Optimal Joint Path Computation and Rate Allocation for Real-time Traffic

Antonio Frangioni * Laura Galli * Giovanni Stea †

Abstract

Computing network paths under worst-case delay constraints has been the subject of abundant literature in the past two decades. Assuming Weighted Fair Queueing scheduling at the nodes, this translates to computing paths and reserving rates at each link. The problem is $\mathcal{NP}$-hard in general, even for a single path; hence polynomial-time heuristics have been proposed in the past, that either assume equal rates at each node, or compute the path heuristically and then allocate the rates optimally on the given path. In this paper we show that the above heuristics, albeit finding optimal solutions quite often, can lead to failing of paths at very low loads, and that this could be avoided by solving the problem, i.e., path computation and rate allocation, *jointly* at *optimality*. This is possible by modeling the problem as a mixed-integer second-order cone program and solving it optimally in split-second times for relatively large networks on commodity hardware; this approach can also be easily turned into a heuristic one, trading a negligible increase in blocking probability for one order of magnitude of computation time. Extensive simulations show that these methods are feasible in today's ISPs networks and they significantly outperform the existing schemes in terms of blocking probability.

1 Introduction

Real-time traffic is nowadays a relevant component of IP-based networks. This definition encompasses a wide range of diverse applications, for which a bound on the end-to-end transit delay—including queueing, transmission and propagation delays—is mandatory for correct operation: for instance, multimedia applications such as live video – which are already widespread – or, again, applications such as remote sensing and control, surveillance systems, factory automation, stock exchange transactions etc., which are receiving increasing attention in the networking community. In the near future, it

*Dipartimento di Informatica, Università di Pisa, Largo B. Pontecorvo 3, 56127 Pisa, Italy. E-mail: {frangio,galli}@di.unipi.it

†Dipartimento di Ingegneria dell'Informazione, Università di Pisa, Largo Lucio Lazzarino 1, 56122 Pisa, Italy. E-mail: g.stea@iet.unipi.it

is foreseeable that the upcoming machine-to-machine (M2M) paradigm will favor the emergence of new real-time applications.

Supporting delay-constrained traffic requires the ability to compute paths where enough resources are available, on one hand, and to reserve these resources, on the other. The two above problems have been widely researched in the last two decades, both jointly and in isolation, under the name of QoS routing (see, e.g., [1–10]) and QoS partitioning (see, e.g., [5, 11–18]). The first problem consists in finding paths, the second one in computing the resources to be allocated on a given path. Some works (e.g., [7–9]), consider that, if Weighted Fair Queueing (WFQ) schedulers [19] are used at each node and traffic flows are leaky-bucket-shaped, then a bound on the end-to-end delay is a non-additive function of the *rate* allocated at each link. Thus, the problem of QoS routing with end-to-end delay bounds translates into finding a path where large enough rates are available. It has been shown in [7,8] that finding such a path is a polynomial problem, even when the rate to be allocated is not known in advance, assuming that a flow is allocated the *same* rate at all nodes, which is called *Equal Rate Allocation* (ERA). More recent works on QoS partitioning (e.g., [15–17]), however, show that ERA may fail to compute a path when a feasible one would exist if *unequal* rate allocation was instead allowed.

To the best of our knowledge, despite the abundance of literature, few works so far have tackled the problem of *joint* path computation and rate allocation with unequal rates at each link, constrained by a (non-additive) maximum end-to-end delay objective. This problem, which we call Single-Flow Single-Path Delay-Constrained Routing problem (SFSP-DCR) is $\mathcal{NP}$-hard, since it generalizes the Constrained Shortest Path problem (CSP); this justifies why most authors might have deemed it impossible to solve it to optimality in the context of real-time traffic allocation. However, just because a problem is $\mathcal{NP}$-hard this does not necessarily imply that it is not solvable for practical dimensions, i.e., comparable to those of today—and tomorrow—network domains. In fact, while complex path computation was clearly not practicable when individual routers were supposed to run it, recent developments in networking architectures make it a viable option nowadays: for instance, Path Computation Element (PCE, citePCE13), standardized by the IETF, are in fact dedicated servers for domain-wide path computation. Moreover, several QoS-oriented architectures (e.g., [21]) envisage per-domain resource managers, capable of complex computations. Last, but not least, Software Defined Networks [22] centralize network intelligence, therein including path computation capabilities. The problem of path computation and resource allocation is largely independent of what a *flow* is defined to be in the network architecture being considered. In some of these, a flow (or *microflow*) is the stream of packets sent by a single application: this is true of the IP IntServ architecture [23], which has well-known scalability problems due to the high number of flow states to be maintained

at the routers, as well as of other networks. In other architectures, a flow (or *trunk*, or *aggregate*) is a (possibly large) set of microflows being conveyed from one end to the other of a domain and requiring homogeneous forwarding behavior. This happens, for instance, in the IP DiffServ architecture [24], in MPLS [25], as well as in other IP-based networks, such as the Evolved Packet Core for cellular communications [26].

To the best of our knowledge, the above problem has been first tackled in our previous work [27]. In the latter, it has been shown that SFSP-DCR can be formulated as a convex Mixed-Integer Non-Linear Program (MINLP), and in particular as a Mixed-Integer Second-Order Cone Program (MISOCP) that can be efficiently solved by general-purpose tools once appropriate re-formulation techniques are used to construct "tight" models (i.e., with a relatively low integrality gap) that are also compact (in terms of both variables and constraints). Also, a number of approximation schemes are proposed for variants of the problems (e.g., when capacity reservation costs are not identical), along with a simple heuristic which tries ERA first and then falls back on solving the SFSP-DCR optimally. A preliminary evaluation of the latter, in a somewhat abstract setting, shows that it obtains extremely small average gaps in significantly smaller average running time than the exact approach.

In this work we first extend the results of [27] to a number of issues that were not addressed therein, and then we assess the actual impact of the proposed approach on the network performance. By means of an extensive simulation over several real-world IP networks with realistic demands and capacities, we show that computing paths and rates by solving the SFSP-DCR problem optimally largely outperforms the existing QoS routing and resource partitioning schemes in terms of blocking probability. More specifically, we compare our scheme against one where ERA is imposed, as in [7,8], and against one where shortest-widest and widest-shortest path computation is used in conjunction with the optimal rate allocation advocated in [16,17]. We show that both these approaches perform surprisingly worse than the optimal one, failing to compute feasible paths when these do exist in a significant percentage of cases, and therefore resulting in remarkably higher blocking probabilities even at very low network loads; close examination of the results show that this is indeed a fundamental issue of the ERA method when the network has links of significantly different (residual) capacity, something which happens in practice. Moreover, the computational cost of the exact approach is affordable: thanks to the effective compact formulation, optimally solving realistic-sized instances takes a time compatible with online computations in an operating environment, i.e., less than one second for a 50-node network on off-the-shelf hardware, which implies that the approach can actually be implemented in real-life equipment. In order to explore the trade-offs between performance and computational cost, we also test the afore-mentioned heuristic approach, showing that it trades

a negligible increase in the blocking probability for an order of magnitude of computation time, and thereby making the (almost) exact solution of SFSP-DCR even more feasible in a realistic setting.

The rest of the paper is organized as follows: Section 2 reviews the related work. Section 3 introduces the system model, and Section 4 states the problem formally and outlines our solutions. These are evaluated numerically in Section 5. Finally, Section 6 reports conclusions and highlights directions for future work on this topic.

2 Related Work

The problems of QoS-oriented path computation and resource allocation, considered either jointly or separately, have attracted a considerable amount of research since the mid '90s. As far as path computation is concerned, a relevant amount of literature has been published under the name of *QoS routing*. The key problem in this stream of literature is how to compute a path subject to multiple constraints (e.g., on the delay, jitter, bandwidth, loss probability, number of hops, etc.), while minimizing some per-path metric (e.g., the number of hops, the delay, etc.). Paper [1] sets the theoretical foundations for such multiconstrained QoS routing problem. The problem is to find a path for one flow between a given source-destination pair, minimizing the hop count and satisfying path constraints which can be categorized into three types, based on their composition function: additive (e.g., the average/maximum delay and jitter), multiplicative (e.g., the packet loss), or concave (e.g., a minimum bandwidth requirement). The authors prove that the above problem is $\mathcal{NP}$-complete if two or more additive/multiplicative constraints are to be satisfied. Conversely, they show that it is possible to find a path constrained by one concave metric and one additive/multiplicative metric in polynomial time, and they suggest that propagation delay (additive) and minimum reserved bandwidth (concave) be actually used in QoS routing schemes.

