Università di Pisa

Dipartimento di Informatica

Technical Report

QoS Routing with worst-case delay constraints: models, algorithms and performance analysis

Antonio Frangioni Laura Galli Giovanni Stea

December 22, 2015

ADDRESS: Largo B. Pontecorvo 3, 56127 Pisa, Italy. TEL: +39 050 2212700 FAX: +39 050 2212726

QoS Routing with worst-case delay constraints: models, algorithms and performance analysis

Antonio Frangioni * Laura Galli* Giovanni Stea †

December 22, 2015

Abstract

In a network where weighted fair-queueing schedulers are used at each link, worst-case end-to-end delays can be inferred from per-link rate reservations. Therefore, it is also possible to compute resource-constrained paths that meet target delay constraints, *and* optimize some key performance metrics (e.g., minimize the overall reserved rate, maximize the remaining capacity at bottleneck links, etc.). Despite the large amount of literature on QoS routing appeared since the mid '90s, few papers so far have discussed solving such path computation problem at optimality in general settings. In this paper, we formulate and solve the *optimal path computation and resource allocation problem* assuming different types of weighted fair-queueing schedulers in the network. We show that, depending on the scheduling algorithm, routing a new flow may or may not affect the performance of existing flows; hence, explicit admission control constraints may be required to ensure that existing flows still meet their deadline afterwards. Yet, for the relevant schedulers proposed in the literature the problem can be formulated as a Mixed-Integer Second-Order Cone problem (MI-SOCP), and can be solved at optimality in split-second times even in fairly large networks.

Keywords: *QoS routing, worst-case delay, weighted fair-queueing, admission control, optimization*

1 Introduction

The research on Quality of Service (QoS) in the '90s has produced a vast number of *packet scheduling algorithms*, to be employed at network links to determine whose flow's head-of-line packet should be sent on the link when more than one flow is backlogged. Many of

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

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

these algorithms aim to approximate Generalized Processor Sharing (GPS, [26]), also called Weighted Fair Queueing (WFQ). The latter is an ideal paradigm, which emulates a reference system where backlogged flows are served simultaneously, and each one is given a share of the link proportional to its *weight*. If the weights are chosen as the flows' requested rates, then each flow always gets no less than its requested rate as long as the sum of the rates does not exceed the link capacity.

Several packet schedulers have been proposed that approximate GPS in an environment where only one packet at a time can be transmitted, instead of several simultaneously. These strike various trade-offs between *accuracy of approximation* and *implementation complexity*, two properties that have been proved to be in opposition [33]. In particular, the accuracy of a scheduler can be measured by its *latency*, i.e., its worst-case scheduling delay; in other words, the maximum lag of a scheduler's service with respect to GPS. The smaller this lag, the more closely that scheduler's operation will resemble that of GPS.

At one end of the spectrum lie *Packet-by-packet Generalized Processor Sharing* (PGPS) and *Worst-case Fair Weighted Fair Queueing* (WF^2Q), both exhibiting the smallest possible latency. More specifically, their latency is inversely proportional to the requested rate, plus a small additive constant; we thus call these *Strictly Rate-proportional* (SRP) schedulers. Their downside is that they have a relatively high complexity, i.e., $O(\log n)$, n being the number of active flows[1]. This might be an issue on high-speed links, where scheduling decisions have to be made in a packet transmission time (few nanoseconds) and many flows can be active simultaneously.

At the other end of the spectrum we find instead *Frame-based* (FB) schedulers such as Deficit Round-robin (DRR) [30] and its derivatives (e.g., [16–18]). These have constant (i.e., $O(1)$) complexity, a result that is only possible because the order of service of backlogged flows is constant over time. As a consequence, their latency is looser than those of the SRP schedulers, since it includes a term which grows *linearly* with the number of flows, corresponding to the time it takes to cycle through all the flows.

In between these two extremes, we have two other possibilities. On one hand, approximations of the GPS paradigm based on *flow grouping* [4,13], which achieve $O(1)$ complexity by using clever data structures, but constraining flow rates to be integer multiples of a basic quantity. The latency expression of these *Group-based* (GB) schedulers is similar to that of SRP schedulers, albeit with higher multiplicative and additive constants. On the other hand, schedulers that dispense with some of the intricacies of emulating a GPS server, hence have a higher latency, still at $O(\log n)$ complexity. This is the case, for instance, of Self-clocked Fair Queueing (SCFQ, [11]). The latency of this scheduler has an additive term that grows linearly with the number of flows, hence we call it *Weakly Rate-proportional* (WRP).

When WFQ packet schedulers are employed, it is possible to compute the worst-case end-to-end delay (WCD) of a flow in a multi-hop path, once its settings—specifically, its minimum guaranteed rate and latency—at each link it traverses are given. In particular, the WCD expression includes the sum of the latencies at all traversed links. A WCD guarantee is important for many applications, e.g., playback-based ones (voice, video, . . .) and real-time

[1]Note that the complexity was believed to be $O(n)$ until [31] proved otherwise. This misconception, lasting for about a decade, made pursuing *approximations* of these schedulers at $O(\log n)$ complexity a worthy task, which was in fact undertaken by several researchers.

ones (machine-to-machine applications, augmented/virtual reality, automated trading, ...). The above-mentioned property allows one to select *rates* on the given path so that the WCD stays below a pre-specified deadline. In turn, this paves the way to algorithms that *compute network paths* where enough rate is available to meet that deadline, something that we call *delay-constrained routing* (DCR) henceforth.

DCR belongs to the field of *QoS routing*, also well researched in the last two decades. However, most of the works on delay-based QoS routing assume *static* additive per-link delays (e.g., [19]), hence neglect the contribution of queueing to the overall end-to-end delay. Alternatively, *stochastic* traffic models are used to compute *average* end-to-end delays (e.g., [28]), which however do not provide reliable guarantees on worst-case ones, and therefore cannot be used in sensitive applications. Two works concerned with DCR are [21, 22], which show that path computation assuming SRP schedulers is $\mathcal{NP}$-hard in general. If, however, the *same rate* is reserved at each link, then DCR becomes a polynomial problem. As recently shown in [6, 8], reserving the same rate at each link is largely suboptimal, i.e., a significant fraction of path requests are rejected unnecessarily only because of that assumption. Furthermore, despite $\mathcal{NP}$-hardness, *optimal* solutions can be found in split-second times even in large-scale networks.

These initial results call for a more systematical investigation of the DCR problem. In [6, 8] only SRP schedulers are treated, and no result exists for the other three categories (GB, WRP, FB), despite the fact that most of these schedulers have been around for decades and are used in practice in today's equipment (especially DRR and variants thereof). This means that the trade-off between employing lower-complexity schedulers (e.g., GB or FB ones) and the corresponding utilization of the network has not been properly characterized yet. This work aims at filling this gap by formulating the DCR problem for all the above-mentioned categories of schedulers. More specifically, given the current state of the network, a *reservation cost* per unit of rate for each link, a source, a destination and a WCD deadline, we determine a path along which the new flow can be routed and the corresponding rate reservation on each link (if they exist) so that the deadline is met, while existing flows still meet theirs, at the minimum possible reservation cost. The fact that the latency of a scheduler does or does not depend on the other flows (their number and/or current reserved rates) simultaneously present on the link affects the way *admission control* needs to be done at flow setup. With SRP and GB schedulers, a new flow can always be admitted as long as the used links have enough rate; hence admitting a new flow will never jeopardize the guarantees of pre-existing flows traversing the same links. For WRP and FB schedulers, instead, checking rates is not sufficient: a flow may in fact be rejected even when there is enough rate, because admitting it would make *other flows* violate their delay guarantees, since *their* latency would grow if the new flow were admitted.

