Proceedings of the IEEE, May 2002

Theories and Models for Internet Quality of Service

Victor Firoiu, Jean-Yves Le Boudec, Don Towsley, and Zhi-Li Zhang

Abstract— **We survey recent advances in theories and models for Internet Quality of Service (QoS). We start with the theory of network calculus, which lays the foundation for support of deterministic performance guarantees in networks, and illustrate its applications to integrated services, differentiated services, and streaming media playback delays. We also present mechanisms and architecture for scalable support of guaranteed services in the Internet, based on the concept of a stateless core. Methods for scalable control operations are also briefly discussed. We then turn our attention to statistical performance guarantees, and describe several new probabilistic results that can be used for a statistical dimensioning of differentiated services. Lastly, we review recent proposals and results in supporting performance guarantees in a best effort context. These include models for elastic throughput guarantees based on TCP performance modeling, techniques for some quality of service differentiation without access control, and methods that allow an application to control the performance it receives, in the absence of network support.**

Keywords— **Quality of Service, Performance Guarantees, Network Calculus, Elastic Services, Differentiated Services, Integrated Services, Scalability**

I. Introduction

The problem of Internet QoS provisioning has been an extremely active area of research for many years. From the earlier Integrated Services (IntServ) architecture [1] to the more recent Differentiated Services (DiffServ) architecture [2], many QoS control mechanisms, especially in the areas of packet scheduling and queue management algorithms, have been proposed. Elegant theories such as network calculus and effective bandwidths have also been developed. Several books have been written on the subject, some focus more on architectural and other practical issues [3], [4], while others on theoretical aspects of QoS provisioning [5], [6]. To provide a more focused overview, in this paper we survey a number of *recent* advances in Internet QoS provisioning, with emphasis on *theoretical* developments. The objective is two-fold: 1) to provide the reader with the state-of-the-art knowledge in a few selective areas in Internet QoS provisioning, with pointers for further readings; and 2) to highlight the issues and challenges still facing the development of scalable Internet QoS provisioning solutions. The selected areas we will survey are: theory of network calculus for deterministic QoS guarantees; and architectures and solutions for scalable QoS support; newly developed theories for providing stochastic services; service differentiation within best effort; architectures and control algorithms for elastic services and adaptive application QoS control. Before we start our survey in these areas, we first introduce a few important notions and issues in Internet QoS provisioning. They will lay the background for our discussion later.

Network QoS can be defined in a variety of ways and include a diverse set of service requirements such as performance, availability, reliability, security, etc. All these service requirements are important aspects of a comprehensive network QoS service offering. However, in this paper we will take a more *performance-centric* view of network QoS and focus primarily on the issues in providing performance guarantees. Typical performance metrics used in defining network QoS are bandwidth, delay/delay jitter, and packet loss rate. Using these performance metrics, network performance guarantees can be specified in various forms, such as *absolute* (or *deterministic*), e.g., a network connection is guaranteed with 10 Mbps bandwidth all the time; *probabilistic* (or *stochastic*), e.g., network delay is guaranteed to be no more than 100 ms for 95% of the packets; *time average*, e.g., packet loss rate is less than 10^{-5} measured over a month. The *guarantee* feature of network QoS is what differentiates it from the "best-effort" network services. The exact form of performance guarantee will be part of the service level agreement (SLA) between the network service provider and its customers.

There are likely two major drivers for network services

The authors' affiliations and contact information are: Victor Firoiu, Nortel Networks, 600 Technology Park Drive, Billerica, MA 01821 USA, vfiroiu@nortelnetworks.com; Jean-Yves Le Boudec, EPFL-ICA, INN(Ecublens), CH-1015 Lausanne, Switzerland, jean-yves.leboudec@epfl.ch; Don Towsley, Department of Computer Science, University of Massachussetts, Amherst, MA 01003, towsley@cs.umn.edu; and Zhi-Li Zhang, Department of Computer Science and Engineering, University of Minnesota, 200 Union Street SE, Minneapolis, MN 55455, zhzhang@cs.umn.edu. This work was supported in part by the National Science Foundation under the grants EIA-0080119, ITR-0085848, EIA-9818338, ANI-0073819, ITR-0085824, and CAREER Award NCR-9734428 as well as by FRL/DARPA Contract No. F30602-00-2-0554. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

with QoS guarantees. One comes from applications with *stringent* QoS requirements. Two possible examples of such applications are IP telephony and video-on-demand (VoD) over the Internet. In IP telephony two end users send packetized voice and the quality of rendered sound depends on low delay and small loss rate of end-end packet transmission. Likewise, streaming videos over the Internet requires adequate bandwidth and packet loss guarantees from the network to ensure TV-broadcast quality. The other major driver for network QoS is the need for *service differentiation* due to competitive nature of the marketplace. For example, one network service provider may support a "virtual private network" (VPN) service over its network with only security guarantee but no performance guarantee. Whereas, another network service provider may support a "virtual leased line" (a form of VPN) service over its network that, in addition to security guarantee, has bandwidth, delay and loss guarantees comparable to a physical leased line. The first network service provider may be forced to enhance its VPN service also with performance guarantees or to lose its customers who demand performance guarantees to its competitor. Hence guaranteed QoS performance can serve as a service differentiating feature for network services.

Apart from over-provisioning network resources, providing QoS guarantees requires deployment of appropriate QoS control mechanisms in the operations and management of a network. A vast variety of QoS control mechanisms have been proposed and developed in last decade or so, with varying degree of complexity and cost. To help understand these mechanisms and their associated complexity and cost, we consider several important aspects of QoS controls.

A key aspect of QoS controls is the *time scale* at which a control mechanism operates. We can roughly divide the time scales of QoS controls into a few levels. The fastest time scale is at the *packet* level ($\sim$1-100s μs), which is the smallest unit a network can exert control. QoS control mechanisms operating at this time scale include *traffic conditioning devices* (e.g., traffic classifiers, markers, policers, and shapers), packet schedulers, and active queue management. The next fastest time scale is *round-trip-time* ($\sim$1-100s ms), at which scale feedback-based QoS control mechanisms such as congestion and flow control operate. Slower than packet time and round-trip-time is the *session* time scale (seconds, minutes or longer). This is the time scale user sessions (defined in whatever meaningful way) typically last, and at which QoS control mechanisms such as admission control and QoS routing operate. Beyond the session time scale, a variety of "long-term" QoS control mechanisms operate at time scales ranging from minutes, hours, to days, weeks, or months. Examples include traffic engineering, time-of-day service pricing, resource provisioning and capacity planning.

Another key aspect of QoS control is the *granularity* of control information (i.e., *control state*) used by a QoS mechanism in making control decisions and exerting control. The *finest* granularity of control is the *per-flow* state information (e.g., as identified by the 5-tuple – the IP source-destination addresses, port numbers and protocol field – carried in the IP header) which can be used to enforce QoS for individual user flows. Coarser-grain controls use information that is specified and maintained for an *aggregate* of user flows: the granularity of coarse-grain QoS controls varies depending on the level of *flow aggregation*, such as per host, per network prefix, per ingress-egress pair, per service class, etc. Closely related to the granularity of control are two other important aspects of QoS control – the *carrier* of control state, i.e., where the control state is stored, whether in routers or in the packet header only; and the *location* of control, i.e., where a control mechanism operates, whether at the end-hosts, the network *edge* or *boundaries* between either users and network or network domains, or inside the network *core*.