Starting from this work, several works have addressed heuristics that approximate the multiconstrained problem in one way or another (e.g., [2,3,28,29]). Most papers (e.g., [4,5,10,30–32]), assume that delays are static and/or additive per-link metrics. This assumption does not consider that queueing, which is not static, is a relevant delay component. Other papers tackle the problem from a probabilistic point of view, assuming a stochastic characterization of traffic and attempting to minimize or bound the average delay, which is hardly relevant at all for real-time traffic. Others, finally, take a pre-computation perspective, i.e., assume that some pre-computation work is done offline (taking whatever time it takes to do so), and then select among the pre-computed paths when online demands are made [6,33]. Fewer works (e.g., citeMaSt97,Orda99,pornavalai1997qos) propose path compu-

tation techniques constrained by deterministic (non-additive) delay bound constraints, also taking resource allocation on a path into account. The general reasoning is that, if WFQ schedulers [19] are used at each node and the requesting flows are leaky-bucket-shaped, then rate reservation translates to i) an end-to-end delay bound, ii) an upper bound on the end-to-end jitter, and iii) an upper bound on the buffer required at each node, once a path is selected. Therefore, a delay-constrained path is just one with enough rate to allow that maximum delay not to be exceeded. [9] finds the shortest path constrained by jitter, delay and buffer bounds under the assumption that the rate to be reserved is known in advance, using a polynomial algorithm. [7] shows that the problem is still polynomial if the rate to be reserved at each node is a variable, although it has to be the same at each node (ERA). [8] proposes lower-complexity approximate solutions to the problem solved exactly in [7]. Furthermore, it presents polynomial algorithms that find the *optimum* path when the cost is either a non-decreasing function of the allocated rate, path length and delay, or the residual rate of the bottleneck link along a path. Work [17] computes asymptotic bounds when rate-based joint routing and resource allocation has to be performed in a multi-class networks, according to fairness requirements.

Over the past 20 years, the literature on packet scheduling has witnessed a plethora of algorithms that *approximate* WFQ, normally at a lower complexity. The last in this line is Quick Fair Queueing (QFQ), [34]. All these algorithms sort packets in a *different* way with respect to WFQ, hence they exhibit a different latency expression. This, in turn, means that the relationship between the end-to-end delay bound and the reserved rate is different.

As far as *QoS partitioning* is concerned, several works exist that achieve optimal partitions for *additive* delays on a given path (see, e.g., [5, 13]). Under this hypothesis, [11] formulates and solves (heuristically) the problem of optimal routing and QoS partitioning given a number of pre-selected routes for each (source, destination) pair, whereas [12] computes the optimal delay partitioning and routing. Considerably fewer works assume instead non-additive end-to-end delay bounds constraints: [16, 17] show that reserving the same rate (as done in [7–9]) may be suboptimal and leads to failing of paths which might otherwise be admissible. The authors then propose an algorithm that allows a minimum-cost delay-feasible resource allocation to be computed *on a pre-selected path*, if such an allocation exists, assuming that costs are equal on each link. Their scheme consists in repeatedly trying ERA at the minimum rate that guarantees the requested delay. If the rate thus computed is available at all links, then the allocation is the minimum cost one; otherwise, the bottleneck links are allocated all the available rate, their delay contribution is discounted from the delay formula, and another iteration is performed to set the rates at the remaining links. Then, either the algorithm converges to a feasible solution (which is also optimal, in this case), or there is no feasible rate allocation along

the path. This leads the authors to conclude that there should be little benefits to be reaped by using non-uniform rate allocation; however, this intuition has been challenged by other results. For instance, [15] proposes a global (i.e., network-wide, or multi-flow) resource allocation scheme with delay bound constraints, assuming that paths have been selected, and exploiting optimization techniques to minimize the overall rate reserved in the network domain: rates are not assumed to be equal at each node for the same flow. The paper shows that the problem can be solved optimally for several classes of schedulers (including WFQ), and it may or may not have a convex formulation depending on the scheduler class; the results show instead that unequal rate allocation is necessary, since equal rate allocation may fail at very low network loads. Paper [35] proposes several criteria for resource allocation in WFQ-based networks (equal, capacity-proportional, and remaining capacity proportional) and shows how they perform in conjunction with least-loaded-path-first routing. The result is that ERA reduces the call blocking probability. A similar analysis is done in [18], with respect to Earliest-Deadline-First (EDF) scheduling: several delay budget partitioning schemes are proposed and evaluated in conjunction with various (delay-agnostic) path computation techniques, such as widest- or shortest-path-first. [14] discusses the advantages of optimal QoS partitioning (against a baseline of equal partitioning) using average end-to-end delay as a metric, and assuming Markovian models. Paper [13] formulates the optimum allocation problem on a given path with non-equal rates as a convex optimization problem, given an utility function of the nodes' rates.

To the best of our knowledge, our previous work [27] is the only one to tackle *joint* path computation and rate allocation with unequal rates. We showed therein that, under a number of conditions, the problem can be formulated in such a way as to be solvable by general-purpose tools in a time compatible with real-world network operations. However, no evaluation on the impact of this approach on the observable network performance (such as blocking probability) was performed.

3 System model

We consider a computer network represented by a directed graph $G = (N, A)$, where N is the set of nodes, corresponding to the routers in the network, and A is the set of directed arcs (ordered pairs of elements of N), corresponding to the links in the network; although links are typically bi-directional, flow in the two directions is not supposed to interfere, and therefore a link between a router $i \in N$ and a router $j \in N$ is appropriately represented by the two "opposite" directed arcs $(i, j) \in A$ and $(j, i) \in A$. We will denote by n the number of routers (the cardinality of N) and by m the number of arcs (the cardinality of A), i.e., twice the number

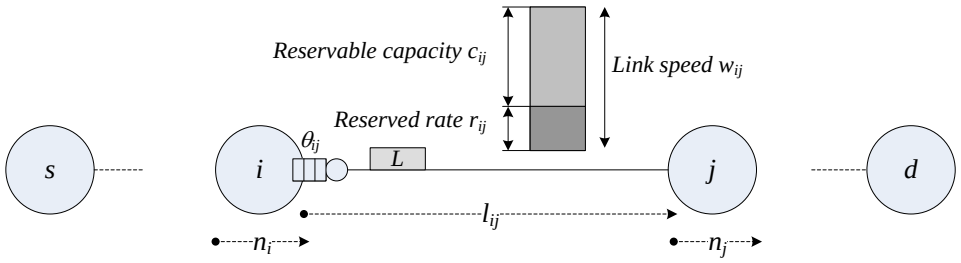

Figure 1: Some quantities in a path

of links in most cases. Our problem is to route one single "new" unicast flow through the network along a minimum cost path, where the cost is any linear function of the reserved capacities on the traversed arcs, with a constraint on the maximum delay that any packet may incur during the traversal. Note that the definition of *flow* is general enough to encompass the activity arising from a single application (sometimes called *microflow*) and a traffic trunk to be routed between the edges of an ISP (sometimes called *aggregate* or *macroflow*). With reference to Figure 1, we denote with $s \in N$ and $d \in N \setminus \{s\}$ respectively the source and destination of the flow. Following previous work on the topic (e.g., [7–9]), we assume that a flow's traffic is regulated by a *leaky-bucket traffic shaper*, whose burst is denoted by σ and whose rate is denoted by ρ. The latter ensures that, in any time interval of duration t, at most $\alpha(t) = \sigma + \rho \cdot t$ bits may enter the network.

Each link (arc) $(i, j) \in A$ in the network is characterized by a fixed *link delay* l_{ij}, a *physical link speed* w_{ij}, and a *reservable capacity* c_{ij} ($\leq w_{ij}$), which is to be shared among the flows traversing that node. The reservable capacity is dynamically updated at the setup and teardown of a flow. Each node $i \in N$ in the network is characterized by a maximum *node transit delay* n_i; moreover, the *maximum transmission unit* L (i.e., the maximal size of any packet) is known and assumed to be constant in the whole network. The flow has a *deadline* δ, after which packets are considered to be useless; in other words, the *worst-case delay* of the flow must be at most δ. We assume that each link $(i, j) \in A$ is managed by a WFQ scheduler [19], hence flows are reserved a *rate* r_{ij} at each link, $0 \leq r_{ij} \leq c_{ij}$ ($\leq w_{ij}$). Given a *routing* for the flow, i.e., the selected s- d path P in G, a *finite* delay bound can only be guaranteed if the minimum reserved rate at each link of the path is at least as large as the flow's rate ρ, i.e.,

$$r_{ij} \geq \rho \qquad \forall (i, j) \in P \tag{1}$$

Once (1) is satisfied, the maximum delay for the flow on path P is

$$D = \frac{\sigma}{\min\{\, r_{ij} \,:\, (i, j) \in P\}} + \sum_{(i,j) \in P} \left(\theta_{ij} + \frac{L}{w_{ij}} + l_{ij} + n_i \right) \tag{2}$$

where θ_{ij} is the node *latency*, i.e., the scheduling delay at the link (i, j). For WFQ schedulers, the latency is strictly rate-proportional, i.e.:

$$\theta_{ij} = \frac{L}{r_{ij}} \tag{3}$$

Note that (3) is a convex function of r_{ij} when $r_{ij} \geq 0$, something that will be exploited in order to devise an efficient solution[1]. Note that it is not possible to determine *a priori* which addendum in (2) is the dominant one. This depends on the burstiness of the flow σ, which is known beforehand, but also on the length of the path, which is an outcome of the computations. In particular, the second addendum increases with the path length, and may actually be the dominant term if long paths (or paths with long delay propagations) are selected.