We show that, despite these differences, the DCR problem can always be formulated as a Mixed-Integer Second Order Cone Program (MI-SOCP), which can be solved in split-second time on off-the-shelf hardware for networks of fairly large size. This makes our approach viable in practice wherever centralized path computation is advocated, e.g., amenable to be incorporated into Path Computation Elements [24] or the control plane of Software Defined Networks. On the other hand, being able to solve the DCR problem optimally for several categories of schedulers allows us to compare these categories as for QoS routing performance, i.e., to better characterize the cost that has to be payed, in terms of network utilization, for using lower-complexity schedulers.

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 solution approaches. These are evaluated numerically in Section 5. Finally, Section 6 reports conclusions and highlights directions for future work on this topic.

2 Related Work

We now describe the most relevant related work on QoS routing and optimal path computation. QoS routing is concerned with finding paths subjects to QoS constraint, such as a minimum guaranteed bandwidth, a maximum delay or jitter, etc.. A seminal paper on the topic is [32], which shows that computing a shortest path under two or more *additive* or *multiplicative* constraints is $\mathcal{NP}$-complete. For instance, per-link delays are additive, and per-link loss probabilities are multiplicative. However, path computation with one additive/multiplicative constraint and *concave* constraints (such as a minimum available bandwdith along a path) is instead polynomial.

There has been a large amount of literature devoted to finding approximate solutions for the multi-constrained QoS routing problem (e.g., [12, 14, 15, 34]), or advocating doing part of the work *offline* (the so-called "pre-computation" approach) to make the online part faster [23]. All the above works, however, consider link delay as a static per-link metric, hence neglect queueing. Relatively fewer works [21, 22, 27] aim to find paths *and* per-link rate reservations that meet a pre-specified non-additive end-to-end worst-case delay. This is possible because the WCD is a decreasing function of (among other things) the rates reserved at each link, if the sending rate of the flow is upper bounded [3]. The above works assume PGPS schedulers [26] and leaky-bucket-shaped flows, and endeavour to find a path with enough available rate. It has been shown in [27] that path computation under the above assumptions is a polynomial problem if the rates to be reserved at each link are the same and known in advance. Later, [21] and [22] have shown that this is still true even with unknown rates, as long as they are all the same. Allowing rates to be different, instead, makes the problem $\mathcal{NP}$-hard. However, our previous works [6,8] have shown that constraining the rates to be the same at all links comes with a high cost in terms of network performance: removing that assumption, in fact, abates the flow rejection probability considerably. Moreover, even though path computation becomes non-polynomial, it is still solvable in split-second times for fairly large-scale networks, hence online path computation with unequal rates is practicable. Furthermore, heuristic approaches can be devised that are much faster than exact solution methods while achieving solution of similar quality, thus reinforcing the above claim.

All these works only consider PGPS schedulers. However, their results can obviously be applied without any modification to every scheduler whose latency expression is the same as PGPS, i.e., to any SRP scheduler. A first classification of schedulers based on their latency expression—and on the impact that these expressions have on optimization problems—can be found in [20]. In that work, group-based approximations of WFQ are not mentioned, since they have only appeared more recently. To the best of our knowledge, this is the first work considering delay-constrained routing using non-SRP schedulers.

3 Background and system model

This section details the hypotheses underlying our contribution, and provides the necessary background on worst-case delay computation and latency expressions.

The network where flows have to be routed is represented as a directed graph $G = (N, A)$, where N is the set of nodes (i.e., routers or switches) and A is the set of arcs (i.e., links) joining them. A node $i \in N$ is characterized by a fixed *node delay* n_i, representing the time it takes for a packet to travel from an input interface to an output interface. A link $(i, j) \in A$ is characterized by its constant *propagation delay* l_{ij} and its *capacity* w_{ij}. Moreover, the *maximum transmit unit* (MTU) L is known and assumed to be constant throughout the network for simplicity.

We focus on a tagged flow to be routed through the network, and call $s \in N$ and $d \in N \setminus \{s\}$ its source and destination nodes. The flow has an end-to-end *deadline* δ, and the routing must be such that the WCD of the flow must not exceed δ. A WCD guarantee can only be given if the injection rate of the flow is upper-bounded. Such upper bound can be expressed in the form of an *arrival curve* $A(\tau) : \mathbb{R}_+ \to \mathbb{R}_+$, which is a wide-sense increasing function which bounds from above the number of bits that the flow is allowed to send in any interval of length τ. That is, if $F(t)$ measures the overall number of bits injected by the flow by time t, we have $F(t + \tau) - F(t) \leq A(\tau)$ for all t and $\tau \geq 0$. We assume that the arrival curve is *affine*, i.e. $A(\tau) = \sigma + \rho \cdot \tau$. This curve is often referred to in the literature as the *leaky-bucket* arrival curve, and its two non-negative parameters σ and ρ are called *burst* and *rate*, respectively.

We assume that each link is managed by a WFQ scheduler (e.g., PGPS, DRR, etc.), where each flow traversing the link has to specify its *reserved rate* for the link. Obviously, in order to verify whether or not a feasible routing exists for the new flow it is necessary to know the current *state* of the network. This is specified by the set K of existing flows: each flow in $k \in K$ is characterized by its chosen path $p(k)$ between its source and its destination, its deadline δ^k, its burst and rate parameters σ^k and ρ^k, and its reserved rates r^k_{ij} for each link $(i, j) \in p(k)$. We will similarly denote by p the path chosen for the new rate, and by r_{ij} the chosen reserved rates for $(i, j) \in p$. Finally, we will denote by $P(i, j) = \{\, k \in K \,:\, (i, j) \in p(k) \,\}$ the set of *existing* flows traversing the arc (i, j); note that this does *not* comprise the new one, which has yet to be routed. All this data is clearly required to define the conditions under which a feasible path p and reserved rates exist for the new flow. For instance, any WFQ scheduler guarantees that—over a suitable interval of time—the new flow will be served at least with the required rate r_{ij}, regardless of the presence of other flows; however, this holds only provided that the link is not oversubscribed, i.e.,

$$r_{ij} + \textstyle\sum_{k \in P(i,j)} r^k_{ij} \leq w_{ij} \quad . \tag{1}$$

Also, in order for the WCD to be *finite* the minimum rate among all links of the path must be at least as large as the traffic injection rate of the flow, i.e.,

$$r_{ij} \geq \rho \qquad \forall (i, j) \in p \quad . \tag{2}$$

Under (2), the WCD along path p is

$$\frac{\sigma}{\min\{\, r_{ij} \,:\, (i, j) \in p\}} + \sum_{(i,j) \in p} \big(\, \theta_{ij} + l_{ij} + n_i \,\big) \tag{3}$$

where θ_{ij} is the *latency* that the flow experiences on link (i, j). That latency models the delay that the head-of-line packet of the tagged flow undergoes due to the scheduling process, and its expression varies from one scheduler to the other. Following and extending [20], we can classify WFQ schedulers into four classes, depending on their latency expressions:

- *Strictly Rate-Proportional (SRP) latency*, i.e., the one of Packet-by-packet GPS (PGPS, [26], also called WFQ) and Worst-case Fair Weighted Fair Queueing (WF^2Q, [2]). The expression for SRP latency is:
$$\theta_{ij} = \frac{L}{w_{ij}} + \frac{L}{r_{ij}} \quad . \tag{4}$$
Note that the latency is inversely proportional to the reserved rate, barring an additive constant L/w_{ij} (which is unavoidable and due to atomic packet transmission), hence the name SRP. Thus, the latency can be reduced by increasing the flow's reserved rate. It has been proved that SRP latency can only be achieved at $O(\log n)$ worst-case per-packet complexity [31, 33], n being the number of flows traversing the link.

