A CASE FOR SIMPLICITY IN PROVIDING NETWORK QUALITY OF SERVICE: CLASS-BASED STRICT PRIORITY QUEUEING

Jens Schmitt, Frank Zdarsky
Computer Science Department, University of Kaiserslautern
Distributed Computer Systems Lab (DISCO)
{jchmitt,zdarsky}@informatik.uni-kl.de

Abstract -- In this paper, we make a case for a *simple* alternative in providing quality of service (QoS) in packet-switched networks: *class-based strict priority queueing*. Simplicity here is meant in a multi-faceted sense with respect to (1) implementation complexity in routers, (2) service interface towards network users, (3) analytical tractability, and (4) configuration and management ease. We find basic properties of the worst-case behaviour in strict priority queueing systems using network calculus. Besides the newly derived worst-case characteristics, there are known results for the average behaviour of strict priority queueing systems from traditional queueing theory. These results are contrasted against the worst-case results by means of numerical investigations. They provide on the one hand a feeling for a provider as to how conservative the worst-case bounds are as well as on the other hand a hint for users on what service they can expect from the network on average.
The rationale behind this work is the appealing simplicity as well as the almost ubiquitous availability of strict priority queueing in today's routers and the thus promising applicability of our results for practical purposes in providing QoS in the Internet.

I. INTRODUCTION

To provide multiple differentiated services over the Internet is a notoriously difficult problem. Many different schemes have been devised and supported in standard efforts ([1], [2]). Yet, success looks different. There is of course many reasons and often they are much more contrived than just technical issues like scalability. We argue that one of the big problems of existing approaches is a lack of simplicity and availability. Therefore, we want to make a step towards a very simple and even today already available solution based on strict priority queueing (many router products have been offering strict priority queueing for some time, see e.g. [3]). Furthermore, we want to keep the interface towards the differentiated services simple by providing worst-case properties like the maximum delay that may be experienced by a flow. From our perspective it is important that *the service interface allows for per-flow guarantees while the service implementation is only based on class differentiation.* This allows for both, simplicity in implementation and simplicity in the semantics of the "service contract".

A. Class-based Strict Priority Queueing

Strict priority queueing, which is available in many router products (e.g., in Cisco routers it is available under the label LLQ (Low Latency Queueing) [3]), provides some enhancement to traffic management. A priority queueing mechanism adds the ability to sort packets based on differences in priority and insert them into separate internal queues or shuffle their insertion within a single queue. The forwarding algorithm always transmits packets of the highest priority first. If there are no packets of the highest priority level, the next highest priority queue is serviced, and so on. Strict priority queueing lends itself to simple implementation and as long as the number of priority classes is moderate it is also very efficient. The priorities in this type of queueing are absolute, i.e. if there is sufficient high priority traffic to saturate a link, all lower priority traffic is locked out. This can be a considerable problem, since some protocols attempt to use the entire available resource. For example, in the absence of competing traffic, a greedy TCP data flow will attempt to maximize its throughput and use all of the available capacity. Thus, a single TCP data flow with a higher priority can lock out all other flows for the duration of the TCP connection. In light of the potential for lower priority traffic to be locked out, great care must be taken when assigning priority levels. The solution to this problem is to enforce bounds on the use of higher priority classes which means to exert traffic regulation for high priority traffic.

A distinct advantage of strict priority queueing is its intuitiveness, there are no "magic" parameters which need "expert tuning". Different classes of service with a strict ordering among them is a well-known concept from many areas of life as for example air travel. This makes the configuration and management of a network based on class-based strict priority queueing very straightforward and intuitive.

B. Outline

In Section II, we recapitulate results for average behaviour in strict priority queueing systems under the assumption of Poisson arrivals and generally distributed service times. In Section III, the worst-case behaviour in strict priority queueing systems is analysed using network calculus and a simple admission control scheme which allows to offer per-flow delay and bandwidth guarantees despite purely class-based traffic control mechanisms on the data path of the system is presented. In Section IV, a comprehensive comparison of average and worst-case behaviour based on numerical investigations is provided. Section V reviews some related work in this area and Section VI concludes the paper.

II. STRICT PRIORITY QUEUEING: AVERAGE-CASE ANALYSIS

One of the strong results of queueing theory is given by the so-called Pollaczek-Khinchine formulae for M/G/1 queueing systems, i.e., systems with Poisson arrivals and general service times (although with finite variance).