We can view the granularity of control, carrier of control state and location of control as forming the *space* dimension of QoS control, whereas the time scale the *time* dimension of QoS control. These two dimensions together define a broad design space from which QoS provisioning architectures can be built, reflecting various trade-offs in QoS service performance, operations and management complexity and implementation cost. For example, control granularity has a direct impact on the operations and management complexity of network data plane (i.e., the network elements such as routers that are directly involved in data packet forwarding) and per-packet processing cost of network elements. It also affects the QoS service performance individual users will experience. Time scale of control determines how frequently control information must be conveyed to network elements, thus affecting their processing, memory and communication overheads. Both the time and space dimensions of QoS controls have enormous implications in the design, operations and management of network control plane (which consists of network control entities such as routing processors, resource managers, service configuration modules that are not directly involved in user data forwarding, but are essential to the operations of a network). For example, a QoS provisioning architecture that employs per-flow QoS control and stores QoS state at every router requires a signaling protocol that conveys QoS states to every router on a per-flow basis. Such an architecture mandates a sophisticated control plane at every router,

complicating its operations and management and thus limiting its scalability. Hence to design a scalable and cost-effective QoS provisioning architecture, it is imperative to make judicious design choices along the time and space dimensions and carefully evaluate their trade-offs and implications in both network data and control planes.

The remainder of this paper is structured as follows. A recent series of development has shown that deterministic computations can be made more powerful with the use of a few simple theories, based on min and max calculus. Section II introduces the reader to these new developments, and illustrates their applications to integrated services, differentiated services, and playback delays. Section III explains the methods used for obtaining a scalable integrated services support, based on the concept of a stateless core. Section IV describes probabilistic results that can be used for a statistical dimensioning of differentiated services; some are based on classical queuing theory, while others capitalize on the deterministic results in Section II to obtain stochastic bounds. In a best-effort context, QoS differentiation and guarantees can be provided based on queue management, traffic conditioning and engineering, but need a considerable amount of network control information, and guarantees are average, approximate. Section V describes the recent theories and the conclusions that can be drawn. Section VI describes methods to provide some quality of service in a pure best-effort environment, without any access control. Section VII describes methods that allow an application to control the QoS it receives, in the absence of network support. Section VIII concludes the paper with a short list of challenges for the future.

II. Network Calculus, A Theory for the Deterministic Setting

Deterministic bounds on quantities such as loss and delay can be expressed if we combine constraints on traffic flows and service guarantees. The bounds depend on the nature of the schedulers, and may be very complex to derive [7], [8], [9], [10], [11], [12]; see also [13] for a review of packet scheduling. Many of these results can be cast into a common framework coined "network calculus", which we explain in this section.

In short, network calculus can be viewed as the application of $\min$ and $\max$ algebra to flow problems. It was pioneered by Chang [14] and Cruz [15], [16], and found its final form in subsequent work by the same authors and by Agrawal, Le Boudec and Rajan [17], [18], [19]. A comprehensive treatment can be found in two textbooks [5], [6]. We first introduce network calculus on an example, then we review applications to integrated services, differentiated services, and the computation of minimum playback delay for video sequences.

A. Introductory Example: The Shaper

Arrival Curves. Differentiated and integrated services assume that individual traffic flows are limited, for example using the concept of token (or "leaky") bucket. More generally, given some wide-sense increasing (= non-decreasing) function $\alpha(t)$, we say that a flow is α-smooth if the amount of data that can be observed on the flow over any time window of duration t is $\leq \alpha(t)$. We also say that α is an arrival curve for the flow. A token bucket, with rate r and burst b corresponds to $\alpha(t) = rt + b$; this is a common constraint imposed in traffic contracts between network and customer. Arrival curve constraints may also arise from physical limitations. Consider a flow that is known to arrive on a link of bit rate equal to C bits/second; if the flow is observed bit by bit, then we can say that it is α-smooth, with $\alpha(t) = Ct$. Consider the same flow, but now observed at the link layer receiver that terminates the link of bit rate C; here we observe entire packets instead of bits. If the packet size is M or less, then the flow is α-smooth, with $\alpha(t) = Ct + M$. Combining a token bucket constraint, imposed as part of a traffic contract, with a physical limitation, gives an arrival curve of the form $\alpha(t) = \min(Ct+M, rt+b)$, which is commonly used in the context of integrated services ("T-SPEC" [20]).

Shapers. Traffic generated by sources cannot be expected to naturally satisfy some a priori arrival curve constraint; a *shaper* is used to force a flow to satisfy some arrival curve constraint. Given some function $\sigma(t)$, a shaper stores incoming bits in a buffer and delivers them in such a way that the resulting output is σ-smooth. A shaper is *greedy* if it delivers the data as soon as possible. If $\sigma(t) = rt + b$, the greedy shaper can be implemented as a leaky bucket regulator, which simply monitors the level of a fictitious token bucket, represented by a single counter[15]. The spacer-controller used in ATM is also an example of shaper [21], [22].

Greedy shapers have a number of simple, physical properties; we focus here on one, the preservation of arrival constraints. Consider a flow, initially known to be α-smooth, which is passed into a shaper in order to be made σ-smooth. This example is commonplace; for example, σ is a token bucket constraint, and α is a constraint imposed by physical limitations or by an upstream shaper (Figure 1). A property of greedy shapers is that the shaper output still satisfies the original arrival curve constraint α, in other words [23] "what is done by shaping cannot be undone by shaping". Note that systems other than greedy shapers do not generally have this property. The preserva-

tion property was initially obtained by Cruz in [15] by an ad-hoc (complex) computation, valid for the specific case of leaky bucket controllers. In the sequel, we give a general result and show how it is obtained.

Fig. 1. Shapers preserve arrival constraints.

Min-Plus Convolution. We now introduce a network calculus formalism. We consider in this section only wide-sense increasing functions of time that are 0 for $t \leq 0$. For any two such functions $f(t)$ and $g(t)$ we define a third one $(f \otimes g)(t)$, called "min-plus convolution", by

$$(f \otimes g)(t) = \inf_{0 \leq s \leq t} (f(s) + g(t-s)) \qquad (1)$$

This operation is the analog of standard convolution, if we replace the two standard operations $+$ and $\times$ by $\min$ and $+$; min-plus algebra is the name of the calculus obtained with this mapping (see [24], [5] or [6] for a general presentation). The analogy bears some fruit – many properties of standard convolution, such as associativity and commutativity, are also true here: $(f \otimes g) \otimes h = f \otimes (g \otimes h) = f \otimes g \otimes h$ and $f \otimes g = g \otimes f$.

We characterize a flow with its cumulative function $R(t)$, defined as the number of bits observed from an arbitrary time origin up to time t. Then, saying that the flow is α-smooth is equivalent to $R \leq R \otimes \alpha$, which is also equivalent to $R = R \otimes \alpha$. This can be seen immediately by applying the definition of min-plus convolution. Consider now a shaper, which forces the flow into an arrival curve σ. We assume that σ is sub-additive, in other words, $\sigma(s+t) \leq \sigma(s) + \sigma(t)$. This is not a restriction, as any arrival curve constraint can be expressed with a sub-additive function [14]. In addition, all concave arrival curves (such as the arrival curves presented above) are sub-additive.

I/O Characterization of Shapers. Call R^* the output of the shaper. It must satisfy the constraints

$$\begin{cases} R^* \leq R \\ R^* \leq R^* \otimes \sigma \end{cases} \qquad (2)$$

The former inequality expresses that the output derives from the input after buffering; the latter expresses that it is σ-smooth. Any wide-sense increasing function $R^*(t)$ that satisfies (2) is the output of some shaper, not necessarily greedy. It turns out that the system (2) is a classical min-plus problem [25] and has one maximum solution, given by

$$R^* = R \otimes \sigma \qquad (3)$$

This statement can be proved in a general min-plus setting, but in this particular case, a direct proof is possible and holds in a few lines ([6], Section 1.5). The greedy shaper output is necessarily the maximum solution, which establishes that (3) is true for the shaper output. The first proof appeared in [17] and uses a different network calculus method than presented here.