- *Group-Based (GB) approximations of WFQ*, e.g. [4, 13]. In these schedulers, flows are grouped by requested bandwidth at logarithmic intervals, which ensures $O(1)$ complexity at the price of a larger latency. The link latency expression is [4]
$$\theta_{ij} = 2\frac{L}{w_{ij}} + 3\frac{2^{\lceil \log_2 w_{ij} L/r_{ij} \rceil}}{w_{ij}} \quad , \tag{5}$$
which can be easily shown to satisfy
$$2\frac{L}{w_{ij}} + 3\frac{L}{r_{ij}} \leq \theta_{ij} \leq 2\frac{L}{w_{ij}} + 6\frac{L}{r_{ij}} \quad . \tag{6}$$
Hence, the latency is still rate-proportional. However, there is a constant (≥ 3) multiplying the rate-dependent term and a larger constant offset. Thus, it is in general larger than (4). We call (5) a *Group-Based* latency.

- Schedulers with *Weakly Rate-Proportional* (WRP) latency, e.g. Self-clocked Fair Queueing [11]. SCFQ was introduced as a simpler approximation of the GPS paradigm, since it does not need to emulate a GPS scheduler. However, it still exhibits logarithmic complexity, and its latency depends on the *number of flows* $|P(i, j)|$ traversing the link simultaneously:
$$\theta_{ij} = |P(i, j)|\frac{L}{w_{ij}} + \frac{L}{r_{ij}} \quad . \tag{7}$$
If $|P(i, j)|$ is high, increasing the reserved rate may decrease the latency only marginally, hence the name given to this class.

- *Frame-Based* (FB) schedulers, such as DRR [30] and similar [16,17], which achieve $O(1)$ complexity by imposing that flows are visited in a fixed order, each for a minimum amount of time (called a *quantum*). The quantum determines the guaranteed rate, which is in fact the ratio of the quantum to the round duration. Thus, in these schedulers latency depends on the number of flows, but also on their quantum, hence on the reserved rate for *each* flow. Note that $O(1)$ complexity can only be guaranteed if all quantas are *lower bounded*. In DRR, the quantum lower bound is equal to the MTU L. Moreover, quantum allocation also reflects rate partitioning, i.e., one flow is guaranteed double the rate as another flow if and only if its quantum is twice as large. This implies that the flow requesting the minimum reserved rate must get a quantum equal to the lower bound L, and all other flows get their quantum accordingly. Thus, the latency expression of DRR, besides the number of flows, also depends on the reserved rates of *other* flows on the link. More specifically, it depends on *both* their sum

$$\bar{r}_{ij} = \sum_{k \in P(i,j)} r_{ij}^k$$

and their minimum

$$r_{ij}^{min} = \min\{ r_{ij}^k \,:\, k \in P(i,j) \} \ .$$

The latency of the DRR scheduler has been computed in [17]; it is easy to verify that the formula obtained therein can be rewritten as

$$\theta_{ij} = \frac{L}{w_{ij}} \frac{\bar{r}_{ij}}{\min\{ r_{ij} \,,\, r_{ij}^{min} \}} + |P(i,j)| \frac{L}{w_{ij}} + \frac{L}{r_{ij}} \ . \tag{8}$$

Other frame-based schedulers (e.g. [16, 17]), which are variants of the basic DRR scheme, achieve smaller latencies by scaling down the lower bound on the quanta (i.e., below the MTU size) by a constant factor κ, and avoiding the complexity penalty by using clever data structures. The resulting latency has the similar expression

$$\theta_{ij} = \frac{L}{w_{ij}} \frac{\bar{r}_{ij}}{\kappa \cdot \min\{ r_{ij} \,,\, r_{ij}^{min} \}} + |P(i,j)| \frac{L}{w_{ij}} + \frac{L}{r_{ij}} \ .$$

The limit for $\kappa \to \infty$ exactly reproduces the WRP latency (7). However, the cost of the scheduling algorithm, either in time or in space if implemented in hardware, grows with κ, which means that κ cannot be taken arbitrarily large. Yet, clever implementations allow to select κ so as to get quite close to the WRP latency, at a reasonable cost; the interested reader is referred to [17] for details. For simplicity in the following we will only work with $\kappa = 1$.

The aim of this paper is to formulate and solve the *Delay-constrained Routing* (DCR) problem: given the current state of the network and a set of link *reservation costs* $f_{ij} > 0$—i.e., the cost of reserving one unit of capacity on (i, j)—find *one* feasible *s*-*d* path, and a feasible reserved rate at each of its links, so that the flow can be routed along the path and meet its end-to-end deadline at the minimum possible reservation cost. Obviously, admitting a new flow must not jeopardize the delay guarantees of other flows already present in the network. This is a practical concern, since both the latency formulas of WRP (7) and FB (8) schedulers include terms that depend on the number of flows traversing the link. Furthermore, (8) also includes the minimum rate reserved for a flow at that link. Therefore, admitting a new flow, by increasing some other flow's latency, might make them violate their deadline, in which case the new flow must be rejected. In other words, the admission of a new flow, for these schedulers, requires *global admission control* to ensure that all other established flows keep meeting their deadlines. Note that this problem cannot occur with

SRP or GSRP schedulers, since their latencies only depend on the rate of the new flow being routed. For these, therefore, global admission control is not required, and the admission of a new flow is only conditioned by the availability of rate along the chosen path. So far, the DCR problem has only been dealt with in the context of SRP schedulers in [6, 8], where it has been proven to be a Mixed-Integer Second Order Cone problem (MI-SOCP). In the following, we formulate it for the other categories of schedulers—notably, those requiring global admission control.

4 Mathematical programming formulation

In [6, 8], the DCR problem for SRP schedulers was formulated as a MI-SOCP problem. We recall that formulation hereafter, as a baseline to understanding its generalizations.