We assume that link buffers are large enough to prevent overflow. The backlog bound at each node can be easily computed using formulas similar to those for the delay bound, hence this condition can be easily checked *a posteriori*.

4 Mathematical programming formulation

Given a flow with a burst σ and a rate ρ, which requests an end-to-end delay no larger than δ along a path from s to d, the SFSP-DCR problem requires to find one path from s to d, and a feasible rate reservation on each of its links so that the flow can meet its deadline at the minimum *cost*. For now, we assume that the cost is a linear function of the rates being reserved at each link; we will explore different objectives later on in this section. We now formulate the SFSP-DCR problem as an optimization problem, starting with its constraints.

We use binary variables $x_{ij} \in \{0, 1\}$ to indicate whether link (i, j) belongs to P: this allows us to write down *flow conservation constraints*:

$$\sum_{(j,i)\in BS(i)} x_{ji} - \sum_{(i,j)\in FS(i)} x_{ij} = \begin{cases} -1 & \text{if } i = s \\ 1 & \text{if } i = d \\ 0 & \text{otherwise} \end{cases} \qquad i \in N \tag{5}$$

The above ensure that – at optimality – the x variables represent an s-d path, if the (indirect) cost of setting any $x_{ij} = 1$ is positive. Here, $BS(i)$ is the subset of A containing the arcs entering node i (the so-called "backward

[1]This is not true of most WFQ approximations. For instance, QFQ's latency (called "Time Fairness Index" in [34]) is given by the following expression:

$$\theta_{ij} = 3 \cdot \frac{2^{\lceil \log_2 w_{i,j} \cdot L/r_{i,j} \rceil}}{w_{ij}} + \frac{L}{w_{ij}}, \tag{4}$$

which is a discontinuous function of the reserved rates.

star" of the node), while $FS(i)$ is the subset of A containing the arcs leaving node i (the so-called "forward star" of the node). Rate reservation variables r_{ij} are instead continuous. We introduce an additional variable r_{min}, with obvious meaning, and the corresponding constraints:

$$0 \leq r_{ij} \leq c_{ij} x_{ij} \qquad (i,j) \in A \tag{6}$$

$$\rho \leq r_{min} \leq r_{ij} + c_{max}(1 - x_{ij}) \qquad (i,j) \in A \tag{7}$$

These ensure – on one hand – that $r_{ij} = 0$ if $x_{ij} = 0$, and – on the other – that $\rho \leq r_{min} \leq r_{ij}$ if $x_{ij} = 1$, so that (1) holds. Note that $c_{max} = \max\{\, c_{ij} \,:\, (i,j) \in A \,\}$ is used in (7) to ensure that any link not in the chosen path ($x_{ij} = 0$) does not contribute to bounding r_{min} from above.

The constraint $D \leq \delta$, with D being given by (2), can be modeled using an auxiliary variable t and a rotated SOCP constraint as follows:

$$t + \sum_{(i,j) \in A} \left(\theta_{ij} + \left(\frac{L}{w_{ij}} + l_{ij} + n_i \right) x_{ij} \right) \leq \delta \tag{8}$$

$$t\, r_{min} \geq \sigma \qquad , \qquad t \geq 0 \tag{9}$$

Note that the $l_{i,j}$ and n_i terms in the sum in (8) are only counted in if $x_{i,j} = 1$, i.e., if the link and node are actually in P. For the same reason, some care must be taken to constrain the latency variable θ_{ij} to be equal to zero if $x_{ij} = 0$, or to an appropriate (convex) nonlinear expression otherwise. This is a *disjunctive set*, being expressed by a disjunction, which is in general *nonconvex*. Different formulations can be used to represent a disjunctive set in a mathematical program; the most widely used are *big-M* ones [36], that however are also well known to result in generally weak lower bounds and to be prone to numerical instability. One way to construct formulations with stronger continuous relaxation is to compute the *convex envelope* of the significant (nonlinear) fragments of the formulation, i.e., the best possible convex function agreeing with the nonconvex one at the points of its (disconnected) domain. Luckily, this can be done for the problem at hand using results originally due to [37, 38] and first put to actual computational use in [48] under the form of the *perspective reformulation*, that has been employed in several applications with success (e.g. [39–43]). Building on that theory, we propose the following formulation for imposing the required constraints on θ_{ij}:

$$\begin{aligned}
&\rho x_{ij} \leq r_{ij} \leq c_{ij} x_{ij} && (i,j) \in A \\
&0 \leq \theta_{ij} \leq (L/\rho) x_{ij} && (i,j) \in A \\
&\frac{L x_{ij}^2}{r_{ij}} \leq \theta_{ij} && (i,j) \in A
\end{aligned} \tag{10}$$

The crucial observation is that (10) can be directly modeled as a rotated SOCP constraint. Finally, we assume a linear objective function:

$$\sum_{(i,j)\in A} f_{ij} r_{ij}, \tag{11}$$

i.e,. minimizing the weighted amount of allocated rate along the path (where $f_{i,j}$ are non-negative constants). The whole model is thus a Mixed-Integer Second-Order Cone Program (MISOCP). Although MISOCPs are non-linear, their continuous relaxation possesses an algebraic structure that allows one to establish a strong duality theory; this in turn paves the way to efficient algorithms (e.g., interior-point methods are able to solve SOCPs in polynomial time), which means that MISOCPs can be solved (although in general *not* in polynomial time) by off-the-shelf, efficient, general-purpose solvers like `Cplex` or `GUROBI`. In [27], the proposed formulation is also compared with one based on standard big-M constraints: it is shown therein that the former is preferable, because its model size is smaller and its continuous relaxation much stronger, resulting in substantially shorter computation times. We will show in Section 5 that these times are indeed compatible with a dynamic network environment, even at fairly large scales and loads.

4.1 Generalizations of the model

We now show that similar MISOCP models can also be derived under more general hypotheses, regarding the objective function and the traffic arrival profile.

Regarding the objective, the weighted-sum objective function (11) can be expanded to include *per-link* (besides per-rate-unit) costs, while still remaining linear, i.e.:

$$\sum_{(i,j)\in A} f_{ij} r_{ij} + F_{ij} x_{ij}$$

Associating fixed costs F_{ij} to links may have a number of uses, e.g., classifying them based on reliability (a higher cost being associated to a less reliable link), breaking ties by favoring shortest paths, etc. Both per-rate-unit and per-link costs can be arbitrarily assigned, as long as they do not depend on the problem variables: for instance, they can be constant, or depend on topological properties, or on the link load (e.g., proportional to the available link capacity). The model even maintains the same structure if the objective function is changed to a min-max (or max-min), i.e., minimizing the maximum allocated rate at a link (or maximizing the minimum remaining rate at a link). All it takes, in fact, is to substitute (11) with:

$$\min \psi \qquad \text{with the additional constraints} \qquad \psi \geq f_{ij} r_{ij} + F_{ij} x_{ij} \quad (i,j) \in A$$

and the same goes for the max-min case *mutatis mutandis*.

Regarding the traffic arrival profile, some of the literature on QoS-routing considers flows shaped by *dual leaky buckets*: one (σ, ρ) for the *average* rate,

and another (L, p) for the *peak* rate, with $\sigma > L$ and $p > \rho$. Injected traffic must conform to both profiles simultaneously, and the delay bound formula changes to:

$$D = \frac{L}{r_{min}} + \max\left\{ 0\,,\ \frac{(\sigma - L)(p - r_{min})}{r_{min}(p - \rho)} \right\} + \sum_{(i,j)\in P} (\theta_{ij} + l_{ij} + n_i) \quad (12)$$

Fortunately, (12) can also be reformulated as a fragment of MISOCP. This is based on the fact that the max term in (12) only depends on whether $p < r_{min}$ or vice versa: in fact, since $\sigma > L$, $p > \rho$ and $r_{min} > 0$, the sign of the expression between curly brackets is the same as that of $p - r_{min}$. Hence, we can rewrite the first two addenda as:

$$\psi(r_{min}) = \begin{cases} \frac{L}{r_{min}} + \frac{K}{r_{min}} & \text{if } r_{min} \le p \\ \frac{L}{r_{min}} & \text{if } r_{min} \ge p \end{cases}$$

where K is a nonnegative constant. The above is a 2-piecewise function, with both pieces being individually convex. Thus the only place to focus our attention on is the point $p = r_{min}$ where ψ is nondifferentiable (but clearly continuous). However,

$$\psi'_-(r_{min}) = -\frac{L}{r_{min}^2} - \frac{K}{r_{min}^2} \le -\frac{L}{r_{min}^2} = \psi'_+(r_{min})$$

Hence, the left derivative in r_{min} is smaller than the right derivative, i.e., the derivative is globally non-decreasing, and therefore ψ is convex. Therefore, (12) too admits a convex (and, in particular, MISOCP) formulation, whose easy derivation need not to be explicitly reported here.