THEOREM 1: *M/G/1 – Pollaczek-Khinchine Formulae for Average Waiting Time and Queue Size*

Let us assume service times have a general distribution with average $1/\mu$ and variance σ^2 and arrivals follow a Poisson process with parameter λ. Then the average waiting time, i.e., the average time packets spend in the queue, is given by

$$E(W) = \frac{\lambda(\sigma^2 + 1/\mu^2)}{2(1-\rho)} \text{ with } \rho = \frac{\lambda}{\mu}\text{, the utilization.} \quad (1)$$

Under the same conditions, the average number of packets in the queue is given by

$$E(q) = \frac{\rho^2}{2(1-\rho)}(1+\mu^2\sigma^2)\ . \quad (2)$$

Note that this theorem is under the additional assumption of FIFO queueing. There is many proofs of this famous result, three of which can be found in [4].

Since we assume strict priority queueing instead of FIFO queueing we need an extension on Theorem 1 under this scheduling discipline. In fact this exists and is given by the following theorem.

THEOREM 2: *M/G/1 – Average Waiting Time under Strict Priority Queueing*

Let us assume we have *n* classes each with Poisson arrivals with parameters $\lambda_1, \ldots, \lambda_n$ and general service time distributions with average $1/\mu_1, \ldots, 1/\mu_n$ and variance $\sigma_1^2, \ldots, \sigma_n^2$. The average waiting time for class *i*, $i = 1, \ldots, n$, is given by:

$$E(W_i) = E(T_0) / \left(1 - \sum_{j=1}^{i} \rho_j\right)\left(1 - \sum_{j=1}^{i-1} \rho_j\right) \text{ with } \rho_j = \frac{\lambda_j}{\mu_j}\text{, and}$$

$$E(T_0) = \lambda E(\tau^2)/2 = \sum_{j=1}^{n} \lambda_j E(\tau_j^2)/2 = \sum_{j=1}^{n} \lambda_j \frac{\sigma_j^2 + (1/\mu_j)^2}{2}$$

with $\lambda = \sum_{j=1}^{n} \lambda_j$ and τ the service time.

While this theorem is quite powerful it still assumes Poisson arrivals and is restricted to the single node case. While the former has been relaxed to some degree the general case is still intractable as of today, the latter is also extremely difficult to treat since due to the priority queueing arrivals at subsequent nodes become dependent on each other in non-trivial manner. Note that we mainly focus on the worst-case behaviour on which the per-flow admission control scheme is based. However, we consider it a great strength of strict priority queueing that also many results for the average-case behaviour exist, which on the one hand allows to assess our worst-case results on strict priority queueing as well as give users and network providers hints on how the system behaves on average.

III. STRICT PRIORITY QUEUEING: WORST-CASE ANALYSIS

In this section, we present basic worst-case properties of strict priority queueing[1] based on network calculus as derived in previous work of ours [5]. The necessary background on network calculus can be found in [6].

A. Strict Priority Queueing under General Arrival Curves

We analyse strict priority queueing for a given number of classes *n* and under the assumption that the input of each class *i* is constrained by arrival curve α_i for $i = 1, \ldots, n$.

THEOREM 3: *Service Curve for Strict Priority Queueing*

Let *C* be the overall capacity of the system. The service curve β_i^P for class *i* is given by

$$\beta_i^P(t) = \left(Ct - \sum_{j=1}^{i-1} \alpha_j(t) - L_i\right)^+ \quad (3)$$

for $i = 1, \ldots, n$, where $L_i = max_{i+1 \le j \le n}\{l_j^{max}\}$.

Here l_j^{max} is the maximum size of a packet in class *j*.

The theorem contains a very constructive result:

There is a quantifiable dependency of lower priority classes' service curves on the arrival curves of higher priority classes.

While the dependency between arrival and service curve might give an uneasy feeling about possible circular dependencies due to the fact that service curves operate on arrival curves to calculate quantities like maximum delay or maximum backlog it must be realized that they are not dependent on the actual arrival processes but can be treated as a given.

B. Strict Priority Queueing with Token Buckets

In this section, we now assume a particular arrival curve, the popular token bucket.