To model routing, we use binary variables $x_{ij} \in \{0, 1\}$ to indicate whether link (i, j) belongs to p: this allows us to write down the standard *flow conservation constraints*

$$\sum_{(j,i)\in BS(i)} x_{ji} - \sum_{(i,j)\in FS(i)} x_{ij} = \begin{cases} -1 & \text{if } i = s \\ 1 & \text{if } i = d \\ 0 & \text{otherwise} \end{cases} \qquad i \in N \ . \tag{9}$$

If the (indirect) cost of setting any $x_{ij} = 1$ is positive, (9) ensure that—at optimality—the x variables represent an s-d path. In (9), $BS(i)$ is the subset of A containing the arcs entering node i (the so-called "backward 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). We then introduce rate reservation variables r_{ij}, that are instead continuous, and an additional variable r_{min}, with an obvious meaning. For these we define the *reservable capacity* $c_{ij} \leq w_{ij} - \bar{r}_{ij}$ at each arc: $r_{ij} \leq c_{ij}$ ensures that (1) holds. Note that c_{ij} may be chosen to be strictly smaller than $w_{ij} - \bar{r}_{ij}$ if some of the link capacity has to be kept for other uses (signaling, backup paths, ...). Then, the constraints

$$0 \leq r_{ij} \leq c_{ij} x_{ij} \qquad (i, j) \in A \tag{10}$$

$$\rho \leq r_{min} \leq r_{ij} + c_{max}(1 - x_{ij}) \qquad (i, j) \in A \tag{11}$$

ensure on one hand that $r_{ij} = 0$ if $x_{ij} = 0$, and on the other hand that $\rho \leq r_{min} \leq r_{ij} \leq c_{ij}$ if $x_{ij} = 1$, so that both (1) and (2) hold. Note that $c_{max} = \max\{\, c_{ij} \ : \ (i, j) \in A \,\}$ is used in (11) to ensure that any link not in the chosen path ($x_{ij} = 0$) does not contribute to bounding r_{min} from above.

The constraint on the WCD given by (3) can be modeled using an auxiliary variable t and a *rotated SOCP* constraint as follows:

$$t + \sum_{(i,j)\in A} \Big(\theta_{ij} + \big(l_{ij} + n_i \big) x_{ij} \Big) \leq \delta \tag{12}$$

$$t\, r_{min} \geq \sigma \qquad , \qquad t \geq 0 \tag{13}$$

Note that the l_{ij} and n_i terms in the sum in (12) are only counted in if $x_{i,j} = 1$, i.e., if the link and node are actually in the chosen path 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, which in the SRP case looks as follows:

$$\theta_{ij} = \begin{cases} \frac{L}{r_{ij}} + \frac{L}{w_{ij}} & \text{if } x_{ij} = 1 \\ 0 & \text{if } x_{ij} = 0 \end{cases} \quad .$$

This is a *disjunctive set*, being expressed by a disjunction, which is in general *nonconvex*. Also note that $x_{ij} = 0 \Longrightarrow r_{ij} = 0$, which renders the L/r_{ij} term ill-defined. In [6] it is proven that the best approach to represent a disjunctive set makes use of *Perspective Reformulation techniques* [5, 9, 10] and results in:

$$\theta_{ij} = Ls_{ij} + (L/w_{ij})x_{ij} \qquad (i,j) \in A \tag{14}$$

$$s_{ij}r_{ij} \geq x_{ij}^2 \ , \quad s_{ij} \geq 0 \qquad (i,j) \in A \tag{15}$$

Clearly, the "$(L/w_{ij})x_{ij}$" term in (14) can be merged into the corresponding "$(l_{ij} + n_i)x_{ij}$" term in (12), obtaining

$$t + \sum_{(i,j)\in A} \left[Ls_{ij} + (L/w_{ij} + l_{ij} + n_i)x_{ij} \right] \leq \delta \tag{16}$$

subject to (15). Finally, we assume the linear objective function

$$\min \ \sum_{(i,j)\in A} f_{ij}r_{ij} \ , \tag{17}$$

i.e., we minimize the weighted amount of allocated rate along the path using the given reservation costs f_{ij}. The whole model is thus a Mixed-Integer Second Order Cone Program, due to the (rotated) conic constraints (13) and (15). MI-SOCPs can be solved (although, in general, *not* in polynomial time) by off-the-shelf, efficient, general-purpose solvers like `Cplex` or `GUROBI`. We remark that different formulations of some of these constraints are possible, which may result in the solvers to be able to solve the problems somewhat more efficiently in some cases, as discussed in [7]. However, the presented formulation is already efficient enough for all the cases we test, hence we stick to it for simplicity. We also remark that formulation (9)–(17) only requires knowledge of the other flows in the definition of the reservable capacities c_{ij}. In fact, with SRP schedulers, once a set of feasible rate reservations has been computed so that the new flow meets the required deadline, there is no need to check whether the other, already admitted flows still meet theirs provided that no link is over-reserved, i.e., (1) holds. Therefore, no further admission control constraints are required. This is no longer true for the other schedulers, which therefore require a different analysis.

4.1 Generalizations to other latency models

We now show that similar MI-SOCP models can also be derived when GB, WRP and FB schedulers are used. We refer the readers to [7] for more details on the derivations and possible alternative models.

4.1.1 Group-based Schedulers

The embodiment of (3) to the case of GB schedulers, whose latency is (5), is not straightforward. In fact, (5) is non-smooth, given the ceiling operator on the exponent. This would lead to complex non-linear, non-convex models, for which very few solvers are available and that could hardly be solved to optimality in split seconds. However, it is easy to observe that both the lower and upper bounds to (5) shown in (6) lead to convex models. In particular, using the upper bound in (6)—which is a safe choice, since we are discussing worst-case performance—we can just replace (14) with

$$\theta_{ij} = 6Ls_{ij} + (2L/w_{ij})x_{ij}$$

leaving (15) unchanged; this means that (16) becomes

$$t + \sum_{(i,j)\in A} \left[\, 6Ls_{ij} + (2L/w_{ij} + l_{ij} + n_i)x_{ij} \,\right] \leq \delta \ .$$

Because the latter only differs from (16) in two coefficients, there is no impact on the shape of the optimization model. Also, no "explicit" admission control mechanism is required in this case, save checking that enough capacity is available for the new flow similarly to the SRP case.

4.1.2 Weakly Rate-Proportional Schedulers

We now consider the latency model (7), which includes term $|P(i,j)|$. Again, the embodiment of (3) in this case is straightforward: just replace (14) with

$$\theta_{ij} = Ls_{ij} + (L/w_{ij})|P(i,j)|x_{ij} \tag{18}$$

leaving (15) unchanged. As $|P(i,j)|$ does not depend on the rates, this again has no impact on the shape of the optimization model. Note, however, that in this case *admission control* is required. In fact, unlike with SRP and GB, in this case a flow's latency depends on $|P(i,j)|$, which *changes* (for some arcs) *if a new flow is admitted*. Hence, we need to ensure that existing flows still meet their deadline after the new flow is admitted; if this is not the case, the new flow must be rejected, even though there may be enough rate for it to meet its own deadline. For each active flow $k \in K$, we therefore define the *delay slack* $\bar{\delta}^k$

$$\bar{\delta}^k = \delta^k - \frac{\sigma^k}{\min\{\, r_{ij}^k \,:\, (i,j) \in p(k)\}} - \sum_{(i,j)\in p(k)} \left(\frac{L}{r_{ij}^k} + (|P(i,j)| - 1)\frac{L}{w_{ij}} + l_{ij} + n_i \right) \tag{19}$$

which represents the amount of extra delay that flow k can tolerate, *without changing either its path* $p(k)$ *or its reserved rates* r_{ij}^k, while still meeting its deadline δ^k. Note that the term $|P(i,j)| - 1$ comes from the fact that the term "$|P(i,j)|$" in (7) does *not* count k, as it refers to the status where k had not been routed yet. On the contrary, when defining $\bar{\delta}^k$ the flow has already been routed, and it is obviously true that $k \in P(i,j)$ for all $(i,j) \in p(k)$ (which in particular means that $|P(i,j)| - 1 \geq 0$). Now, in order to ensure that the delay of k along $p(k)$, using the *fixed* reserved rates r_{ij}^k, does not increase more than $\bar{\delta}^k$, it is sufficient to add the linear *admission control constraint*

$$\sum_{(i,j)\in p(k)} (L/w_{ij})x_{ij} \leq \bar{\delta}^k \ . \tag{20}$$

It goes without saying that the model is still MI-SOCP, since the above admission control constraints are linear. Of course, one needs one constraint (20) for each $k \in K$, and therefore the number of added constraints w.r.t. the original SRP and GB formulation is $O(|K|)$.