Consequences. This establishes that shapers are min-plus linear systems. We show now how this implies the preservation property mentioned above. The associativity of min-plus convolution can be used:

$$R^* \otimes \alpha = (R \otimes \sigma) \otimes \alpha = (R \otimes \alpha) \otimes \sigma = R \otimes \sigma = R^*$$

The last but one equality is because the input is α-smooth and thus $R \otimes \alpha = R$. Thus, this establishes that $R^* = R^* \otimes \alpha$ as well, which means that the output of the shaper is α-smooth, as required (of course it is also σ-smooth as well).

Another consequence of the min-plus representation of shapers in Equation (3) is that a concatenation of I shapers in sequence with curves $\sigma_i, i = 1, ..., I$ is equivalent to a global shaper with curve $\sigma = \sigma_1 \otimes ... \otimes \sigma_I$. If the curves σ_i are concave, then $\sigma = \min_{1 \leq i \leq I} \sigma_i$. This is commonly used to implement shapers for concave piecewise linear functions as the concatenation of leaky bucket controllers. A striking fact is that the order of the concatenation does not play a role here.

Packetization Effects. The theory presented in this section ignores packetization constraints, which play a role when packets of a flow are of different sizes. Packetization effects are modeled with the concept of packetizer introduced in [26], [27], [28], which can be thought of as a device that collects bits until entire packets can be delivered. The results mentioned earlier remain valid, as long as the arrival curves are concave and have a jump at the origin at least as large as one maximum packet size [22]. Else, the insertion of a shaper weakens the arrival curve by one maximum packet size.

B. IntServ and Service Curves

The Principle of Reservations. The IETF Integrated Services (IntServ) architecture supports different reservation principles; we focus here on the *guaranteed* service [20], which provides deterministic guarantees (statistical guarantees are discussed in Section IV). IntServ uses *admission control*, which operates as follows.

- In order to receive the guaranteed or controlled load service, a flow must first perform a reservation during a flow setup phase.
- A flow must conform to an arrival curve of the form $\alpha(t) = \min(M + Ct, rt + b)$ (T-SPEC).

• All routers along the path accept if they are able to provide a service guarantee and enough buffer for loss-free operation. The service guarantee is expressed during the reservation phase, using the concept of service curve, as explained below.

Service Curves, a Min-Plus Approach. The reservation phase assumes that all routers can export their characteristics using a very simple model. The problem is that routers may implement very different scheduling strategies. This is solved with the concept of *service curve*. It was introduced by Parekh and Gallager [7] and Cruz [23] in a restricted sense, then independently in its final form by Agrawal, Chang, Cruz, Le Boudec, Okino and Rajan [19], [18], [17]. It is defined as follows. Consider a system $\mathcal{S}$ and a flow through $\mathcal{S}$ with input and output function R and R^*. Let $\beta(t)$ be a non-negative wide-sense increasing function We say that $\mathcal{S}$ offers to the flow a *service curve* β if and only if

$$R^* \geq R \otimes \beta \tag{4}$$

In practical terms, it means that for any time t, there exists a time $s \leq t$ such that

$$R^*(t) \geq R(s) + \beta(t-s) \tag{5}$$

This definition may seem obscure, but it turns out to be the right abstraction. First, it captures well the classical queuing systems, but also applies to complex systems. Consider a queue that serves a flow with a rate at least equal to c (for example, a generalized processor sharing node [7]); such a node offers a service curve equal to $\beta(t) = ct$ (for $t \geq 0$). More generally, a node that guarantees to serve at least $\beta(t)$ bits during any interval of duration t inside a busy period guarantees a service curve equal to the function $\beta(t)$; in that case we say that we have a *strict* service curve. In practice though, the concept of strict service curve does not mean much for a complex system, because there are delay elements. Consider for example a system about which we only know that the delay is bounded by some value T; assume that the input is a small but steady flow of data, at a rate ϵ; the system is always in a busy period; however, the output rate ϵ can be arbitrarily small, thus the only *strict* service curve we could express would be 0. In contrast, with the definition of service curve given above, this system offers a service curve $\beta = \delta_T$, defined by $\delta_T(t) = +\infty$ if $t > T$ and $\delta_T(t) = 0$ if $t \leq T$. In some sense, the service curve concept replaces the analysis by busy period which is commonplace in queuing theory, but does not apply to complex systems.

Second, the definition supports concatenation. Consider a tandem of two nodes, offering service curves β_1 and β_2, with the output of the first feeding the input of the second. It follows immediately from (4) and the associativity of min-plus convolution that the tandem, viewed as a single system, offers the service curve $\beta = \beta_1 \otimes \beta_2$. Thus, it is very easy to compute a service curve for complex nodes. For example, the min-plus convolution of[1] $\beta(t) = ct^+$ and $\delta_T(t)$ is equal to the so-called "rate-latency" function $\beta(t) = c(t-T)^+$, thus the concatenation of a node with guaranteed rate c and a node with maximum delay T offers a rate-latency service curve. IntServ requires that all routers can be abstracted with such a service curve [29] (or equivalently, as a guaranteed rate node, see below).

Third, the combination of arrival curve and service curve supports the derivation of the following tight bounds. Let a system offer a service curve β to a flow that is constrained by some arrival curve α. Then the backlog for this flow is bounded by the vertical deviation

$$v(\alpha, \beta) := \sup_{s \geq 0} [\alpha(s) - \beta(s)] \tag{6}$$

If the node serves the bits of this flow in FIFO order (an assumption that is true in the IntServ context), then the delay is bounded by the horizontal deviation (Figure 2)

$$h(\alpha, \beta) := \sup_{t \geq 0} [\inf \{d \geq 0 \text{ such that } f(t) \leq g(t+d)\}] \tag{7}$$

For a flow with arrival curve $\alpha(t) = \min(Ct + M, rt + b)$

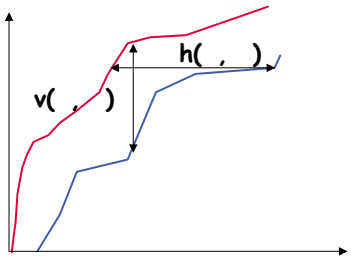


Fig. 2. Bounds on backlog and delay derived from arrival and service curves.

and a rate-latency service curve $\beta(t) = R(t-T)^+$, this gives the backlog bound [12], [30]

$$v(\alpha, \beta) = b + r \max\left(\frac{b-M}{C-r}, T\right)$$

and the delay bound

$$h(\alpha, \beta) = \frac{M + \frac{b-M}{C-r}(C-R)^+}{R} + T \tag{8}$$

End-to-end bounds. The above results can be combined to obtain the worst case end-to-end delay across an

[1] We use the notation $m^+ = \max(m, 0)$.

IntServ network. A flow that goes through a sequence of routers $i = 1, ..., I$, each with service curve $\beta_i(t) = R_i(t - T_i)^+$, sees the network as a system offering the service curve $\beta = \beta_1 \otimes ... \otimes \beta_I$. A direct computation shows that $\beta(t) = R(t-T)^+$ with $R = \min_i R_i$ and $T = \sum_i T_i$. Together with the delay bound (8), this is used by routers during the reservation setup phase, in order to determine if a reservation should be accepted [6].