THEOREM 4: *Service Curve under Token Buckets*

Let $\alpha_j = \gamma_{r_j, b_j}$ be the arrival curve for traffic class *j*, $j = 1, \ldots, n$, i.e., each traffic class is constrained by a token bucket (each with its own token rate r_j and bucket depth b_j). The service curve for class *i* under strict priority queueing is then given by

[1] We assume the packet scheduling to be non-preemptive as usually assumed in this context. However, the case of preemptive strict priority queueing can be easily derived from the formulae given (L_i=0).

$$\beta_i^P = \beta_{R^P, T^P} \text{ with } R_i^P = C - \sum_{j=1}^{i-1} r_j \text{ and}$$

$$T_i^P = \left(\sum_{j=1}^{i-1} b_j + L_i\right) \Big/ \left(C - \sum_{j=1}^{i-1} r_j\right) \tag{4}$$

That means the service curve is of the rate-latency type [6].

Using the service curve for strict priority queueing the worst-case delay bound as well as the maximum backlog bound for each traffic class can be derived.

THEOREM 5: *Per-Class Backlog Bound under Token Buckets*

For stability we assume that $C \geq \sum_{i=1}^{n} r_i$.

The maximum backlog per traffic class *i* is bounded by the vertical deviation between the arrival curve to class *i*, γ_{r_i, b_i}, and its service curve, β_i^P

$$v(\gamma_{r_i, b_i}, \beta_i^P) = r_i \times \left(\sum_{j=1}^{i-1} b_j + L_i\right) \Big/ \left(C - \sum_{j=1}^{i-1} r_j\right) + b_i \tag{5}$$

THEOREM 6: *Per-Class Delay Bound under Token Buckets*

The maximum delay per traffic class *i* is bounded by the horizontal deviation between the arrival curve to class *i*, γ_{r_i, b_i}, and its service curve, β_i^P

$$h(\gamma_{r_i, b_i}, \beta_i^P) = \left(\sum_{j=1}^{i} b_j + L_i\right) \Big/ \left(C - \sum_{j=1}^{i-1} r_j\right) \tag{6}$$

So we can now compute besides the known results for average behaviour also the worst-case properties for strict priority queueing if we assume each traffic class conforms to a token bucket (respectively make it conform to it by either using admission control at ingress to the network or drop packets according to the token bucket).

C. Per-Flow Worst-Case Admission Control

We now use the basic results on worst-case bounds for strict priority queueing from the preceding subsection to design an admission control scheme which allows to give per-flow delay and rate guarantees despite the purely class-based strict priority queueing. Here, we assume that we are given maximum bandwidth shares per class, $\phi_i > 0$, $i = 1, \ldots, n$, with

$$\sum_{i=1}^{n} \phi_i \leq 1 \tag{7}$$

and want to ensure certain class delay targets D_i, $i = 1, \ldots, n$, with $D_i < D_j$ if $i < j$. The following theorem provides how the class token buckets have to be dimensioned:

THEOREM 7: *Dimensioning of Class Token Buckets*

To achieve the class delay targets D_i, $i = 1, \ldots, n$ and to ensure that each class obtains its bandwidth share ϕ_i, the class token buckets have to be chosen as

$$r_i = \phi_i C \tag{8}$$

$$b_i = D_i\left(C - \sum_{j=1}^{i-1} r_j\right) - \sum_{j=1}^{i-1} b_j - L_i \Leftrightarrow \sum_{j=1}^{i} b_j = D_i R_i^P - L_i \tag{9}$$

for $i = 1, \ldots, n$.

Note that (9) constitutes a system of *n* linearly independent equations of *n* unknowns, i.e., it always has a unique solution. This can be easily seen by computing the determinant of its coefficient matrix $A^{(n)}$. $A^{(n)}$ is a lower triangular matrix with

$$a_{ij}^{(n)} = \begin{cases} 1 & i \geq j \\ 0 & i < j \end{cases}$$

which results in $detA^{(n)} = 1$ independent of *n*.

Furthermore, as (9) results in a lower triangular matrix it is very simple to solve. However, if some of the b_i are negative this indicates that for the given bandwidth shares and class delay targets there is *no* allocation of token buckets which can achieve these.