4.1.3 Frame-based Schedulers

Finally, we consider FB schedulers. When comparing their latency expression against WRP's, (4), it is apparent that FB latency includes both a "simple" additive term $(L/w_{ij})|P(i,j)|$ like (7), *and* the rate-dependent term

$$\frac{L}{w_{ij}} \frac{w_{ij} - r_{ij}}{\min\{\, r_{ij} \,,\, r_{ij}^{min} \,\}} \tag{21}$$

However, clearly, (21) *only applies if* $x_{ij} = 1$, i.e., arc (i, j) is chosen to be in the path for the new flow; in fact, the term would otherwise go to $+\infty$ when $x_{ij} = 0 \Longrightarrow r_{ij} = 0$. Now, (21) is *not* a jointly convex function in r_{ij} and all the r_{ij}^k (r_{ij}^{min}). However the latter are *fixed* in this setting and therefore so is r_{ij}^{min}; this makes (21) convex in r_{ij}. In fact, the function

$$\phi(r_{ij}) = (w_{ij} - r_{ij})/\min\{\, r_{ij} \,,\, r_{ij}^{min} \,\}$$

that describes (up to a constant) (21) can be rewritten as

$$\phi(r_{ij}) = \begin{cases} \phi_1(r_{ij}) = w_{ij}/r_{ij} - 1 & \text{if } r_{ij} \leq r_{ij}^{min} \\ \phi_2(r_{ij}) = (w_{ij} - r_{ij})/r_{ij}^{min} & \text{if } r_{ij} \geq r_{ij}^{min} \end{cases} .$$

It is then immediate to see geometrically, and easy to verify algebraically, that not only $\phi_1(r_{ij}^{min}) = \phi_2(r_{ij}^{min})$, but also $\phi_1(w_{ij}) = \phi_2(w_{ij})[= 0]$. It follows that $\phi_2(r_{ij}) \geq \phi_1(r_{ij})$ for all $r_{ij} \in [r_{ij}^{min}, w_{ij}]$, whereas $\phi_1(r_{ij}) \geq \phi_2(r_{ij})$ for $r_{ij} \in (0, r_{ij}^{min}]$. Hence, in the interval $[\rho, w_{ij}]$ that matters for our problem one can alternatively define

$$\phi(r_{ij}) = \max\{\, \phi_1(r_{ij}) \,,\, \phi_2(r_{ij}) \,\} \;.$$

Hence, using standard representation of convex max-functions we can formulate (21) as

$$\theta_{ij} = Ls_{ij} \frac{L}{w_{ij}} |P(i,j)| x_{ij} + v_{ij} \qquad (i,j) \in A$$

$$v_{ij} \geq Ls_{ij} - L/w_{ij} \;,\;\; v_{ij} \geq (L/r_{ij}^{min}) x_{ij} - L r_{ij}/(w_{ij} r_{ij}^{min}) \;,\;\; v_{ij} \geq 0 \qquad (i,j) \in A$$

(again, including (15) as well).

As can be expected, admission control constraints for FB schedulers are more complex than WRP's due to the need to express the nonlinear term (21). However, one can use the same definition of delay slack used for WRP; this is because, as already mentioned, the FB latency is equal to WRP's plus the rate-dependent addendum. As the latter depends on the choices made for the new flow, hence is not constant, it cannot be included in the delay slack, i.e., the righ-hand-side of the constraint. In other words, one can write the admission control constraint as

$$\sum_{(i,j) \in p(k)} \frac{L}{w_{ij}} \left(x_{ij} + \frac{w_{ij} - r_{ij}^k}{\min\{\, r_{ij} \,,\, r_{ij}^{min} \,\}} \right) \leq \bar{\delta}^k \tag{22}$$

with $\bar{\delta}^k$ of (19). This is clearly related to the WRP version (20), but for the extra part (21); it has to be remarked, again, that that term only applies when $x_{ij} = 1 \implies r_{ij} > 0$ (otherwise it would send that term to $+\infty$). Exploiting the already discussed properties of (21), a SOCP formulation can be easily obtained:

$$\textstyle\sum_{(i,j)\in p(k)} (L/w_{ij})(x_{ij} + (w_{ij} - r^k_{ij})z_{ij}) \leq \bar{\delta}^k \qquad (23)$$

$$z_{ij} \geq 1/r^{min}_{ij} \quad , \quad z_{ij} \geq s_{ij} \qquad (i,j) \in p(k) \qquad (24)$$

Note that the $1/r^{min}_{ij}$ term is always well-defined because $(i,j) \in p(k)$, hence arc (i,j) is *not empty*—it contains at least the flow k—and therefore $r^{min}_{ij} > 0$. It is also important to remark that *neither the variables s_{ij} nor the z_{ij} depend on the flow k*; that is, these can be defined just once for all arcs $(i,j) \in A$ and then used to define the admission constraints for all the active flows. Actually, the z_{ij} variables only need to be defined for all (i,j) for which at least one active flow is routed. Therefore, in this case as well the model is still a MI-SOCP, despite the additional admission control constraints. Note that their number is higher than those of WRP, as constraints (23) are $O(|K|)$, whereas (24) are $O(|K| + |A|)$; also, some $O(|A|)$ extra variables are needed.

5 Numerical Results

We now analyze the impact on network performance of employing different schedulers for QoS routing. In general, simpler schedulers come with higher latencies, and this must reflect on their ability to admit traffic in the network. We analyze this effect by measuring the *blocking probability*, i.e., the relative ratio of unfeasible path computations, in several scenarios. We also show that solving to optimality the DCR problem is *affordable* with *all* the scheduler classes: the solution time is invariably well below one second, on off-the-shelf hardware, even for large networks and for a wide range of loads. This has already been shown for SRP schedulers in [8]; we now extend that result by showing that even factoring in admission control constraints (for WRP and FB schedulers) does not increase the computational burden significantly. Furthermore, modeling improvements may further improve the efficiency of the solution process, as discussed in [7].