By computing the end-to-end service curve as the min-plus convolution of the service curves of all nodes, it can also be established that the worst case delay over a concatenation of nodes is less than the sum of the worst case delay at every node. A similar statement is known under the term "pay bursts only once", which says that the impact of the burstiness parameter b in the arrival curve $\alpha(t) = \min(Ct + M, rt + b)$ of a flow does not accumulate over the number of nodes traversed by the flow, but, in contrast, occurs only once. This is a direct application of the results above ([6] Section 1.4.3).

Re-shaping is for Free. Another property which can be established with this abstract setting is "re-shaping is for free". Re-shaping is often introduced inside a network, or at network boundaries, in order to control the accumulation of burstiness that may otherwise occur. Assume now that a flow, constrained by an arrival curve α, is input to a tandem of networks, each offering service curves β_1, β_2 (Figure 3). Assume a greedy shaper, with curve $\sigma \geq \alpha$ is inserted between the two systems. The condition means that the re-shaper enforces some or all of the initial curve constraint. It follows directly from (3)

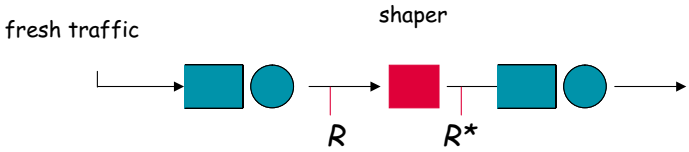


Fig. 3. Reshaping example.

that the re-shaper offers a service curve equal to σ. Thus, the worst case delay for the combination with re-shaper is $d' = h(\alpha, \beta_1 \otimes \sigma \otimes \beta_2)$ whereas for the original combination it is $d = h(\alpha, \beta_1 \otimes \beta_2)$. Now min-plus convolution is associative and commutative, thus $d' = h(\alpha, \sigma \otimes \beta_1 \otimes \beta_2)$; we interpret this as the worst case delay for a new combination where the re-shaper is put immediately before the first network, instead of between the two. But in that case, the input traffic is α-smooth, thus also σ-smooth, and the re-shaper never delays any bit of data. Thus we can remove the re-shaper from the new combination and $d' = d$. We have shown in these few lines that the delay bound for the system without shaper is also valid for the system with shaper. In other words, nodes may re-shape flows without exporting that information.

Other Aspects. The concepts of service and arrival curves have been used by Cruz and Sariowan [31], [32], Georgiadis, Guérin, Peris and Rajan [12] to design schedulers that optimize the combination of delay guarantees, buffer and bit rate requirements, and go beyond the initial design ideas of Kalmanek, Kanakia and Restrick [10] and H. Zhang and Ferrari [11]. Some of these schedulers are designed to have have service curves that are not rate-latency, therefore, their properties are not well exploited within the IntServ framework.

All computations so far were done with the assumption that the systems are empty at time 0. This is valid for static reservations, but not for dynamic reservations, which are supported by IntServ and ATM-ABR. The modifications to the calculus presented above were found by Giordano et al in [33].

Delay and Delay Jitter. For playback operations, only the variable part of delay, called delay jitter, is important (Section II-D). In contrast, for interactive services, the total delay is also of importance. Thus, both delays must be accounted for; this can be done as follows. If the latency terms of service curves do not incorporate constant delays, then delay bounds such as (8) give the delay jitter; a bound on total delay is then obtained by adding to it the sum of all constant delays.

Guaranteed Rate Servers, a Max-Plus Approach. The service curve concept defined above can be approached from the dual point of view, which consists in studying the packet arrival and departure times instead of the functions $R(t)$ (which count the bits arrived up to time t). This latter approach leads to max-plus algebra (which has the same properties as min-plus), is often more appropriate to account for details due to variable packet sizes, but works well only when the service curves are of the rate-latency type. It is used in Section III with the core stateless approach to obtain detailed relations between packet departure times across a network. It also useful when nodes cannot be assumed to be FIFO per flow, as may be the case with DiffServ (Section II-C). We now describe this approach here and how it relates to the min-plus approach.

A node is said to be of the Guaranteed Rate (GR) type [9] (also called Rate-Latency server), with rate r and latency e, if the departure time d_n of the nth packet, counted in order of arrival, satisfies

$$d_n \leq f_n + e \tag{9}$$

where f_n (virtual finish time) is given by the recursion (a_n is the arrival time, l_n the length in bits, of packet n):

$$\begin{cases} f_0 = 0 \\ f_n = \max\left[a_n, f_{n-1}\right] + \frac{l_n}{r} \text{ for } n \geq 1 \end{cases} \quad (10)$$

GR is an alternative way of describing the rate-latency service curve property. More precisely, a GR node with rate r and latency e can be decomposed as a node offering the rate-latency service curve $\beta(t) = r(t-e)^+$, followed by a packetizer [6]. Note that adding a packetizer weakens the service curve property by one maximum packet size, but does not increase the packet delay. Conversely, but only for a FIFO node, the rate-latency service curve $\beta(t) = r(t-e)^+$ implies GR with rate r and latency e. It follows from this equivalence that the delay bounds in Equation (8) hold for a FIFO GR node; it is shown in [34] that it also holds for non FIFO nodes. Specifically, the packet delay for a flow that is α-smooth is bounded by

$$\sup_{t>0}\left[\frac{\alpha(t)}{r} - t + e\right] \quad (11)$$

For GR nodes that are FIFO per flow, the concatenation result obtained with the service curve approach applies. Specifically, the concatenation of I GR nodes (that are FIFO per flow) with rates r_i and latencies e_i is GR with rate $r = \min_i r_i$ and latency $e = \sum_i e_i + (I-1)\frac{L_{\max}}{r}$, where $L_{\max}$ is the maximum packet size for the flow. The term $(I-1)\frac{L_{\max}}{r}$ is due to packetizers. For GR nodes that are not FIFO per flow, this result is no longer true [34].

The recursion in (10) can be solved easily, using the properties of max-plus algebra. We obtain that GR is equivalent to saying that for all n there is some $k \in \{j+1, ..., n\}$ such that

$$d_n \leq a_k + \frac{l_k + ... + l_n}{r} + e \quad (12)$$

which is the dual of (5) with $\beta(t) = r(t-e)^+$ [5].

C. DiffServ, Aggregate Scheduling and Adaptive Service Curves

The IETF Differentiated Services (DiffServ) architecture differs from the IntServ architecture in that flows are treated in an aggregate manner inside a network. DiffServ is a framework which supports many services; we focus here on Expedited Forwarding (EF)[2]. Roughly speaking, EF can be thought of as a priority service. Packets marked as EF (namely, with the "PHB" field in the IP header set to "EF") receive a low delay and practically loss-free service. This is typically used for circuit emulation or high quality videoconferencing.

Expedited Forwarding and Intuition Behind. At every router, all EF packets are viewed as one single aggregate. In contrast, at network access points, individual flows of EF packets (called "microflows") are shaped one-by-one, according to an arrival curve similar to the T-SPEC defined in Section II-B. As with IntServ, the arrival curves constraints put on micro-flows are expected to support hard end-to-end quality of service guarantees. Unlike IntServ though, microflows are not scheduled separately. The intuition is that, as long as the intensity of EF traffic is small, EF queues remain empty and delays are small delays remain small.