While the objective function determines the properties of the optima (hence the user-observed performance of the resulting QoS-routing scheme, e.g., how much traffic you can actually fit in your network), the choice of which objective is preferable may depend on several factors, including the network administrator policies, hence a performance comparison of various objective functions is outside the scope of this work. Furthermore, the QoS-routing scheme that we compare against, namely ERA [7, 8], accounts for single leaky-bucket shaping of traffic, and the optimal QoS-partitioning scheme that we compare against [16, 17] only considers minimizing the sum of the allocated rates (i.e., assumes $f_{ij} = 1$). Thus, to allow for a fair comparison, in the rest of the paper we also use a weighted-sum objective with unitary weights and a single-leaky-bucket delay formula. However, it is important to remark that a significant benefit of modeling the problem as a MISOCP is that the above extensions to more complex objective functions and different traffic arrival profile is quite easy, and they could be expected to have a minor impact on the overall performance.

5 Numerical Results

We now show that the optimal solution to the SFSP-DCR problem is both *affordable* and *effective* in realistic settings. To back up the first claim, we show that the solving time of the problem is normally below one second, on off-the-shelf hardware, even for large networks and for a wide range of loads. The second claim is supported by comparing our scheme with its direct competitors: on one hand ERA [7, 8], and on the other hand the optimal unequal resource allocation scheme proposed in [16,17] in conjunction with a shortest-widest-first or a widest-shortest-first path computation scheme. In particular we ran a large amount of simulation experiments on real-world IP networks, under varying network parameters, and we measured the *blocking probability*, i.e., the relative ratio of unfeasible path computations. We first describe the setup, and then report and comment our results.

5.1 Simulation setup

Constructing a set of meaningful DCR instances is a nontrivial exercise. We selected real-world IP network topologies taken from the Internet Topology Zoo [47]. Table 1 reports the network topologies we used; we exercised care to extract a set as heterogeneous as possible with respect to network dimension, connectedness (represented by the average node rank) and geographic span (summarized by the average per-link propagation delay), which ranges from regional (e.g., Belnet2009) to world-wide (e.g., DeutscheTelekom). Since the geographic coordinates of the nodes are provided, we set link delays l_{ij} as the ratio of the geodesic distance between i and j to the speed of light in a fiber. As far as node delays are concerned, [44] shows that they are typically small (20-40μs); since the above paper is already ten years old at the time of writing, we expect today's routers to be more performing, if possible, and less prone to delay spikes (which in [44] are attributed to IP option parsing, something that e.g., MPLS forwarding dispenses with), thus we fixed the node delay to 40μs, expecting this figure to err on the safe side. We also remark here that the Sago topology (shown in Figure 2) is an extended star with a three-link hub, hence having a single path between any couple of nodes, as this detail will be significant in the analysis (clearly, the path-finding part of the proposed approaches has no influence in this case).

Link capacities are assigned using the `FNSS` tool [45], which provides algorithms specifically designed for modeling link capacity assignment in ISP backbones. The one we chose exploits the *edge betweenness centrality* metric to select one among a finite set of link capacity values (in our case $\{1, 10, 40\}$ Gbps). `FNSS` also supports generation of realistic *traffic matrices* based on the network capacity. We use this feature to generate the ρ value of each request between two nodes in such a way that the request can be accepted in

Table 1: Topologies used in the simulations (Topology Zoo dataset, [47]).

Topology	# nodes	# links	# flows	avg. node rank	avg. prop. delay (ms)
Abilene	11	28	110	2.55	5.03
AttMpls	25	112	600	4.48	4.54
Bellcanada	48	128	2256	2.67	2.83
Belnet2009	21	48	420	2.29	0.19
DeutscheTelekom	39	124	912	3.18	13.79
Geant2010	37	112	1332	3.03	3.93
Ibm	18	48	306	2.67	4.67
Iris	51	128	2550	2.51	0.27
Sago	18	34	306	1.89	0.36
Tw	76	230	4970	3.03	2.66

an unloaded network; this ensures a reasonable baseline to assess blocking probabilities. Traffic matrices are generated using a log-normal distribution with a mean rate equal to 0.8 Gbps and a variance of 0.05. The MTU L is fixed to 1500 bytes, while flow bursts σ are set to a variable number of MTUs in order to evaluate different operating conditions, as discussed below. Finally, to define flow deadlines δ, we calculate two extreme values: δ_{min}, corresponding to the one obtained by allocating the entire link capacity and then calculating the delay-shortest path given this fixed allocation, and δ_{max} corresponding to the delay bound obtained by allocating a rate equal to ρ on all the links of the shortest path. Clearly, delay requests smaller than δ_{min} cannot be met, whereas requests higher than δ_{max} are likely to make the delay constraint redundant. Therefore, δ is randomly chosen uniformly within the interval $[\,\delta_{min}\,,\,\delta_{min} + (\delta_{max} - \delta_{min})\beta\,]$ for a fixed parameter $\beta \in (0, 1)$; the smaller β, the more difficult meeting the delay constraint can be expected to be.

Path computation requests are generated at time intervals exponentially distributed, with a varying rate λ: each path lasts for an exponentially distributed time with a mean equal to 1s, hence λ represents the number of erlangs. The number of path computations requested is

$$\frac{10}{\lambda}\max\left\{ m\,,\,\frac{1}{P_{min}} \right\} \quad \text{where} \quad P_{min} = \begin{cases} 10^{-2} & \text{if } \lambda > 5 \\ 10^{-3} & \text{otherwise} \end{cases}$$

which yields enoug samples to estimate blocking probabilities correctly even at low values of λ. Each point in the blocking probability graphs is obtained as the average of five independent replicas. 95% confidence intervals are also reported.

In our experiments we compared five different schemes: ERA, *shortest-widest* path first with optimal unequal rate allocation (SWPF-URA), *widest-shortest* path first with optimal unequal rate allocation (WSPF-URA), our

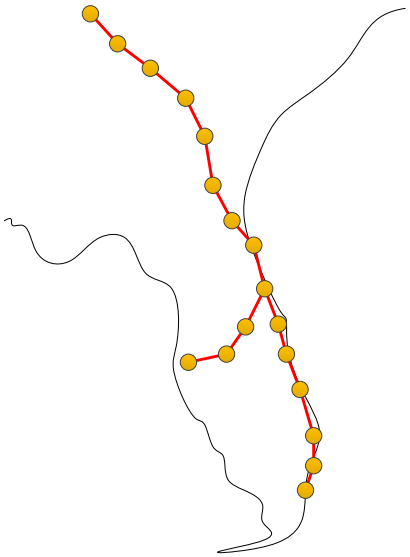

Figure 2: The Sago topology on an outline of south-east US

optimal MISOCP, and the *Three-Pronged Heuristic* (TPH) of [27]. The latter can be quickly described as follows. First, one checks the existence of at least one feasible path: this can be done in polynomial time (despite SFSP-DCR being $\mathcal{NP}$-hard) exploiting the fact that according to (2)–(3) the delay is a *decreasing* function of the rates, which means that setting $r_{ij} = c_{ij}$ for each arc (i, j) provides the best (least) possible contribution to the delay (cf. the computation of δ_{min} above). Clearly, if this computation fails, the problem is infeasible, and we terminate. If instead feasibility is ascertained, ERA (which still runs in polynomial time) is attempted: if ERA succeeds its solution is returned, otherwise (i.e., ERA does not find any path, although we are sure there actually is one) one falls back on solving the MISOCP optimally. TPH requires a much smaller computation time *on average*, since on one hand infeasible problems are quickly ruled out at a very small cost, and on the other hand the number of times that SFSP-DCR is solved optimally via MISOCP is significantly reduced. Simulations have been performed on a 2.299 Ghz AMD Opteron(tm) Processor 6376 with 16Gb RAM running a 64 bits Linux operating system (Ubuntu 12.4). All the codes were compiled with `gcc 4.4.3` and `-O3` optimizations. The MISOCP was solved by off-the-shelf commercial solver `Cplex 12.5`, run with default parameters. Hence, significant efficiency improvements w.r.t. our results can be obtained at little cost, e.g., by appropriately tweaking the algorithmic parameters of the solver. It should be noted that general-purpose solvers tend to become better and better with time due to the large amount of development effort

devoted to their improvement, so it can be expected that the solution speed of the MISOCP model will significantly improve over time, with no extra implementation effort, just on the account of the combined advances in the available hardware and software environments; this alone makes it attractive as a long-term option. Of course, development of specialized approaches better exploiting the structure of the problem is also possible and has the potential to further significantly improve the performance. The results of the experiments are reported and discussed in the following.