5.1 Simulation setup

Constructing a set of meaningful instances to compare the various scheduling classes is a nontrivial exercise. We follow the guidelines of our earlier paper [8], which we summarize here for ease of reading. A number of real-world IP network topologies, shown in Table 1, are taken from the Internet Topology Zoo [1]. These topologies are heterogeneous with respect to network dimension, connectedness (represented by the average node rank) and geographic span (summarized by the average per-link propagation delay), ranging from regional (e.g., Belnet2009) to world-wide (e.g., DeutscheTelekom). Link delays l_{ij} are set according to the geographic coordinates, dividing the geodesic distance between i and j by the speed of light in a fiber. Node delays are selected equal to 40μs, a figure that can be expected to be on the safe side according to [25]. Link capacities are assigned using the `FNSS` tool [29], and

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

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

selected among $\{1, 10, 40\}$ Gbps, according to the *edge betweenness centrality* metric. `FNSS` also generates realistic *traffic matrices* based on the network capacity, and we exploit this to generate the ρ value of each request so that it can be accepted in an unloaded network. 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. Flow deadlines δ are set with the following process. We first compute two extreme values: δ_{min}, corresponding to the minimum feasible deadline obtained by allocating the entire link capacity and then calculating the delay-shortest path, *using SRP schedulers*, 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, still using SRP schedulers. Delay requests smaller than δ_{min} cannot be met, whereas requests higher than δ_{max} are likely to make the delay constraint redundant with SRP schedulers. Thereafter, δ is randomly chosen uniformly within the interval $[\,\delta_{min}\,,\, \delta_{min} + \beta(\delta_{max} - \delta_{min})\,]$ 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 request inter-arrivals are exponential 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 large enough to estimate blocking probabilities correctly even at low values of λ. Each point in the graphs is obtained as the average of five independent replicas, and 95% confidence intervals are also reported.

Simulations have been performed on a 2.299 Ghz AMD Opteron 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 MI-SOCPs were solved by off-the-shelf commercial solver `Cplex 12.6`, ran with default parameters. Thus, it is not unlikely that just tweaking the algorithmic parameters of the solver could buy us an additional speedup.

5.2 Results: blocking probability

We simulate path computation with all the scheduler classes, namely SRP, WRP, FB and GB; for the latter we use the *lower bound* approximation of the latency given in (5), for reasons that will become clear soon. We plot the blocking probability in Figure 1, as a function of λ, for all the above topologies when $\beta = 0.2$ and $\sigma = 3$MTU.

For all the schedulers classes, the blocking probability ranges from negligible (when $\lambda =$

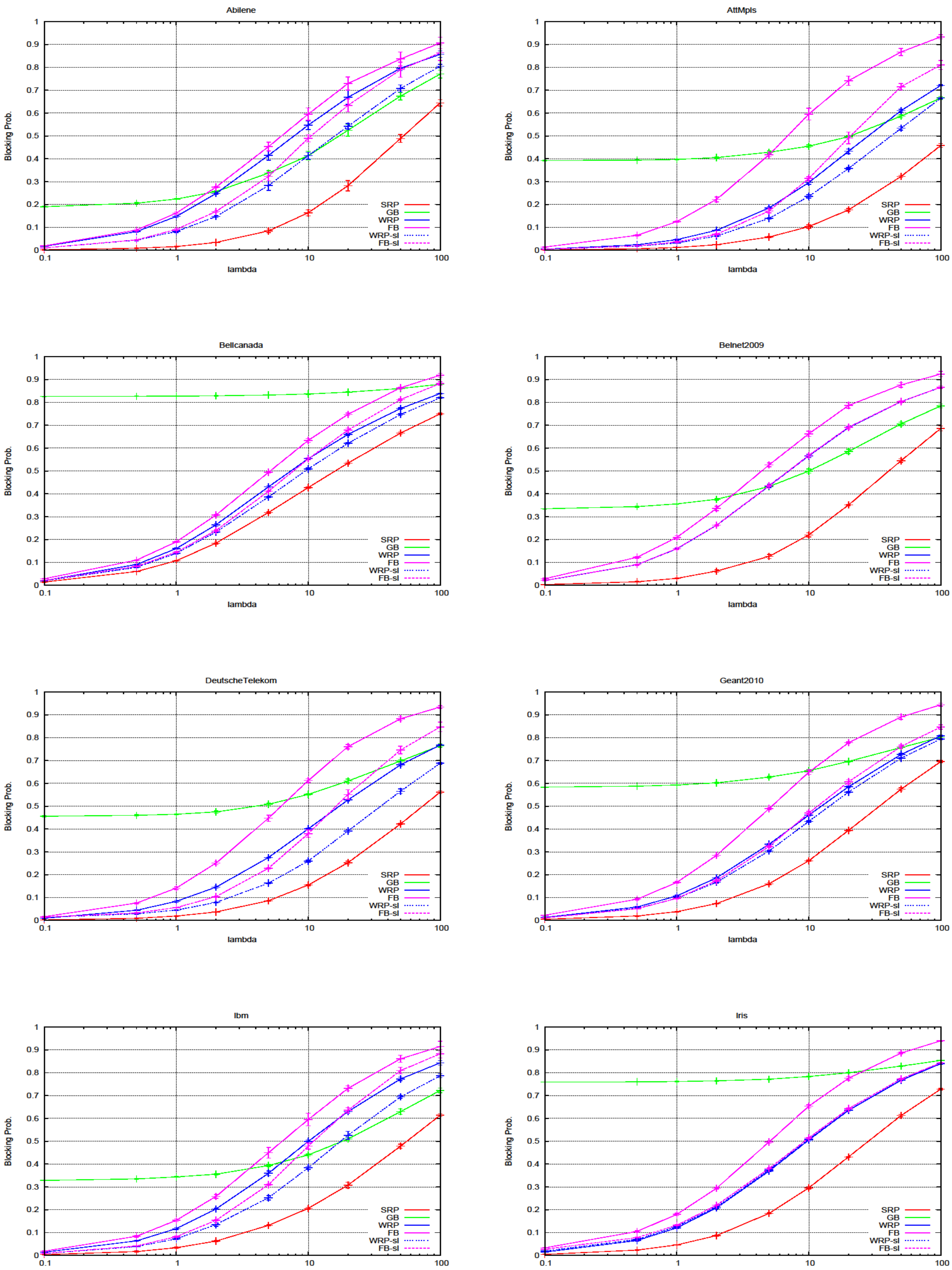

Figure 1: Blocking probability for all topologies, $\beta = 0.2$ and $\sigma = 3$ MTU

0.1) to almost certain (when $\lambda = 100$). The figure clearly shows that SRP outperforms all the other scheduler classes. What makes GB perform poorly is the fact that latency (5) includes a factor two to multiply the rate-dependent term. This fact is unique among all scheduler classes, and is such that GB is often unable to meet the deadlines (which, we recall, were already challenging for SRP schedulers) even in an unloaded network. Note that we have chosen a *lower bound* approximation on purpose, so as to discount the hypotheses that GB's poor performance were due to a pessimistic upper-bound approximation.

The charts show that WRP and FB perform considerably worse than SRP, although they exhibit the same dynamics; that is, their blocking probability does decrease at small loads, unlike that of GB. Among them, FB performs considerably worse than WRP, which is expected in that it always has larger latency for the same allocated rates (cf. the discussion about the extra term (21) in §4.1.3). The performance gap between WRP and SRP could be explained similarly, at least for larger loads where several active flows are present on the network: cf. the discussion about the extra term $|P(i,j)|$ in the latency formulæ of §4.1.2. However, a deeper analysis shows that an entirely different factor is also at play here: namely, *admission control*. The relevant mechanism is the following: when a flow is admitted, its rates are computed based on the *current* value of $|P(i,j)|$ (and of r_{min} for FB). In particular, because a minimum-cost allocation is sought, the *smallest* possible rates are computed. Given the inverse proportionality between rates and delay, this means that, for the given path, the selected rates are those that produce the *largest possible feasible delay*, i.e., δ. In other words: when a flow is admitted, its delay slack (19) is necessarily null. The consequence of this choice is that, unless some flow disappears later on, *any other* flow that attempts to use the same links will increase $|P(i,j)|$, thus increasing the WCD of any existing flow: but because the WCD is already at its maximum, even a fractional improvement is impossible. Thus, a new flow may be found to be impossible to route not because there is not enough rate to support it, but because of the fact that it would disrupt the current flows (i.e., because of admission control). In other words: once any flow seizes a link (i,j), it prevents any other flow from using it until it is removed.