More precisely, the original description of EF in [35] was implicitly assuming that sources are, in the worst case, periodic (this is now dropped from the formal definition of EF). Then, if the EF traffic intensity is small, it is plausible that the delay variation for packets inside one microflow is less than the period of the source. As a result, packets from the same microflow would never catch up and the service would be simple to analyze and use. Chlamtac et al [36], [37], [38] have shown that this intuition does hold, but in an ATM context, under the assumption that sources satisfy the "source rate conditions", which require that the period of a source (in time slots) be at least as large as its route interference number. The route interference number is the number of times when the path of a given source merges with that of other sources. However, it is difficult to transpose this result from ATM to Internet, first because of variable packet sizes, and second because the FIFO assumption may be too strict.

PSRG, Formal Definition of EF. Thus, the current definition of EF is not based on this result. In contrast, it is based on an abstract node model, inspired by GPS [7], called "Packet Scale Rate Guarantee", from which a delay bound can be obtained. This is analog to IntServ assuming that every router can be modeled a GR node, but with some differences, to which we come back later in this section. A node is said to offer to a flow of packets (here: the EF aggregate) the packet scale rate guarantee [39] with rate r and latency e if the departure time d_n of the nth packet, counted in order of arrival, satisfies (9) where f_n is given by the following recursion:

$$\begin{cases} f_0 = 0 \\ f_n = \max\left[a_n, \min\left(d_{n-1}, f_{n-1}\right)\right] + \frac{l_n}{r} \text{ for } n \geq 1 \end{cases} \quad (13)$$

(a_n is the arrival time , l_n the length in bits, of packet n).

[2]DiffServ makes a distinction between *service* and *Per-Hop Behavior* (PHB). In classical, OSI, terminology, the former means the service provided by a network, and the latter is the service provided by a network element.

A non-preemptive priority scheduler with rate r satisfies the definition, with $e \times r$ equal to the maximum size of low priority packets; as explained later PSRG applies to more complex nodes, possibly non-FIFO. PSRG differs from GR defined in Section II-B by the d_{n-1} term in (13). It follows that PSRG is stronger than GR, i.e., any PSRG node satisfies the GR property with the same parameters. We will use these properties now to obtain an end-to-end delay bound.

End-to-end Delay Bound for EF. Charny and Le Boudec have obtained in [40] a bound on delay variation that is valid for EF, as follows. Assume that microflow i is constrained by the arrival curve $\rho_i t + \sigma_i$ at the network access. Inside the network, EF microflows are *not* shaped. At node m, the EF aggregate is served according to the packet scale rate guarantee, with rate r_m and latency e_m (Figure 4)). Call H a bound on the number of hops used

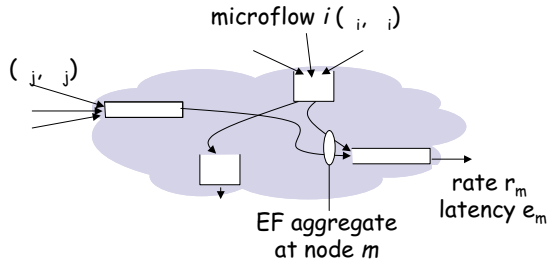


Fig. 4. Model of EF network

by any flow (this is typically 10 or less, and is much less than the number of nodes). Call D a bound on the queuing delay undergone by a flow at any single node (assuming a finite bound exists, which is shown in [40]), and consider some arbitrary node m. The data that feeds node m has undergone a variable delay in the range $[0, (H-1)D]$, thus an arrival curve for the EF aggregate at node m is $\nu r_m(t + (H-1)D) + r_m\tau$, where ν (maximum utilization factor) is a bound[3] on $\frac{1}{r_m}\sum_{i \ni m} \rho_i$ and τ (maximum packet delay variation) is a bound on $\frac{1}{r_m}\sum_{i \ni m} \sigma_i)$. By application of (11), the delay seen by any packet is bounded by $D \leq e + \tau + (H-1)D\nu$; thus if the utilization factor ν is less than $\frac{1}{H-1}$, we have the following bound on delay at one hop

$$D \leq \frac{e + \tau}{1 - (H-1)\nu} \tag{14}$$

The bound can be improved if we have more information about the peak rate at which the EF aggregate may arrive at the node [40].

The bound is valid only for small utilization factors; it explodes at $\nu > \frac{1}{H-1}$, which does not mean that the worst case delay does grow to infinity [41]. As far as we know, this issue is still unresolved ([6] Section 6.3). However, it is shown in [39] that any better bound must make more assumptions about the network than is suitable in the EF framework. See also Section IV-B for statistical bounds that are valid under the same setting.

PSRG versus Service Curve – Delay from Backlog. Why do we need for EF a definition such as Packet Scale Rate Guarantee, instead of using for example the service curve or GR characterization of IntServ? Indeed, a GR definition might be a valid node abstraction for EF, since the delay bound mentioned above used only the GR property. The reason for choosing PSRG instead is based on the desire to have a delay-from-backlog bound, which is used in cases with statistical multiplexing.

Indeed, GR (and service curve guarantee) may give birth to the "lazy scheduler" syndrome, which consists in that it is perfectly valid for a GR scheduler to serve the first k packets of a flow faster than necessary, and then take advantage of this advance to delay subsequent packets for an arbitrarily long amount of time [13]. As a result, it is not possible to derive a bound on the delay undergone by a packet from the backlog it sees upon arrival, unlike the case of an ideal GPS scheduler [6]. In contrast, with PSRG, the effect of the d_{n-1} term is that, if a packet is served earlier than its deadline, then the deadline of all subsequent packets is reduced accordingly. The following delay-from-backlog bound is shown in [34]: for a packet served in a PSRG node, that sees a backlog equal to Q upon arrival, the delay is bounded by $\frac{Q}{r} + e$.

With IntServ, it is natural to assume that a node serves the packets inside a flow in FIFO order. This per-flow FIFO assumption cannot usually be made with DiffServ. Indeed, with DiffServ, a scheduler sees an entire EF aggregate as one flow. Since the EF aggregate usually enters a router via more than one input ports, the delay though the router internal may vary a lot across packets, and as a result, the node may not be globally FIFO. All bounds on delay mentioned above are true for PSRG nodes, even non FIFO [34].

For the special case of FIFO nodes, PSRG is equivalent to the *adaptive service curve* property, a variant of the service curve property defined by Agrawal, Cruz, Okino and Rajan [42]. Concatenation rules based on min-plus convolution apply here also, but they do not extend to non-FIFO nodes.

Min-Max Algebra. We have explained in Section II-B how service curves and the IntServ framework are based on min-plus and max-plus algebras. For DiffServ, min-max algebra has to be invoked also to derive properties of PSRG, as we explain now. The iterative definition of f_n in

[3] the notation $i \ni m$ means that node m is on the path of microflow i

(13) can be re-written as a min-max equation:

$$f_n = \left[a_n + \frac{l_n}{r}\right] \vee \left[\left(d_{n-1} + \frac{l_n}{r}\right) \wedge \left(f_{n-1} + \frac{l_n}{r}\right)\right] \quad (15)$$

Now min-max algebra enjoys the same properties as min-plus algebra; this is used in [34] to show that PSRG is equivalent to saying that for all n and all $0 \leq j \leq n-1$ either

$$d_n \leq e + d_j + \frac{l_{j+1} + ... + l_n}{r} \quad (16)$$

or there is some $k \in \{j+1, ..., n\}$ such that

$$d_n \leq e + a_k + \frac{l_k + ... + l_n}{r} \quad (17)$$

Equations (16) and (17) constitute a characterization of PSRG without the virtual finish times; they are the key relations from which all properties of PSRG mentioned earlier in this section [34] are derived.