5.2 Results

We start by reporting the blocking probability in Figure 3, as a function of λ, for the various topologies when $\beta = 0.2$ and $\sigma = 3$ MTU. We repeated the simulations for different values of β and σ, obtaining qualitatively similar results, which are not reported for the sake of conciseness.

The figure clearly shows that MISOCP outperforms all the other schemes, although TPH performs quite near. The improvement is more evident at low loads, where the other schemes exhibit a surprisingly high (relatively speaking) blocking rate, and obviously reduces as the network grows overloaded [2]. As a general rule, MISOCP and TPH appear to perform better when the network is highly connected (see, e.g., the AttMpls, DeutscheTelekom, Geant2010 and Tw topologies, whose average node rank is above three); this could be expected, because the more connected a network is, the larger the solution space is in general. We remark, however, that such a reduction of the blocking probability was not obvious at all a priori: given that MISOCP computes the optimum and the competitors do not, it is straightforward that, *starting from the same network state*, the chance of MISOCP finding a feasible path are higher than its competitors'. However, nothing guarantees that this will still be true as the respective network states diverge, i.e. with MISOCP accepting paths that its competitors reject, hence occupying more resources; this is instead what happens in all the cases.

The fact that ERA and both the URA-based schemes (i.e., SWPF-URA and WSPF-URA) exhibit relatively high blocking probability even when the network is unloaded begs some explanation. When $\lambda = 0.1$, in fact, a back-of-the-envelope computation is enough to convince the reader that the probability that the network is empty at the time of a flow request is more than 90%[3]. For ERA, the high blocking probability is likely due to

[2]Note that, even in overload conditions, our schemes never fall behind the others, and they sometimes exhibit differences that are significant, though harder to observe in a logarithmic plot: for instance, in the AttMPLS plot with $\lambda = 100$, MISOCP blocks 45% of the flows, and TPH 52%, whereas ERA and SWPF-URA block 59% (i.e., 31% more than MISOCP), and WSPF-URA blocks 72% (i.e., 60% more).

[3]In fact, at least for sufficiently low blocking probabilities, the system can be modeled as an $M/M/\infty$ queue, hence the probability that an incoming request finds the system empty is $e^{-\lambda/\mu} = 0.904$.

Table 2: Additional data for two network topologies at $\lambda = 0.1$.

	Abilene	DeutscheTelekom
blocking probability	0.07%	0.24%
% of unequal alloc.	5.74%	23.13%
avg. Jain's fairness index	87.5%	82.4%
avg. path length	2.51	2.99
avg. path l. of SWPF	2.23	2.54

Table 3: Occurrence of URA at various loads

λ	Abilene	DeutscheTelekom
0.1	5.81%	23.13%
1	7.98%	23.08%
10	27.66%	37.56%
100	68.63%	72.90%

the fact that a path—with the *same* rate at all nodes—having the requested delay may not exist at all. This is confirmed by the data in Table 2, that complement some of those in Figure 3. For instance, in the DeutscheTelekom topology, ERA fails 23% of the times, whereas MISOCP (which instead exhibits a negligible blocking probability) allocates rates *unequally* 23.13% of the times. This number is almost equal to the blocking probability of ERA at the same load. The same happens with the Abilene topology. Note that, when MISOCP allocates unequal rates, these are generally quite different: this is testified by the relatively low value of Jain's fairness index in Table 2, computed only on the paths where rates are allocated unequally. Table 3 reports the occurrence of unequal rate allocations as a function of the load: as the load increases, the number of times where delay bounds are met by harvesting rate where it is available increases as well. As far as rate allocation is concerned, Figure 4 reports the cumulative distribution function of the ratio of the average reserved rate along the path to the flow's rate ρ, for all the algorithms being evaluated. The figure shows that, for MISCOP and TPH, a rate equal or very similar to ρ is allocated roughly 10% of the times, and the median reserved rate is five times the flow's rate. The correlation between the path length (which is between one and six hops in this case) and the overprovisioning is slightly positive, i.e. 0.2. In any case, all the algorithms tend to exhibit a similar overallocation behavior, with WSPF-URA overallocating more heavily, due to the fact that it favors longer paths, hence needs to compensate for larger link-dependent delays (i.e,. propagation and transmission). The fact that ERA appears to be more thrifty than MISOCP with rate allocation, albeit by a very low margin, should not overshadow the hundredfold difference in the blocking probability between the two.

The Sago topology deserves an ad hoc interpretation: since the edge links in the spokes of its extended star have the lowest capacity (1Gbps), any request having one edge node as an endpoint can only be accommodated with ERA if 1Gbps is a large enough rate to guarantee the requested delay (which is seldom the case). This shows that, unlike what stated in [16, 17], assuming equal rate allocation *does* impair effective resource allocation, even in this extreme case where path computation is instead unaltered. The Sago example provides the blueprints for understanding the high failure probability in other networks. Indeed, one can expect that most networks have a reasonably well-meshed core served by high-bandwidth links surrounded by peripheral lower-bandwidth links, as shown in Figure 5. It may then well happen that all possible paths between a given *s*-*d* pair (filled nodes in the figure) have a relatively long segment in the core part of the network, where high (residual) capacities are available (thick lines in the figure), but must traverse at least one low-capacity link (thin lines in the figure). As in the Sago example, the ERA assumption forces all the links in the path (dashed in the figure) to receive the same low rate dictated by the bottleneck low-capacity link, which may well result in the inability to satisfy the delay constraint; however, allocating substantially higher rates on the (many) high-capacity links may significantly reduce the delay experienced there, and therefore allow the flow to meet its deadline.

As far as URA-based schemes are concerned, the high blocking probability can be due to the fact that the shortest-widest (resp., widest-shortest) path may not be the optimal choice, i.e., computing path and resources disjointly is generally unadvisable. This is confirmed by the data in Table 2, where it is shown that MISOCP may compute longer paths than SWPF-URA (e.g., in the DeutscheTelekom topology). Note that in the Sago topology URA-based schemes perform exactly like MISOCP and TPH, since there is only one path between two endpoints, and unequal resource allocation is solved optimally. More in general, in roughly half the cases, URA-based schemes outperform ERA, often by a large margin.

Average computation times are shown in Figure 6. The figure shows that, in our implementation, ERA is by far the fastest scheme; its computation time increases with the load, showing that it becomes increasingly difficult to find a resource-constrained path when the network is saturated. The computation time for both URA-based schemes is around 10ms, and it is weakly dependent on the topology. This is due to the fact that SWPF and WSPF schemes do not allocate resources, and complete within short times over all the topologies we tested. On the other hand, the size of the SOCP solved for resource allocation depends on the *length* of the selected path (rather than on the size of the topology), and the latter is small in general, hence not overly different from one topology to the other. Finally, both schemes exhibit a decreasing trend with the load, which is due to the fact that the optimization problem becomes unfeasible at high loads,

and general-purpose solvers often detect unfeasible problems faster than they solve feasible ones of the same size; this is also confirmed by the fact that computation times are flatter in more connected networks. It should be remarked that our implementation of the URA-based schemes could be made faster by employing the technique of [16, 17] instead of a general-purpose solver to solve the resource-allocation problem for the fixed path; this might easily make them competitive with ERA efficiency-wise, although it would do nothing to ameliorate the effectiveness gap w.r.t. MISOCP. The exact approach is the most costly, which was expected; however, the order of magnitude of computation times is indeed affordable, being in the tens of milliseconds. Computation times do depend on the topology, which again could be expected, and specifically on both its connectedness and dimension (the former being more revealing of the size of the solution space than the latter); the decreasing trend with the load can be explained in the same terms as for URA-based schemes. Finally, TPH is indeed more efficient than MISOCP, achieving times often close —and sometimes inferior — to those of URA-based schemes while resulting in a comparatively much smaller blocking probability. The improvement brought about by TPH is due to the fact that, in most cases, the optimal solution is indeed one where all the rates are equal (see again Table 2), hence the ERA part *does* find it in very little time.

The above results seem to indicate that MISOCP (or TPH in its stead) can be successfully used when path computation needs to be done in real time, in highly dynamic environments. To further examine the feasibility of this approach from the efficiency standpoint, Figure 7 reports box plots of the computation times of MISOCP in two of the largest topologies (bottom/top whiskers are at the 5th and 95th percentile respectively); these results are typical for all the topologies, hence the other graphs need not be reported. The figures show that the distribution of the computation times is narrow, and that they depend weakly on the load, which indicates that MISOCP has a predictable performance.