The only way to get around this problem is to *overallocate* rates somewhat, under WRP and FB, so as to *buy flows some slack* at their admission. This can, for instance, be done by decreasing the deadline by a small percentage in the WCD constraint, so as to force the model to allocate more rates than these that would actually be necessary. That is, we can substitue the deadline constraint (12) with

$$t + \sum_{(i,j)\in A} \big(\theta_{ij} + (l_{ij} + n_i)x_{ij} \big) \leq \delta(1 - \varepsilon)$$

for some small $\varepsilon > 0$. This way, some delay slack is introduced to protect the flow from a subsequent increase in $|P(i,j)|$ and/or decrease in r_{min}. Of course, one should not overdo it, lest the problem is made unfeasible because the deadline is made too strict, or the increase in rate consumption makes it impossible for other flows to be admitted for lack of available capacity that could have been admitted otherwise. In order to verify that this mechanism is indeed significant we have repeated the experiments, for WRP and FB, with the small value $\varepsilon = 5 \cdot 10^{-5}$. The results are again plotted in Figure 1 (lines “WRP-sl” and “FB-sl”), alongside the non-slackened approaches. The charts show that even a small slack is enough to abate the blocking probability substantially in most topologies, thus confirming that the admission control is a significant factor for the blocking probability of WRP and SF schedulers.

5.3 Results: running times

Average computation times for all schedulers are shown in Figure 2, proving that computation times, being in the tens to hundreds of milliseconds, are indeed affordable. As observed in [8], computation times depend on the topology, and specifically on both its connectedness and dimension, the former having larger impact on the size of the solution space than the latter. All schedulers exhibit a decreasing trend with the load: this is because the optimization problem becomes unfeasible at higher loads, as testified by the increase in the blocking probability, and the solver detects unfeasible problems faster than it solves feasible ones of the same size. This phenomenon also explains why SRP is not consistently the fastest one despite having the simplest model; indeed, especially for large loads SRP usually requires more time than the other schedulers, largely because it has a lower failure rate. For analogous reasons, the "slackened" variants (WRP-sl and FB-sl) usually require more time than the corresponding non-slackened one (WRP and FB). On the contrary, at lower loads, where the failure rate of all the schedulers is comparable, the other models often are (fractionally) more costly to solve than SRP, which is expected due to them being more complex. In fact, FB is also typically (fractionally) more costly to solve than WRP, since it has more complex constraints, both for describing the latency of the new flow and for admission control. What is relevant, however, is that overall admission control constraints do not generate major overhead in the solution time, as testified by the fact that WRP and FB computation times, both with and without slack, are comparable to those of SRP and GB.

6 Conclusions and future research

In this paper we have proposed—to the best of our knowledge, for the first time—a centralized path computation and resource allocation approach that can be employed with all classes of fair-queueing schedulers. We believe that our work provides the following interesting contributions:

- It is possible to formulate the DCR problem for all relevant classes of fair-queueing schedulers as MI-SOCPs; this was not a foregone conclusion, since there was no a-priori guarantee that latency formulæ had to be convex, and a fortiori Second-Order Cone representable.
- As a consequence, there is an actual possibility to use WRP and FB schedulers for QoS routing purposes, using a centralized-decision approach, employing standard off-the-shelf optimization tools. The overhead of doing this is comparable to that of using more complex SRP schedulers, which again was not obvious since the corresponding mathematical models are more complex.
- However, WRP and FB perform worse than SRP in terms of network performance, i.e., what you gain in implementation complexity you pay for in terms of blocking

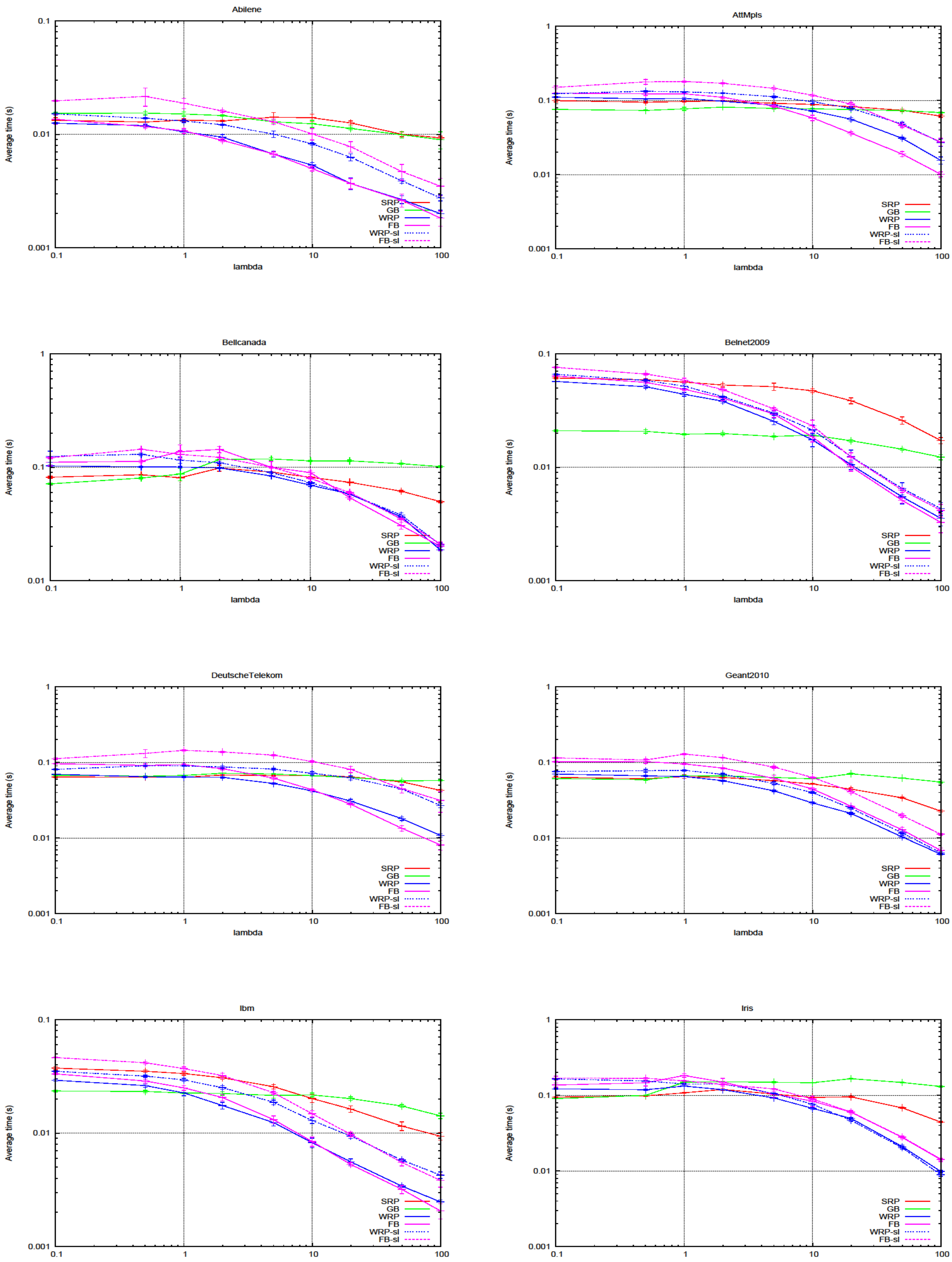

Figure 2: Average computation times for all topologies, $\beta = 0.2$ and $\sigma = 3$MTU