Low Jitter Alternatives to EF. A number of research proposals aim to obtain better bounds than (14), at the expense of more elaborate schedulers, while preserving aggregate scheduling. A first proposal uses the concept of damper [43], [44], which has the effect of compensating delay variation at one node in the next downstream node. With dampers applied to the EF aggregate, the end-to-end delay bound becomes much smaller and is finite for all utilization factors less than 1 [6].

A simpler and more powerful alternative is proposed by Z.-L. Zhang et al under the name of Static Earliest Time First (SETF) [45]. Assume that packets are stamped with their time of arrival at the network access, and that they are served within the EF aggregate at one node in order of time stamps. More precisely, we assume that nodes offer a GR guarantee to the EF aggregate, as defined by (10) or (12), but that packets are numbered in order of their arrival at the network access (not at this node). Then the analysis that led to the end-to-end delay bound (14) can be modified as follows. Call D_h a bound on the end-to-end delay after h hops, $h \leq H-1$. Consider a tagged packet, with label n, and call d_h its delay in h hops. Consider the node m that is the hth hop for this packet. Apply (12): there is some label $k \leq n$ such that

$$d_n \leq e + a_k + \frac{l_k + ... + l_n}{r} \quad (18)$$

where a_j and d_j are the arrival and departure times at node m of the packet labeled j, and l_j its length in bits. Now packets k to n must have arrived at the network access before $a_n - d_h$ and after $a_k - D_{H-1}$. Thus

$$l_k + ... + l_n \leq \alpha(a_n - a_k - d_h + D_{H-1})$$

where α is an arrival curve at network access for the traffic that will flow through node m. We have $\alpha(t) \leq r_m(\nu t + \tau)$. By (11), the delay $d_n - a_n$ for our tagged packet is bounded by

$$e + \sup_{t \geq 0}\left[\frac{\alpha(t - d_h + D_{H-1})}{r_m} - t\right] = e + \tau + \nu(D_{H-1} - d_h)$$

thus

$$d_{h+1} \leq d_h + e + \tau + \nu(D_{H-1} - d_h)$$

The above inequality can be solved iteratively for d_h as a function of D_{H-1}; then take $h = H-1$ and assume the tagged packet is one that achieves the worst case h-hop delay, thus $D_{H-1} = d_{H-1}$ which gives an inequality for D_{H-1}; last, take $h = H$ and obtain the end-to-end delay bound

$$D_H \leq (e + \tau)\frac{1 - (1-\nu)^H}{\nu(1-\nu)^{H-1}} \quad (19)$$

The bound is finite for all values of the utilization factor $\nu < 1$, unlike the end-to-end bound in (14). Note that for small values of ν, the two bounds are equivalent.

We have assumed here infinite precision about the arrival time stamped in every packet. In practice, the timestamp is written with some finite precision; in that case, Zhang [45] finds a bound which lies between (14) and (19) (at the limit, with null precision, the bound is exactly (14)).

D. Playback Delay for Pre-Recorded Video

Consider a client reading a pre-recorded video file from a server across a network. Assume the network guarantees a bound on variable delay T but requires the flow to be σ-smooth (these assumptions correspond to sending the video over EF; a similar example is studied in [46] but with IntServ instead of DiffServ). On the client side, the flow is processed with high priority before being sent to the display; this is modeled by assuming that the flow receives a rate-latency service curve, with a rate equal to the processing rate, and a latency accounting for the maximum interruption [47]. It follows that the combination of network delay and processor delay at the client side can be modeled with a service curve, say $\beta(t)$. Before being sent into the network, the flow is processed by a smoother in order to be made conformant to the arrival curve constraint σ; the smoother is similar to the shaper described in Section II-A, except that since the file is pre-recorded, it may send bits in advance of their natural reading time (in other words, it does not have to be causal).

Once processed at destination, the flow is played back into a decoding buffer which has to re-create the original timing of the flow. We assume that this is done by delaying the first packet of data for some amount D called

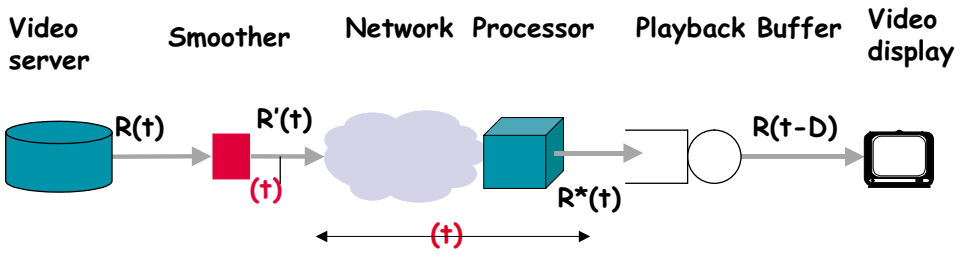


Fig. 5. Playing a video file over a network.

the playback delay. If the arrival curve constraint is very large, then there is no need for smoothing and the decoding buffer need only compensate for the delay jitter due to network and client processor; here, it is necessary and sufficient for D to be an upper bound on delay jitter. Otherwise, in the general case, smoothing is necessary and the decoding buffer needs to compensate for both delay jitter and the timing difference due to smoothing (Figure 5). Call $R(t)$ the number of bits of the original flow, when it is read in real time (the rate of which is not assumed to be constant). A lazy smoother will simply delay $R(t)$, like a shaper would do; a more aggressive smoother may anticipate bursts in $R(t)$ and thus obtain a smaller playback delay (e.g., using prefetch smoothing [48]).

We are interested in finding the minimum playback delay D that can be achieved, given σ and β, among all smoothing strategies. Rexford and Towsley [49] find the solution when the network service is constant bit rate, which corresponds to $\sigma(t) = \beta(t) = rt$ for some r. Le Boudec and Verscheure find the solution in the general case, by modeling the problem with a set of inequalities and apply the same method mentioned with Equation (2) in Section (II-A). They find in [46] that the minimum playback delay is given by the horizontal deviation:

$$\overline{D} = h(R, \sigma \otimes \beta) \tag{20}$$

Figure 6 illustrates the formula. [46] also find an optimal smoother output (one that achieves the minimum playback delay $\overline{D}$) and obtain an explicit representation using min-plus operations. A number of applications follow from this representation. First, the optimal smoother is not a shaper; indeed, a shaper smooths out bursts in $R(t)$ once they occur, whereas in most cases, the optimal smoother has to pre-fetch the bursts. Second, the strategy that would consist in equalizing delay jitter before presenting data to the decoder buffer is not optimal because of a "pay bursts only once" syndrome. Third, the optimal smoother output is anti-causal, in other words, the optimal time at which frame n should be sent depends only on the sizes of frames $m \geq n$. Thus the production of small playback delays is based on the ability to look-ahead in the stored video file. This is used in [50] to construct the encoding $R(t)$ which minimizes distortion, given σ, β and a target playback delay $\overline{D}$. Extension of optimal video smoothing to a multicast environment (with application-level QoS mechanisms, see Section VII) can be found in [51].