This result can now be used for a simple admission control scheme: if a new flow with bandwidth requirements (r_{new}, b_{new}) and a maximum delay requirement d_{new} arrives it can be assigned to the lowest priority class i for which

$$d_{new} \geq D_i, \ \sum_{j=1}^{k} r_i^j + r_{new} \leq r_i \text{ and } \sum_{j=1}^{k} b_i^j + b_{new} \leq b_i \tag{1}$$

where r_i^j and b_i^j are the bandwidth requirements of already accepted flows in class i, $j = 1, \ldots, k$.

IV. AVERAGE AND WORST-CASE BEHAVIOUR IN CLASS-BASED STRICT PRIORITY QUEUEING

In this section, we perform some numerical investigations on the formulae presented in Section II and III. When comparing average and worst-case behaviour we need to keep in mind that the assumptions are quite different. For the average-case we assume Poisson arrivals whereas for the worst-case we have no restricting assumptions on the arrival process for a given traffic class other than that it is bounded by a token bucket.

At first, we provide a comprehensive numerical example of a strict priority queueing system with 8 traffic classes, before we then investigate the influence of different parameters more closely. For the sake of simplicity we assume in the following investigations that the maximum packet size is the same over all classes. Since traffic for these classes is aggregated traffic of possibly all kinds this is also a realistic assumption. In particular, we set the maximum packet size for all classes to 1500 bytes. Furthermore, we assume for the packet sizes that their average is 420 and their standard deviation is about 521 which corresponds to up-to-date measurements from [7].

A. Comprehensive Numerical Example

We assume 8 traffic classes each with 10% load of the overall server capacity which is assumed to be 100 Mbps. For each class' token bucket size we assume it to be 20% of the token bucket rate. This accounts for (infinitely fast) bursts of a volume corresponding to 200 ms of average activity in the class which seemed reasonable to us. In Table 1 the different delay and queue size values for all classes are given.

Class *i*	Av. Delay	W. C. Delay	Av. Queue Size	W. C. Queue Size
1	0.02	20	0.06	500
2	0.03	45	0.08	556
3	0.04	75	0.10	625
4	0.05	114	0.13	715
5	0.07	167	0.18	834
6	0.11	240	0.27	1001
7	0.18	350	0.45	1251
8	0.36	534	0.91	1668

Table 1: Average and worst-case delays (in ms) and queue sizes (in packets).

As can be seen, the worst-case behaviour is about 3 orders of magnitude above the average-case behaviour for delay and about 4 orders of magnitude above for queue size. While for both, average and worst-case behaviour, we can observe good differentiation between the classes we can see a more pronounced and balanced differentiation with respect to worst-case delay in particular for high priority classes.
In the following we examine the influence of different parameters, in particular the ratio of high priority traffic classes to low priority ones and the server capacity. For these it is for presentational purposes more convenient to look only at 2 classes instead of 8 since the basic effects can be more easily identified by contrasting high priority against low priority traffic.

B. *Ratio of High Priority Traffic*

In this experiment, the ratio between high and low priority traffic is varied while everything else is kept fixed, in particular the total load in the system is kept at 80%. Fig. 1 shows the average delay for an increasing load from high priority traffic while Fig. 2 shows the worst-case delays. While again being 3 orders of magnitude apart from each other we can observe a very similar behaviour: for increasing high priority traffic load the low priority traffic is punished harder once a certain load of high priority traffic is exceeded. This certainly calls for keeping

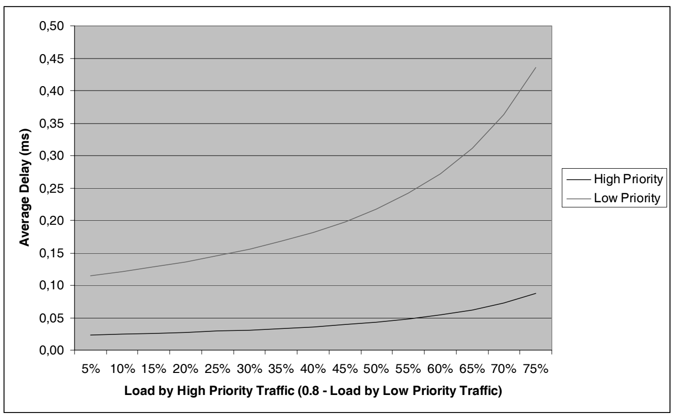

Figure 1: Average Delay for Different Ratios of High & Low Priority Traffic.