In the same vein, Figures 8 and 9 examine the impact of the burst σ on both the blocking probability and the computation time (for the two largest topologies, namely Iris and Tw; however, once again these results are typical). In both figures we vary the burst in $\{1, 3, 10\}$ times the MTU, with $\beta = 0.2$; as the burst increases, the requested rate at each link increases as well, which makes path computation harder. The blocking probability is hardly at all influenced by the size of the burst, as shown in Figure 8, meaning that MISOCP finds a way to accommodate larger bursts in the network; this comes at the price of slightly more involved computations, as shown in Figure 9 (although in all cases the running time tends to decrease with the load. We performed analogous experiments varying the requested deadline, i.e., $\beta \in \{0.5, 1.0\}$: the effect is similar to varying the burst although in the

opposite direction, since a higher β implies a smaller allocated rate on average. The results are analogous but the differences are even less noticeable, since the deadline is extracted randomly within an interval, hence we omit the corresponding graphs.

6 Conclusions

In this paper we have shown that optimal joint path computation and rate allocation, under worst-case delay constraint, appears to be a promising approach. Indeed, while the problem is $\mathcal{NP}$-hard, it can be solved in split-second time for today's networks on off-the-shelf hardware and using general-purpose tools, provided that an appropriate mixed-integer second-order cone formulation is employed. The approach is much more effective, in terms of blocking probability, than alternative polynomial-time schemes relying on either equal rate allocation, or disjoint path computation and (possibly unequal) rate allocation, even in unloaded networks. This appears to be due to the following facts:

- Meeting stringent deadlines requires allocating large rates, possibly much larger than the flow's minimum requirement. When a path must traverse bottleneck links (e.g., network edges), equal rate allocation poses too high a demand on the bottlenecks, which makes the computation unfeasible, while our scheme can circumvent the problem by allocating higher rates in the core than at the edges.

- When path computation is resource-agnostic (e.g., shortest-widest-path-first), there is a high chance that the paths thus computed will be proved unfeasible when the problem of resource allocation on these paths is subsequently tackled.

Besides solving the problem optimally, we have shown that there is space for further exploring tradeoffs between computation times and path computation accuracy; for instance, a simple heuristic trades very little accuracy for almost an order of magnitude of computation time.

There are several directions for future work, some of which are actively being pursued at the time of writing. First, there is clearly further space for improvement in the efficiency of the SFSP-DCR solution. Indeed, while the tight MISOCP formulation already provides pretty reasonable solution times when general-purpose solvers are used, special-purpose solution techniques may significantly improve upon this; in that respect, Lagrangian-based solution algorithms appear to be a promising idea.

Second, having established that optimal joint path computation and resource allocation is a viable option, it would be interesting to examine whether and how different objective functions (e.g., weighted rate sums,

maximization of the min rate along a path, etc.) impact the effectiveness and efficiency of path computation.

Third, this work considers only WFQ scheduling at the nodes. Commercial routers also implement other, simpler scheduling algorithms, e.g. variants of Deficit Round Robin [46], or WFQ approximations, e.g. QFQ [34]. In the former, latency depends on several parameters, such as the number of flows traversing a link and their bandwidth, whereas in the latter the latency is higher than in WFQ and generally a non-convex function of the rate. Solving the SFSP-DCR problem with such schedulers is likely to be considerably more difficult, due to the different latency expression; furthermore, they may require more resources than WFQ to guarantee the same delay bound. It thus becomes interesting to assess what you pay for having simpler schedulers (e.g., higher path computation complexity, higher blocking probability, or both).

Finally, it would also be interesting to investigate the *multipath* extension of the present work; to the best of our knowledge, multipath delay-constrained routing is in fact, as of today, unexplored.

References

[1] Wang, Z. and Crowcroft, J. (1996) QoS routing for supporting resource reservation. *IEEE Journal on Selected Areas in Communications*, **14(7)**, 1228–1234.

[2] Korkmaz, T. and Krunz, M. (2001) Multi-constrained optimal path selection. *Proceedings of INFOCOM'01*, Anchorage, AK, 22-26 April, pp. 834–843. IEEE, Piscataway, NJ.

[3] Yuan, X. (2002) Heuristic Algorithms for Multiconstrained QoS routing. *IEEE/ACM Transactions on Networking*, **10(2)**, 244–256.

[4] Misra, S. and Xue, G. and Yang, D. (2009) Polynomial time approximations for multi-path routing with bandwidth and delay constraints. *Proceedings of INFOCOM'09*, Rio de Janeiro, BR, 19-25 April, pp. 558–566. IEEE, Piscataway, NJ.

[5] Lorenz, D. H. and Orda, A. (2002) Optimal partition of QoS requirements on unicast paths and multicast trees. *IEEE/ACM Transactions on Networking*, **10**, 102–114.

[6] Orda, A. and Sprinston, A. (2000) *QoS Routing: the Precomputation Perspective*, In *Proceedings of INFOCOM'00*, Tel Aviv, Israel, 26-30 March, pp. 128–136. IEEE, Piscataway, NJ.

[7] Ma, Q. and Steenkiste, P. (1997) Quality-of-Service Routing for Traffic with Performance Guarantees. *Proceedings of IWQOS'97*, New York City, NY, 21–23 May, pp. 115–126. Springer, Berlin.

[8] Orda, A. (1999) Routing with End-to-End QoS Guarantees in Broadband Networks. *IEEE/ACM Transactions on Networking*, **7(3)**, 365–374.

[9] Pornavalai, C. and Chakraborty, G. and Shiratori, N. (1997) QoS based routing algorithm in integrated services packet networks. *Proceedings of ICNP'97*, Atlanta, GA, USA, 28-31 October, pp. 167–174. IEEE Computer Society, Washington, DC, USA.

[10] Yu, Z. and Ma, F. and Liu, J. and Hu, B. and Zhang, Z. (2013) An Efficient Approximate Algorithm for Disjoint QoS Routing. *Hindawi Mathematical Problems in Engineering*, **2013**, 1–9.

[11] Atov, I. and Tran, H. T. and Harris, R. J. (2005) OPQR-G: algorithm for efficient QoS partition and routing in multiservice IP networks. *Computer Communications*, **28**, 1987–1996.

[12] Lorenz, D. H. and Orda, A. and Raz, D. and Shavitt, Y. (2006) Efficient QoS partition and routing of unicast paths and multicast. *IEEE/ACM Transactions on Networking*, **14**, 1336–1348.

[13] Saad, M. and Leon-Garcia, A. and Yu, W. (2007) Optimal Network Rate Allocation under End-to-End Quality-of-Service Requirements. *IEEE Transactions on Network and Service Management*, **4(3)**, 40–49.

[14] Cho, H. and Girard, A. and Rosenberg, C. (2007) On the advantages of optimal end-to-end QoS budget partitioning. *Telecommunications Systems*, **34**, 91–106.

[15] Lori, A. and Stea, G. and Vaglini, G. (2010) Towards Resource-Optimal Routing Plans for Real-Time Traffic. In Margaria, T. and Steffen, B. (eds), *Leveraging Applications of Formal Methods, Verification, and Validation*, vol. 6415 of *Lecture Notes in Computer Science*. Springer, Berlin.

[16] Diwan, A. and Kuri, J. and Sanyal, S. (2012) Optimal allocation of rates in guaranteed service networks. *Informatica*, **4**, 201–212.

[17] Diwan, A. and Kuri, J. and Kumar, A. (2002) Optimal per-Node Rate Allocation to provide per-Flow End-to-End Delay Guarantees in a Network of Routers supporting Guaranteed Service Class. *Proceedings of ICC'02*, New York City, NY, Apr. 28 - May 2, pp. 1112–1117. IEEE, Piscataway, NJ.

[18] Elsayed, K. M. F. (2005) A framework for end-to-end deterministic-delay service provisioning in multiservice packet networks. *IEEE Transactions on Multimedia*, **7**, 563–571.

[19] Parekh, A. K. and Gallager, R. G. (1993) A generalized processor sharing approach to flow control in integrated services networks: the single-node case. *IEEE/ACM Transactions on Networking*, **1**, 344–357.

[20] Paolucci, F. and Cugini, F. and Giorgetti, A. and Sambo, N. and Castoldi, P. (2013) A survey on the path computation element (PCE) architecture. *IEEE Communications Surveys and Tutorials*, **99**, 1–23.

[21] Mingozzi, E. et al. (2009) Euqos: End-to-end quality of service over heterogeneous networks. *Computer Communications*, **32(12)**, 1355–1370.