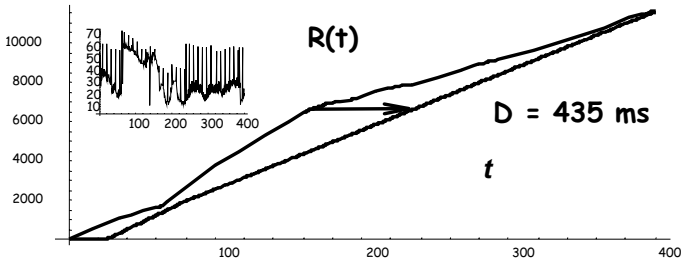
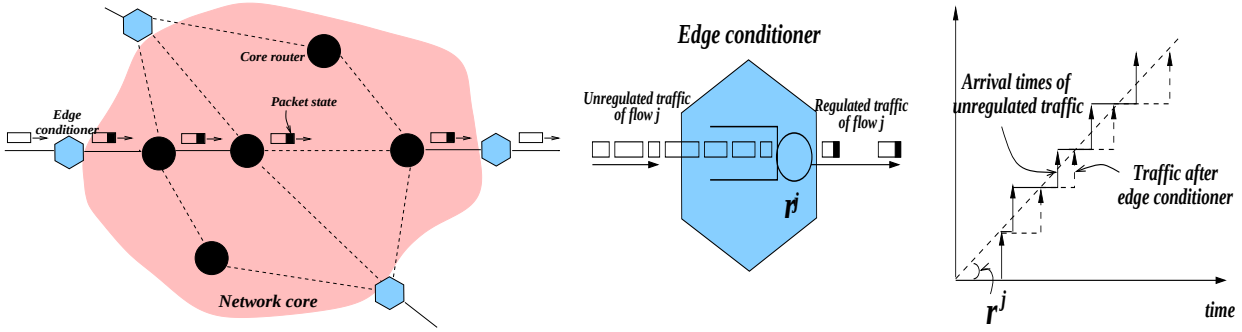


Fig. 6. Computation of minimum playback delay for an MPEG sequence. The top left box shows $r(t)$, the number of bytes for the tth frame. $R(t) = \sum_{s=1}^{t} r(t)$ is the corresponding cumulative function. The minimum playback delay, shown by the arrow, is the horizontal deviation between $R(t)$ and $(\sigma \otimes \beta)(t)$.

III. Architectures for Scalable QoS Support

Scalability is a key issue in the design of Internet QoS provisioning architectures, in both *data plane* and *control plane*. In network data plane appropriate control state information is needed for *per-packet* processing such as scheduling and queue management at core routers so as to support differentiated packet treatment and provide QoS guarantees. Granularity of such control state information and how it is obtained and maintained determine the complexity of QoS state management in data plane, and thus its scalability. Likewise, appropriate control state information is also needed in network control plane for resource reservation and QoS provisioning. Complexity and scalability of control plane operations depend critically on the granularity and time scale of such control state information.

In addressing the scalability issues in data plane, class-based aggregate scheduling is an important approach, as is adopted in DiffServ. However, as we have seen earlier,



(a) A conceptual network model.

(b) Edge conditioner and its effect

Fig. 7. Edge conditioning in the virtual time reference system.

this increased scalability is achieved at the expense of reduced performance, at least in terms of worst-case end-to-end delay performance. Another attractive approach is the *dynamic packet state* approach [52], where control state information necessary for packet scheduling is carried in packet headers; core routers perform simple per-packet state update. As a result, using the dynamic packet state approach, per-flow end-to-end QoS guarantees similar to those provided by IntServ can be supported without per-flow management at core routers. In section III-A we will provide an overview of the *virtual time reference system* – a unifying scheduling framework to provide scalable support for guaranteed services based on the dynamic packet state approach [53].

To reduce the complexity and thus enhance the scalability of control plane operations, a number of new approaches have been developed. They can be roughly categorized into three general approaches: *lightweight signaling, end-point/edge admission control* and *"centralized" bandwidth broker*. In section III-B we will briefly describe some representative examples of these three different approaches.

A. Dynamic Packet State and Virtual Time Reference System

The notion of dynamic packet state was first proposed by Stoica and Zhang [54], [55], [52], where control state information is carried in data packets and updated at core routers for scheduling purposes. In [55] Stoica and Zhang demonstrated that a *core stateless* version of Jitter Virtual Clock (Jitter-VC) can be implemented using the dynamic packet state technique to attain the same end-to-end delay bound without per-flow management. Their scheme was generalized by Zhang et al in [53], where using the dynamic packet state approach, a *general* core stateless framework – the virtual time reference system (VTRS) – was developed to provide scalable support for guaranteed services. The key construct in the virtual time reference system is the notion of *packet virtual time stamps*, which are referenced and updated as packets traverse each core router. As we will see shortly, the virtual time stamps associated with packets of a flow form the *thread* that "weaves" together the per-hop behaviors of core routers along the path of the flow to provide QoS guarantees for the flow. A key property of packet virtual time stamps is that they can be computed using solely the packet state carried by packets (plus a couple of fixed parameters associated with core routers). In this sense, the virtual time reference system is *core stateless*, as no per-flow state is needed at core routers for computing packet virtual time stamps.

Conceptually, the virtual time reference system consists of three logical components: *packet state* carried by packets, *edge traffic conditioning* at the network edge (see Figure 7), and *per-hop virtual time reference/update mechanism* at core routers (see Figure 8). These three components are briefly described below.

Edge Traffic Conditioning. Edge traffic conditioning plays a key role in the VTRS, as it ensures that packets of a flow[4] will never be injected into the network core at a rate exceeding its reserved rate (see Figure 7(b)). Formally, for a flow j with a reserved rate r^j, the inter-arrival time of two consecutive packets of the flow at the first hop core router is such that $\hat{a}_1^{j,k+1} - \hat{a}_1^{j,k} \geq \frac{L^{j,k+1}}{r^j}$, where $\hat{a}_1^{j,k}$ denotes the arrival time of the kth packet $p^{j,k}$ of flow j at the network core, $L^{j,k}$ the size of packet $p^{j,k}$, and r^j the reserved rate of flow j. This is equivalent to passing the flow through a shaper with $\sigma(t) = r_j t$, followed by a packetizer.

Packet State. After going through the edge conditioner

[4]Here a flow can be either an individual user flow, or an aggregate traffic flow of multiple user flows, defined in any appropriate fashion.

at the network edge, packets entering the network core carry, in their packet headers, certain packet state information that is initialized and inserted at the network edge. The packet state carried by the kth packet $p^{j,k}$ of a flow j contains three types of information: 1) QoS reservation (a rate-delay parameter pair $\langle r^j, d^j \rangle$) of the flow; 2) the virtual time stamp $\tilde{\omega}_i^{j,k}$ of the packet that is associated with the router i currently being traversed; and 3) the virtual time adjustment term $\delta^{j,k}$ of the packet. At the network edge, the rate-delay parameter pair $\langle r^j, d^j \rangle$, which is determined by a bandwidth broker (see Section III-B) based on flow j's QoS requirement, is inserted into every packet of the flow. For the kth packet of flow j, its virtual time stamp $\tilde{\omega}_1^{j,k}$ is initialized to $\hat{a}_1^{j,k}$, the actual time it leaves the edge conditioner and enters the first core router along the flow's path. The virtual time adjustment term $\delta^{j,k}$ for packet $p^{j,k}$ is set to $\Delta^{j,k}/q$, where q is the number of rate-based schedulers (shall be defined shortly) employed by the routers along the flow's path, and $\Delta^{j,k}$ is the cumulative delay experienced by packet $p^{j,k}$ in an *ideal dedicated per-flow system*, where packets of flow j are serviced by q tandem servers with constant rate r^j.