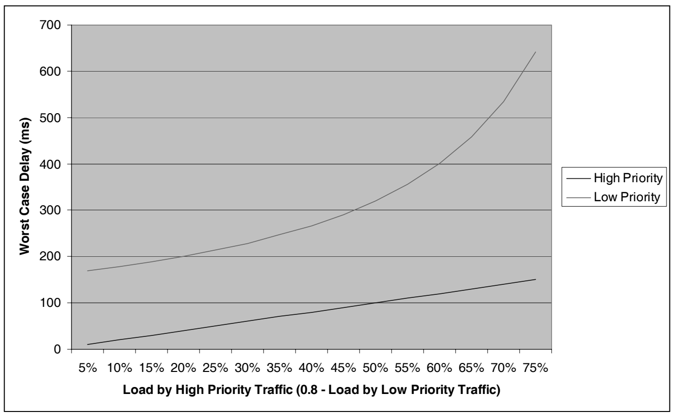

Figure 2: Worst-case Delay for Diff. Ratios of High & Low Priority Traffic.

the load from high priority traffic low since other traffic is otherwise suffering considerably.

C. *Server Capacity*

Next, we investigate the influence of the server capacity. Fig. 3 shows the average delay for varying server capacities, while Fig. 4 shows the worst-case delay. As can be seen, while increasing the server capacity has a positive influence on average delays it has no effect on the worst-case delays which remain almost constant. This means that the difference between worst-case and average delays very much depends on the server capacity and is growing with higher server capacity. This phenomenon should lead to possibly large deviations between delays, i.e. large jitter in high capacity systems – a reason to have more stringent control over the worst-case delay for traffic being sensitive to jitter.

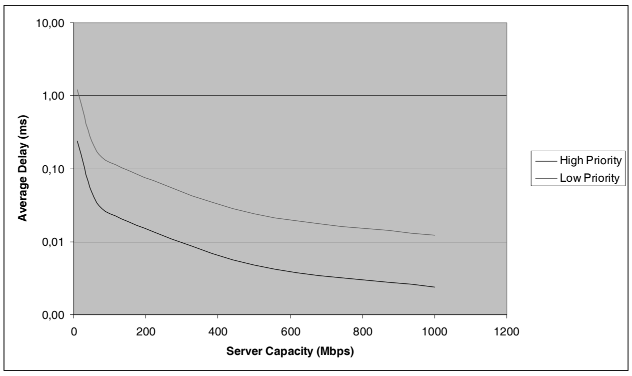

Figure 3: Average Delay for Different Server Capacities.

V. RELATED WORK

There is of course an almost intimidating body of work in providing QoS in the Internet (see [8] for a recent and excellent overview). An interesting practical work is described in [9]. They build up a small testbed consisting of Cisco routers and employ their priority queueing scheme. Their experimental results are absolutely consistent with the average queueing behaviour results which are predicted by an M/G/1 queueing system. Of course, due to the experimental nature they cannot provably make any statements about worst-case delays for priority queueing.

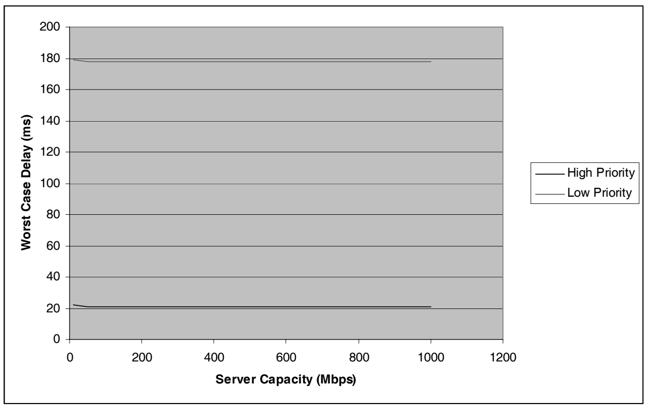

Figure 4: Worst-Case Delay for Different Server Capacities.

In previous work [5] we proposed the per-flow admission control scheme presented in Section III. Besides some refinements, this paper focuses on the evaluation of the overall framework of a strict priority queueing based network QoS solution for the worst as well as the average case behaviour, it is thus complementary to our previous work.