[22] Open Networking Foundation. (2013) Software-defined networking: The new norm for networks. *ONF White Paper*.

[23] IETF RFC 1633 (1994) Integrated Services in the Internet Architecture: an Overview. The Internet Society, Reston, VA.

[24] IETF RFC 2475 (1998) An Architecture for Differentiated Services. The Internet Society, Reston, VA.

[25] IETF RFC 3031 (2001) Multiprotocol Label Switching Architecture. The Internet Society, Reston, VA.

[26] 3GPP TS 23.401 General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access 3rd Generation Partnership Project (3GPP).

[27] Frangioni, A. and Galli, L. and Scutellà, M.G. (2013) Delay-Constrained Shortest Paths: Approximation Algorithms and Second-Order Cone Models. Technical Report 13-06, Dipartimento di Informatica, Università di Pisa.

[28] Juttner, A. and Szviatovszki, B. and Mecs, I. and Rajko, Z. (2001) Lagrange relaxation based method for the QoS routing problem. *Proceedings of INFOCOM'01*, Anchorage, AK, 22-26 April, pp. 859–868. IEEE, Piscataway, NJ.

[29] Kuipers, F. and Korkmaz, T. and Kruntz, M. (2002) An overview of constraint-based path selection algorithms for QoS routing. *IEEE Communications Magazine* **40(12)**, 50–55.

[30] Banerjee, G. and Sidhu, D. (2002) Comparative analysis of path computation techniques for MPLS traffic engineering. *Computer Networks*, **40**, 149–165.

[31] Hashimoto, M. and Oki, E. (2011) A simplified delay-guaranteed traffic engineering method. *IEEE Journal of Selected Areas in Telecommunications (JSAT)*, 1–8.

[32] Yang, W. L. (2004) Optimal and heuristic algorithms for quality-of-service routing with multiple constraints. *Performance Evaluation*, **57**, 261–278.

[33] Liu, B. L. and Yuan, M. and Chen, G. and Peng, J. (2013) One QoS Routing Algorithm for Load Balancing in IP Backbone Networks: Design and Simulation. *Applied Mechanics and Materials* , **427–429**, 2664–2667.

[34] Checconi, F. and Rizzo, L. and Valente, P. (2012) QFQ: Efficient Packet Scheduling with Tight Guarantees. *IEEE/ACM Transactions on Networking*, **21**, 802–816.

[35] Elsayed, K. M. F. and Saad, A. and El-Hadidi, M. T. (2002) Performance evaluation of resource reservation and call admission policies for deterministic services in PGPS-based packet networks. *Computer Communications*, **25**, 1513–1526.

[36] Nemhauser, G. L. and Wolsey, L. A. (1988) *Integer and Combinatorial Optimization*. Wiley-Interscience.

[37] Ceria, S. and Soares, J. (1999) Convex programming for disjunctive convex optimization. *Mathematical Programming*, **86**, 595–614.

[38] Tawarmalani, M. and Sahinidis, N. V. (2001) Semidefinite Relaxations of Fractional Programs via Novel Convexification Techniques. *Journal of Global Optimization*, **20**, 137–158.

[39] Frangioni, A. and Gentile, C. (2009) A Computational Comparison of Reformulations of the Perspective Relaxation: SOCP vs. Cutting Planes. *Operations Research Letters*, **37(3)**, 206–210.

[40] Frangioni, A. and Gentile, C. and Grande, E. and Pacifici, A. (2010) Projected Perspective Reformulations with Applications in Design Problems. *Operations Research*, **59(5)**, 1225–1232.

[41] Frangioni, A. and Gentile, C. and Lacalandra, F. (2009) Tighter Approximated MILP Formulations for Unit Commitment Problems. *IEEE Transactions on Power Systems*, **24(1)**, 105–113.

[42] Günlük, O. and Linderoth, J. (2012) Perspective Reformulation and Applications. In Leyffer, S. and Lee, J. (eds), *Mixed Integer Nonlinear Programming*, vol. 154 of *The IMA Volumes in Mathematics and its Applications*. Springer-Verlag, Berlin.

[43] Hijazi, H. and Bonami, P. and Cornuejols, G. and Ouorou, A. (2010) Mixed Integer NonLinear Programs featuring "On/Off" Constraints: Convex Analysis and Applications. *Electronic Notes in Discrete Mathematics*, **36(1)**, 1153–1160.

[44] Papagiannaki, K. and Moon, S. and Fraleigh, C. and Thiran, P. and Diot, C. (2003) Measurement and analysis of single-hop delay on an IP backbone network. *IEEE Journal on Selected Areas in Communications*, **21(6)**, 908–921.

[45] Saino, L. and Cocora, C. and Pavlou, C. (2013) A toolchain for simplifying network simulation setup. *Proceedings of SIMUTOOLS '13*, Cannes, France 06–08 March. ICST, Brussels, Belgium.

[46] Lenzini, L. and Mingozzi, E. and Stea, G. (2007) Performance analysis of modified deficit round robin schedulers. *IOS Journal of High Speed Networks*, **16(4)**, 399–422.

[47] The internet topology zoo. `http://www.topology-zoo.org/`.

[48] Frangioni, A. and Gentile, C. (2006) Perspective Cuts for a Class of Convex 0–1 Mixed Integer Programs. *Mathematical Programming*, **106(2)**, 225–236.

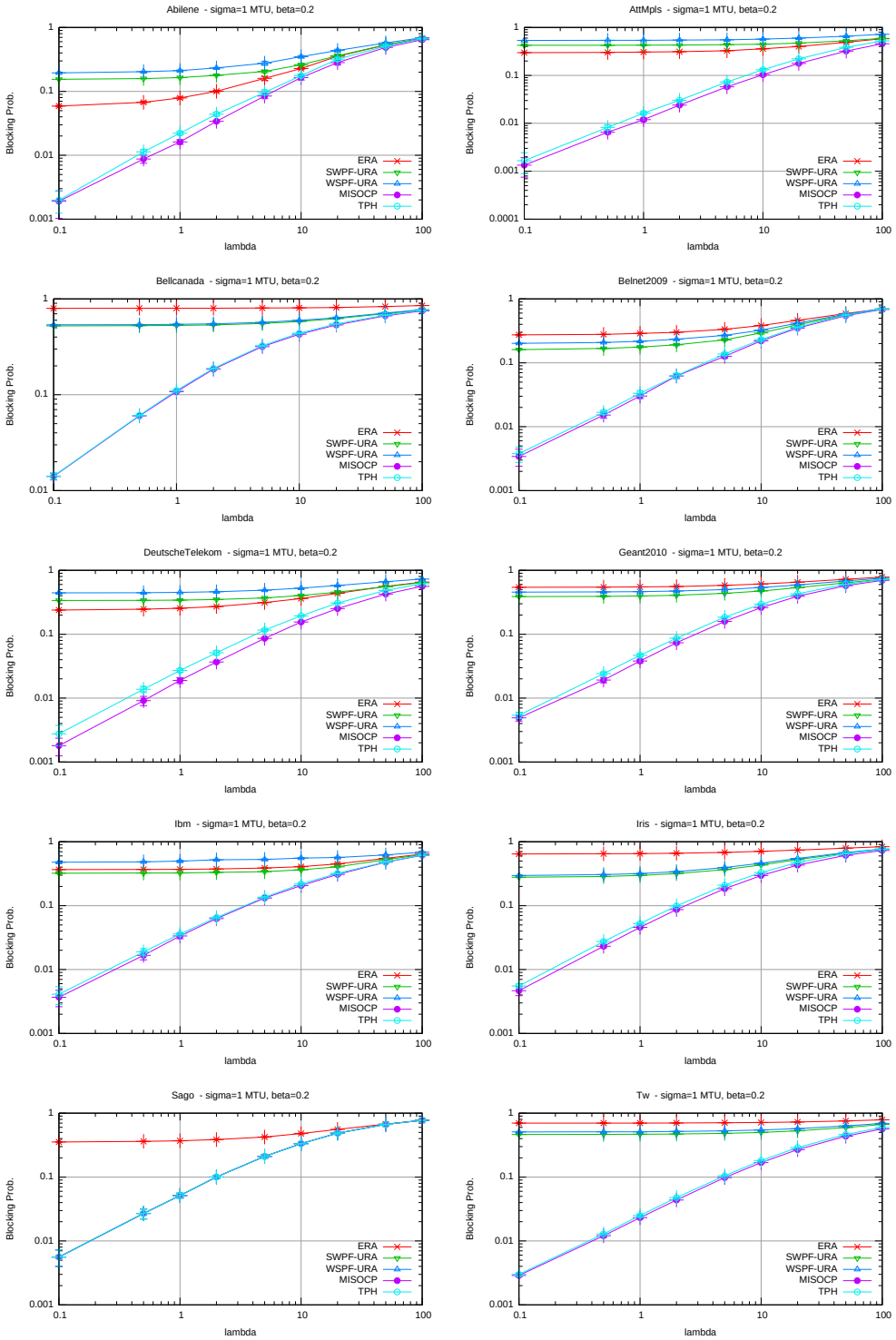

Figure 3: Blocking probability for all topologies, $\beta = 0.2$ and $\sigma = 3$ MTU

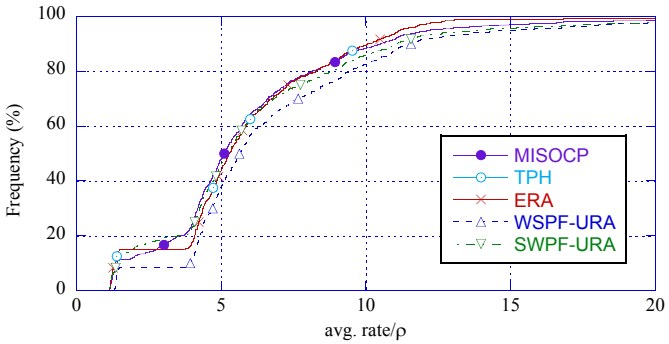

Figure 4: CDF of the overprovisioniong ratio in the DeutscheTelekom network, $\lambda = 0.1$.