probability. While this could be expected, since WRP and FB have, all the rest being equal, higher latency than SRP, the magnitude of the difference really is relevant. This means that when designing a network for sustaining highly delay-sensitive flows, the choice of the scheduling protocol at the routers may play a more important role than is currently realized. In particular GB schedulers, while being a clever $O(1)$ approximation of SRP ones, perform rather poorly as far as QoS routing is concerned—even if we solve the problem at optimality while using a lower-bound approximation for the latency, thereby requiring smaller rates than what would actually be needed to meet the required QoS bound. To the best of our knowledge, our results are the first that offer insight in the trade-off between the cost of the scheduling protocol and the quality of the corresponding QoS routing.

- Our results for the first time reveal that the policy of minimizing the sum of rates allocated to the newly routed flow, which has hitherto been assumed in all the works on QoS scheduling, may not be the best one when admission control is required. In other words, minimizing the rate, while intuitively appealing, is not necessarily a good proxy for actual network performances, as measured e.g. by blocking probability. This means that different notions of "quality of obtained path and rates" will have to be developed in order to be embodied in the mathematical models. We have proposed a first, simple approach to this issue by selecting a fixed delay slack, but it is apparent that our proposal is a somewhat crude one: there must be an optimal slack—since both a null and a "high" ones lead to poor performance—but estimating it does not seem to be trivial.

We therefore believe that the topic deserves future investigations. In particular, it seems to be important to come up with more appropriate objective functions and/or constraints describing the quality of QoS routing than just the sum of the used rates along the links. As an example, it would be interesting to investigate slack selection policies for schedulers requiring admission control.

References

[1] The internet topology zoo.

[2] J.C.R. Bennett and H. Zhang. WF2Q: Worst-case fair weighted fair queueing. In *INFOCOM*, pages 120–128, 1996.

[3] J.-Y. Le Boudec and P. Thiran. *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet.* Springer LNCS, 2001.

[4] F. Checconi, L. Rizzo, and P. Valente. QFQ: Efficient packet scheduling with tight guarantees. *IEEE/ACM Trans. Netw.*, 21(3):802–816, 2013.

[5] A. Frangioni, F. Furini, and C. Gentile. Approximated Perspective Reformulations: a Project&Lift Approach. *Mathematical Programming*, (submitted), 2012.

[6] A. Frangioni, L. Galli, and M.G. Scutellà. Delay-constrained shortest paths: Approximation algorithms and second-order cone models. *Journal of Optimization Theory and Applications*, 164(3), 2015.

[7] A. Frangioni, L. Galli, and G. Stea. Delay-constrained routing problems: Accurate scheduling models and admission control. Technical Report, Dipartimento di Informatica, Università di Pisa, 2015.

[8] A. Frangioni, L. Galli, and G. Stea. Optimal joint path computation and rate allocation for real-time traffic. *The Computer Journal*, Vol. 58 No. 6, 2015.

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

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

[11] S.J. Golestani. A Self-Clocked Fair Queueing Scheme for Broadband Applications. *Proc. of IEEE INFOCOM'94, Toronto, Canada*, pages 636–646, 1994.

[12] A. Juttner, B. Szviatovszki, I. Mecs, and Z. Rajko. Lagrange relaxation based method for the QoS routing problem. In *Proceedings of INFOCOM'01, 22-26 April 2001, Anchorage, Alaska USA*, pages 859–868. IEEE, 2001.

[13] M. Karsten. Approximation of generalized processor sharing with interleaved stratified timer wheels. *IEEE/ACM Trans. Netw.*, 18(3):708–721, 2010.

[14] T. Korkmaz and M. Krunz. Multi-constrained optimal path selection. In *Proceedings of INFOCOM'01*, 2001.

[15] F. Kuipers, T. Korkmaz, and M. Kruntz. An overview of constraint-based path selection algorithms for QoS routing. *IEEE Communications Magazine*, 2002.

[16] L. Lenzini, E. Mingozzi, and G. Stea. Eligibility-Based Round Robin for Fair and Efficient Packet Scheduling in Interconnection Networks. *IEEE Transactions on Parallel and Distributed Systems*, 15(3):254–266, 2004.

[17] L. Lenzini, E. Mingozzi, and G. Stea. Tradeoffs between low complexity, low latency and fairness with deficit round robin schedulers. *IEEE/ACM Transactions on Networking*, 12(4):681–693, August 2004.

[18] L. Lenzini, E. Mingozzi, and G. Stea. Performance Analysis of Modified Deficit Round Robin Schedulers. *IOS Journal of High-Speed Networks*, 16(4):399–422, 2007.

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

[20] A. Lori, G. Stea, and G. Vaglini. Towards Resource-Optimal Routing Plans for Real-Time Traffic. In T. Margaria and B. Steffen, editors, *Leveraging Applications of Formal Methods, Verification, and Validation*, volume 6415 of *Lecture Notes in Computer Science*, pages 214–227. 2010.

[21] Q. Ma and P. Steenkiste. Quality-of-Service Routing for Traffic with Performance Guarantees. In *In Proc. IFIP International Workshop on Quality of Service*, pages 115–126, 1997.

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

[23] A. Orda and A. Sprinston. QoS routing: the precomputation perspective. In *Proceedings of INFOCOM'00, 26-30 March 2000, Tel Aviv, Israel*, pages 128–136. IEEE, 2000.

[24] F. Paolucci, F. Cugini, A. Giorgetti, N. Sambo, and P. Castoldi. A Survey on the Path Computation Element (PCE) Architecture. *IEEE Communications Surveys Tutorials*, 2013.

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

[26] K. Parekh and R.G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: the Single Node Case. *IEEE/ACM Trans. on Networking*, 1:344–357, 1993.

[27] C. Pornavalai, G. Chakraborty, and N. Shiratori. QoS based routing algorithm in integrated services packet networks. Proceedings of ICNP'97, pages 167–174. 1997.

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

[29] L. Saino, C. Cocora, and G. Pavlou. A toolchain for simplifying network simulation setup. In *Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques*, SIMUTOOLS '13, ICST, Brussels, Belgium, Belgium, 2013. ICST.

[30] M. Shreedhar and G. Varghese. Efficient fair queueing using deficit round robin. *IEEE/ACM Transactions on Networking*, 4:375–385, June 1996.

[31] P. Valente. Exact gps simulation with logarithmic complexity, and its application to an optimally fair scheduler. In *SIGCOMM*, pages 269–280, 2004.

[32] Z. Wang and J. Crowcroft. Qos routing for supporting resource reservation. *IEEE Journal on Selected Areas in Communications*, September 1996.

[33] J. Xu and R.J. Lipton. On fundamental tradeoffs between delay bounds and computational complexity in packet scheduling algorithms. In *SIGCOMM*, pages 279–292, 2002.

[34] X. Yuan. Heuristic Algorithms for Multiconstrained QoS routing. *ACM/IEEE Trans. on Networking*, 2002.