Closely related to our work are ([10], [11]). They derive a similar result as our Theorem 4 (see Section III.B) for the special case of two service classes in the context of DiffServ's Expedited Forwarding Per-Hop Behaviour (EF PHB) [12], however then focus on a different aspect, namely what they call aggregate scheduling. This is the problem where flows from multiple entry points to the network accumulate inside the network and how this affects the worst-case bounds. They assume no knowledge of the network topology and thus arrive at very restrictive bounds. We focus on the case where the topology and the paths taken by flows are known by the admission control scheme, e.g., by using a (logically) centralized bandwidth broker or by using multi-protocol label switching (MPLS) [13] to prevent flows to accumulate inside the network.

In [14], statistical guarantees for two class priority queueing are derived based on the so-called negligible jitter conjecture. We focus on worst-case respectively deterministic guarantees possibly enriched by hints on average behaviour. In our opinion, statistical assurances bring along a number of difficulties at the service interface since violations of the service contract cannot be interpreted unambiguously.

VI. CONCLUSION

In this paper, we tried to make a case for a *simple* and *available* alternative to provide network QoS: *class-based strict priority queueing*. Undoubtedly, it is simple to implement and understand, both from users' and providers' perspective. Moreover, we analytically investigated the behaviour of strict priority queueing systems both for average and worst-case and thus showed its (fairly) simple analytical tractability compared to other network QoS alternatives. Therefore, we perceive class-based strict priority queueing to (re)gain importance in packet networks due to schemes like DiffServ that allow to realise it as a cheap alternative for offering performance guarantees in so-called class of service networks.

REFERENCES

[1] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: an Overview. Informational RFC 1633, June 1994.

[2] K. Nichols, S. Blake, F. Baker, and D. Black. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. Proposed Standard RFC 2474, December 1998.

[3] Cisco Systems: Configuring Priority Queueing, 2000. Available at http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121cgcr/qos_%cqcprt2qcdpq.pdf

[4] T. G. Robertazzi. *Computer Networks and Systems*. Springer, 3rd Edition, 2000.

[5] Jens Schmitt, Paul Hurley, Matthias Hollick, and Ralf Steinmetz. Per-flow Guarantees under Class-Based Priority Queueing. In *Proceedings of IEEE Global Telecommunications Conference 2003 (GLOBECOM'03), San Francisco, CA, USA*, pages 4169-4175. IEEE, December 2003. ISBN 0-7803-7974-8.

[6] J.-Y. Le Boudec and P. Thiran. *Network Calculus - A Theory of Deterministic Queueing Systems for the Internet*. Springer, Lecture Notes in Computer Science, LNCS 2050, 2001.

[7] Cooperative Association for Internet Data Analysis (CAIDA). Packet Size Distributions, 2000. Available at http://www.caida.org/analysis/AIX/plen_hist/.

[8] V. Firoiu, J.-Y. Le Boudec, D. Towsley, and Z.-L. Zhang. Theories and Models for Internet Quality of Service. *Proceedings of the IEEE*, 90(9):1565–1591, September 2002.

[9] T. Ferrari, G. Pau, and C. Raffaelli. Measurement based analysis of delay in priority queuing. In *Proceedings of IEEE Global Telecommunications Conference 2001 (GLOBECOM'01)*, pages 1834–1840. IEEE, November 2001.

[10] F. Farkas and J.-Y. Le Boudec. A Delay Bound for a Network with Aggregate Scheduling. In *Proceedings of Sixteenth UK Teletraffic Symposium on Management of Quality of Service, Harlow, UK*, May 2000.

[11] A. Charny and J. Y. L. Boudec. Delay Bounds in a Network with Aggregate Scheduling. In *Proceedings of Quality of future Internet Services Workshop (QofIS 2000), Berlin, Germany*, pages 105–116. Springer LNCS, September 2000. ISBN 3-540-41076-7.

[12] B. Davie, A. Charny, J. Bennett, K. Benson, J.-Y. L. W. Courtney, S. Davari, V. Firoiu, and D. Stiliadis. An Expedited Forwarding PHB (Per-Hop Behavior). Proposed Standard RFC 3246, March 2002.

[13] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol Label Switching Architecture. Proposed Standard RFC 3031, January 2001.

[14] T. Bonald, A. Proutiere, and J. Roberts. Statistical performance guarantees for streaming flows using expedited forwarding. In *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'2001)*, pages 1104–1112. IEEE, April 2001.