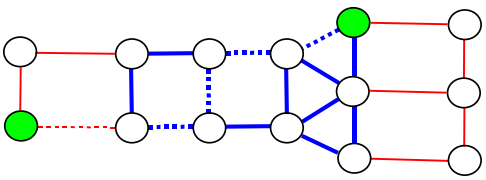

Figure 5: Pictorial representation of a prototypical failure case for ERA. The thickness of a link is indicative of its capacity.

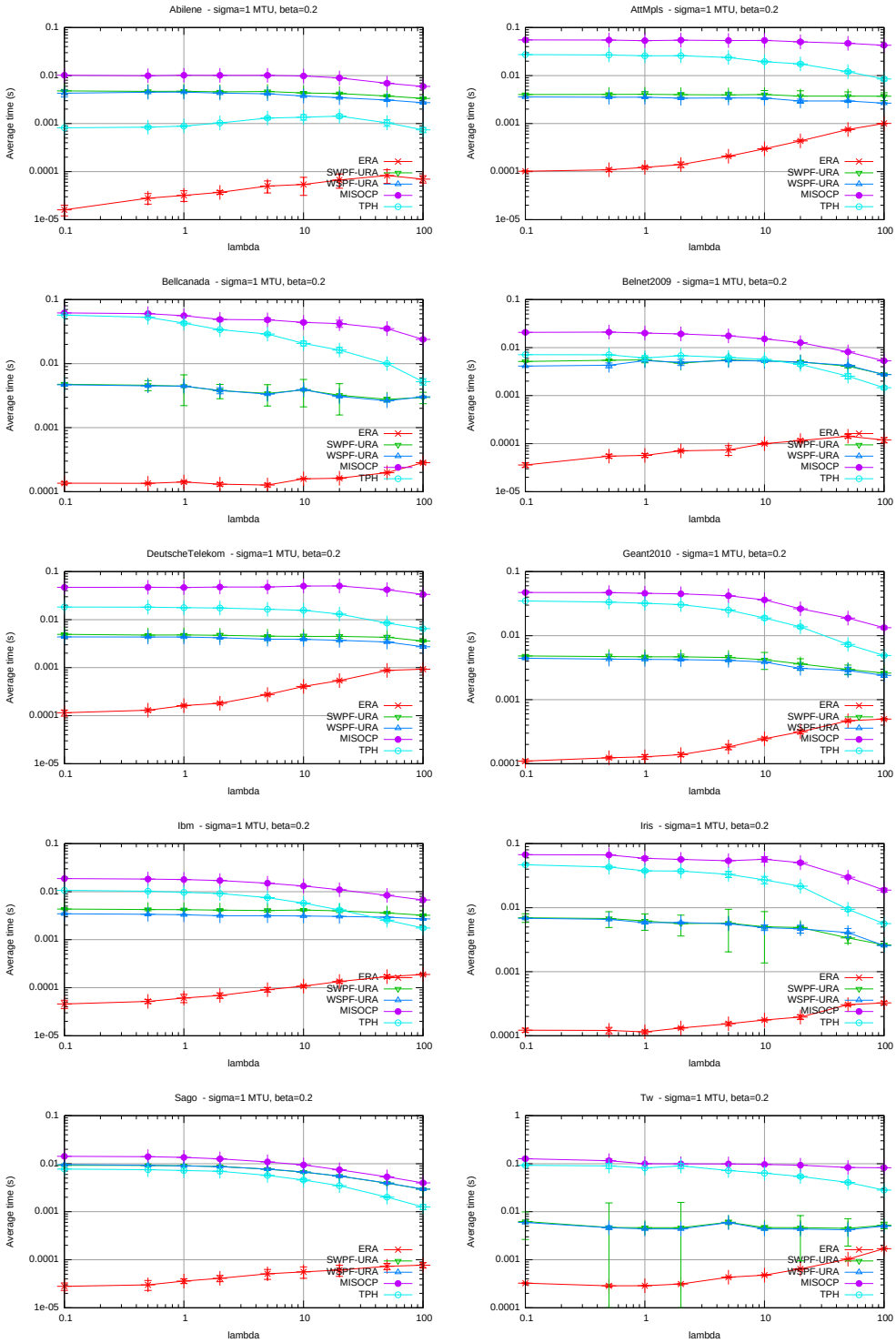

Figure 6: Average computation times

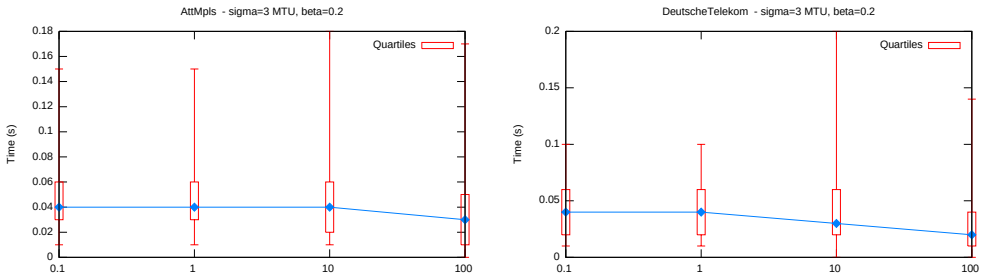

Figure 7: Boxplot of computation times of MISOCP, AttMpls (left) and DeutscheTelekom (right) networks

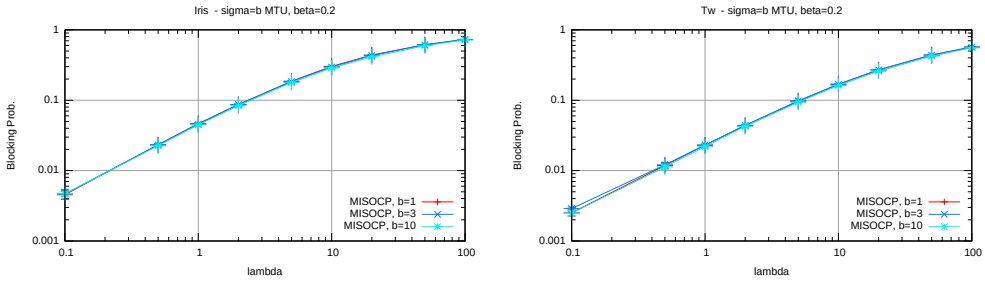

Figure 8: Blocking probability as a function of the burst, in the Iris (left) and Tw (right) networks.

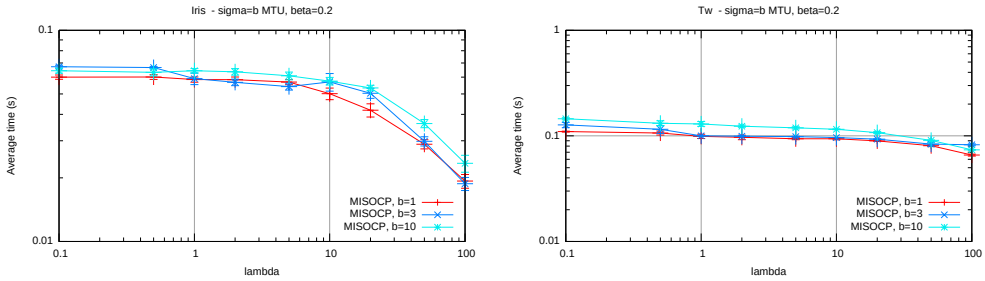

Figure 9: Average computation time as a function of the burst, in the Iris (left) and Tw (right) networks.