model and the number of links of the topologies we used. The results are presented in Figure 3. For each algorithm, we computed the mean μ and the standard deviation σ of the CBR from 20 iterations. Each bar shown in Figures 3(a), 3(c) and 3(e) represents the interval $(\mu - \sigma, \mu + \sigma)$ related to the CBR achieved by each algorithm. In tests 1 and 2 (Figures 3(a) and 3(c)), SAMCRA-B achieves respectively a slightly larger CBR than the minimum and the minimum CBR. The CBR of New Mira and TE-DB is higher (around 30%) than the minimum, so as that of SAMCRA (below 10%). In test 3 (Figure 3(e)), instead, the minimum CBR is achieved by New MIRA, closely followed by TE-DB and SAMCRA-B. In all the simulations we have carried out SAMCRA-B achieves the minimum CBR or a CBR close to the minimum. The bandwidth blocking rate results (not shown here) are similar to CBR ones, except that the algorithms based on the maxflow concept (New MIRA and TE-DB) perform better in terms of BBR than CBR. This behaviour suggests that New MIRA and TE-DB accept those flows with larger bandwidth requirement.

While the CBR plot shows a mean value over all the iterations, the average throughput plot gives us information on the average behaviour during an iteration. Figures 3(b), 3(d) and 3(f) indicate that the behavior of the algorithms from the viewpoint of throughput is similar to that in terms of CBR. In the sense that the algorithm achieving the minimum CBR also presents the maximum throughput. Throughput plots enable to ascertain that the difference in performance between algorithms is maintained during the whole iteration.

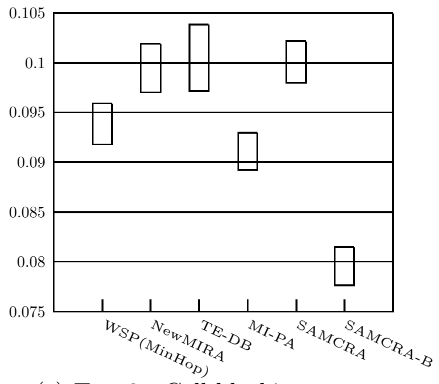
The analysis of the average computation times reveals that SAMCRA approximately requires the same time ($\approx 5 \cdot 10^{-4}$s) as widest-shortest path. As expected, due to the less efficient search space reduction, SAMCRA-B ($\approx 9 \cdot 10^{-4}$s) is slower than SAMCRA. Instead, the time required by New MIRA and TE-DB ($\approx 6 \cdot 10^{-2}$s) is two orders of magnitude larger than that of SAMCRA.



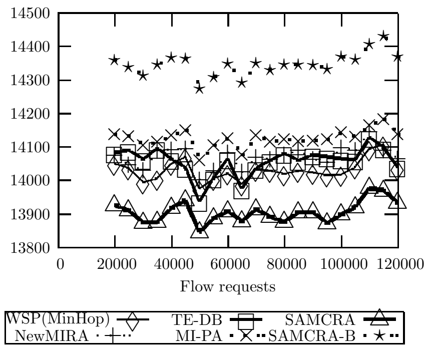
(a) Test 1 - Call blocking rate



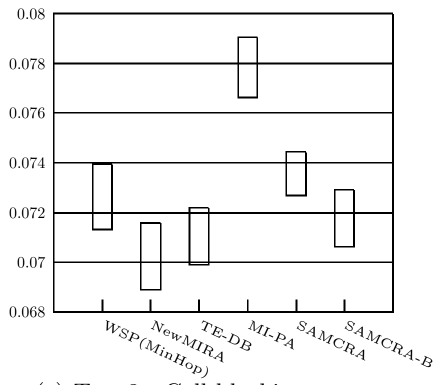
(b) Test 1 - Throughput



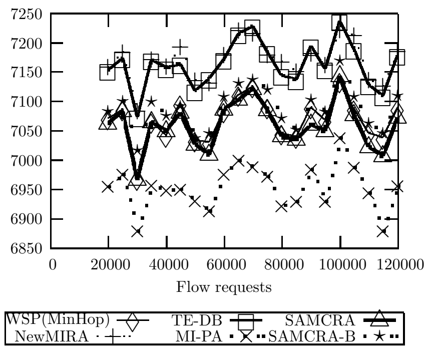
(c) Test 2 - Call blocking rate



(d) Test 2 - Throughput



(e) Test 3 - Call blocking rate

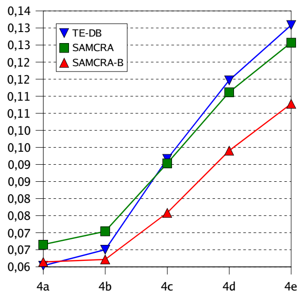


(f) Test 3 - Throughput

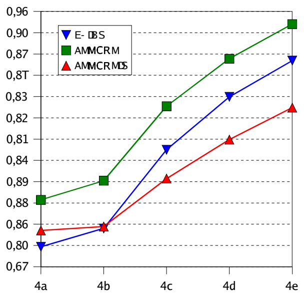
Fig. 3. Scenario 1: simulation results

Table 3. Scenario 2: flow parameters

Test	4a	4b	4c	4d	4e
`IntArriv`	Exp(1/0.33)				
`FlowDur`	Exp(1/200)				
`Bwd`	$\begin{cases} U(1, 10) & \text{with } P = 0.75 \\ U(80, 100) & \text{with } P = 0.25 \end{cases}$				
`QoS`$_1$	U(792,800)	U(30,50)	U(24,44)	U(22,42)	U(21,41)
`QoS`$_2$	U(891,900)	U(35,55)	U(29,49)	U(27,47)	U(26,46)



(a) Call blocking rate



(b) Bandwidth blocking rate

Fig. 4. Scenario 2: simulation results

5.2 Tight QoS constraints

This set of simulations aims at comparing the algorithms which select a path taking explicitly into consideration additive QoS constraints: SAMCRA, SAMCRA-B and TE-DB. They are evaluated from the viewpoint of resource optimization as the QoS constraints become more stringent. We present five tests, which differ only in the QoS constraints (see Table 3). The topology is the same as that used in test 2. Figure 4 shows the plots representing the average call and bandwidth blocking rates for each test. When QoS constraints are large enough (test 4a), the minimum CBR is achieved by TE-DB. But, as the QoS constraints become more stringent, SAMCRA-B performs the best. Moreover, the gap between the average CBR of SAMCRA-B and those of SAMCRA and TE-DB increases. This suggests that SAMCRA-B is less sensitive to the tightening of QoS constraints than SAMCRA and TE-DB are. Also, the CBR of SAMCRA is initially larger than that of TE-DB, but eventually it becomes smaller. This suggests that SAMCRA, too, is less sensitive to the tightening of QoS constraints than TE-DB. The same conclusions can be drawn from the viewpoint of BBR (Figure 4(b)). Figure 4(b) also confirms our insight regarding the fact that TE-DB performs better from the viewpoint of BBR than CBR. Finally, the path computation times are similar to those indicated in the previous section.

6 Conclusions

We proposed a new path length function for SAMCRA and carried out simulation studies in order to compare SAMCRA-B to previous algorithms for dynamically routing requests having a bandwidth requirement and a number of constraints on additive QoS measures. Two scenarios have been analyzed, the first under loose QoS constraints and the other by the tightening of QoS constraints. For every test of the first scenario the call blocking rate of SAMCRA-B was the minimum or very close to the minimum. The simulations therefore revealed that the proposed path length function of SAMCRA-B (based on the current bandwidth availability) allows a considerable advantage over SAMCRA. The second scenario showed that SAMCRA-B performs better and better than the other algorithms (SAMCRA and TE-DB) when decreasing the QoS constraints. SAMCRA, too, reduces its gap from TE-DB as the QoS constraints become more stringent. If we also consider the analysis of path computation times, we can conclude that SAMCRA-B achieves the best performance at a low computational cost.

The scenarios covered by our simulations were necessarily limited and therefore the results only indicate a potential for SAMCRA with a properly chosen path length function. Further investigation is needed to confirm our claim.

References

1. Kuipers, F., Van Mieghem, P., Korkmaz, T., Krunz, M.: An overview of constraint-based path selection algorithms for QoS routing. IEEE Communications Magazine **40** (2002) 50–55
2. Van Mieghem, P., Kuipers, F.: Concepts of Exact QoS Routing Algorithms. IEEE/ACM Transactions on Networking **12** (2004) 851–864
3. Guerin, R., Williams, D., Orda, A.: QoS routing mechanisms and OSPF extensions. In: Proc. Globecom. (1997)
4. Kar, K., Kodialam, M., Lakshman, T.: Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications. IEEE Journal on Selected Areas in Communications **18** (2000) 2566–2579
5. Ahuja, R., Magnanti, T., Orlin, J.: Network Flows: Theory, Algorithms and Applications. Englewood Cliffs, NJ: Prentice-Hall (1993)
6. Wang, B., Su, X., Chen, C.: A New Bandwidth Guaranteed Routing Algorithm for MPLS Traffic Engineering. In: Proc. of IEEE International Conference on Communications, ICC 2002. (2002)
7. Banerjee, G., Sidhu, D.: Comparative analysis of path computation techniques for MPLS traffic engineering. Computer Networks **40** (2002) 149–165
8. De Neve, H., Van Mieghem, P.: TAMCRA: A Tunable Accuracy Multiple Constraints Routing Algorithm. Computer Communications **23** (2000) 667–679
9. Iliadis, I., Bauer, D.: A New Class of Online Minimum-Interference Routing Algorithms. Networking 2002, LNCS 2345 (2002) 959–971
10. Medina, A., Matta, I., Byers, J.: On the Origin of Power Laws in Internet Topology. ACM Computer Communications Review **30** (2000)