The iterative computation of $\Delta^{j,k}$ is a key result on which the method is based. Call $f_i^{j,k}$ the departure time of packet k of flow j from the ith ideal server; similar to (10) in Section II-B, we have (propagation delays are removed):

$$f_i^{j,k} = \max\{f_{i-1}^{j,k}, f_i^{j,k-1}\} + \frac{L^{j,k}}{r^j}.$$

This max-plus relation is used in [53] to show that

$$f_i^{j,k} = \max\{\hat{a}_1^{j,k} + i\frac{L^{j,k}}{r^j}, f_i^{j,k-1} + \frac{L^{j,k}}{r^j}\},$$

from which $\Delta^{j,k} = f_q^{j,k} - \hat{a}_1^{j,k} - q\frac{L^{j,k}}{r^j}$ can be computed at the network edge using the following recursive formula: $\Delta^{j,1} = 0$ and for $k \geq 2$:

$$\Delta^{j,k} = \max\{0, \Delta^{j,k-1} + q\frac{L^{j,k-1} - L^{j,k}}{r^j} + \hat{a}_1^{j,k-1} - \hat{a}_1^{j,k} + \frac{L^{j,k}}{r^j}\}$$

Virtual Time Reference/Update Mechanism and Per-Hop Router Behavior Characterization. In the *conceptual* framework of the virtual time reference system, each core router is equipped with a per-hop virtual time reference/update mechanism to maintain the continual progression of the *virtual time* embodied by the packet virtual time stamps. This virtual time stamp $\tilde{\omega}_i^{j,k}$ represents the arrival time of the kth packet $p^{j,k}$ of flow j at the ith core router *in the virtual time*, and thus it is also referred to as the *virtual arrival time* of the packet at the core router. The virtual time stamps, $\tilde{\omega}_i^{j,k}$, associated with packets of flow j satisfy the following two important properties: 1) *virtual spacing property:* $\tilde{\omega}_i^{j,k+1} - \tilde{\omega}_i^{j,k} \geq \frac{L^{j,k+1}}{r^j}$, and 2) the *reality check property:* $\hat{a}_i^{j,k} \leq \tilde{\omega}_i^{j,k}$, where $\hat{a}_i^{j,k}$ denotes the actual arrival time of packet $p^{j,k}$ at router i. These two properties are important in ensuring that the end-to-end delay experienced by packets of a flow across the network core is bounded.

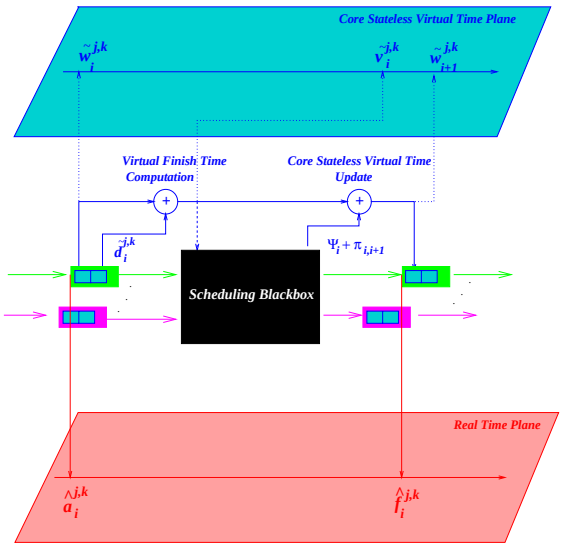


Fig. 8. Virtual time reference system: per-hop behavior and operations.

In order to ensure that these two properties are satisfied, the virtual time stamps must be appropriately referenced or updated as packets enter or leave a core router. The referencing/updating rule depends on the scheduling algorithm (or *scheduler*) employed by a core router and its characteristics. We distinguish two types of schedulers: rate-based and delay-based, depending on how the *virtual delay* parameter and *virtual finish time* are computed for packets traversing it. For example, if the scheduler $\mathcal{S}_i$ at the ith router is rate-based, packet $p^{j,k}$ is associated with the virtual delay parameter $\tilde{d}_i^{j,k} = L^{j,k}/r^j + \delta^{j,k}$ and its virtual finish time is defined as $\tilde{\nu}_i^{j,k} = \tilde{\omega}_i^{j,k} + \tilde{d}_i^{j,k}$. Whereas, if $\mathcal{S}_i$ is delay-based, $p^{j,k}$ is associated with the virtual delay parameter $\tilde{d}_i^{j,k} = d^j$ and its virtual finish time is again defined as $\tilde{\nu}_i^{j,k} = \tilde{\omega}_i^{j,k} + \tilde{d}_i^{j,k}$.

The *per-hop behavior* of a core router (or rather, its scheduler) is characterized by an *error term*, which is defined with respect to the virtual finish time and *actual* finish time of packets at the router. Let $\hat{f}_i^{j,k}$ denote the actual time packet $p^{j,k}$ departs the scheduler $\mathcal{S}_i$. We say that $\mathcal{S}_i$ can guarantee flow j its reserved rate r^j (if $\mathcal{S}_i$ is rate-based) or its delay parameter d^j (if $\mathcal{S}_i$ is delay-based) with an error term Ψ_i, if for any k, $\hat{f}_i^{j,k} \leq \tilde{\nu}_i^{j,k} + \Psi_i$. In other

words, each packet of flow j is guaranteed to depart $\mathcal{S}_i$ by the time $\tilde{\nu}_i^{j,k} + \Psi_i = \tilde{\omega}_i^{j,k} + \tilde{d}_i^{j,k} + \Psi_i$. This amounts to saying that $\mathcal{S}_i$ is a GR node (Section II-B) with rate r^j and latency Ψ_i if $\mathcal{S}_i$ is rate-based or with infinite rate and latency $d^j + \Psi_i$ if if $\mathcal{S}_i$ is delay-based.

Given the error term Ψ_i of the scheduler $\mathcal{S}_i$, the virtual time stamp of packet $p^{j,k}$ after it has traversed $\mathcal{S}_i$ is updated using the following reference/update rule:

$$\tilde{\omega}_{i+1}^{j,k} = \tilde{\nu}_i^{j,k} + \Psi_i + \pi_i = \tilde{\omega}_i^{j,k} + \tilde{d}_i^{j,k} + \Psi_i + \pi_i \quad (21)$$

where π_i denotes the propagation delay from the ith router to the next-hop router along the flow's path. In [53] it is shown that using the reference/update rule in (21) the virtual spacing and reality check properties of virtual time stamps are satisfied at every router.

End-to-end Delay Bounds and QoS Abstraction of Data Plane. An important consequence of the virtual time reference system outlined above is that the end-to-end delay bound on the delay experienced by packets of a flow across the network core can be expressed in terms of the rate-delay parameter pair of a flow and the error terms of the routers along the flow's path. Suppose there are total h hops along the path of flow j, of which q routers employ rate-based schedulers, and $h - q$ delay-based schedulers. Then for each packet $p^{j,k}$ of flow j, we have

$$\hat{f}_h^{j,k} - \hat{a}_1^{j,k} \leq d_{core}^j = q\frac{L^{j,max}}{r^j} + (h-q)d^j + \sum_{i=1}^{h}\Psi_i + \sum_{i=1}^{h-1}\pi_i,$$

where $L^{j,max}$ is the maximum packet size of flow j. This can be obtained by applying (11), where $r = r^j$, e is the sum of all latencies and $\alpha(t) = r^j t + L^{j,max}$ is the arrival curve that applies to the traffic entering the core (it is the output of a shaper with shaping curve $r^j t$ followed by a packetizer, see Section II-A).

Suppose the traffic profile of flow j is specified using the standard dual-token bucket regulator $(\sigma^j, \rho^j, P^j, L^{j,max})$ where $\sigma^j \geq L^{j,max}$ is the maximum burst size of flow j, ρ^j is the sustained rate of flow j, P^j is the peak rate of flow j. Then the maximum delay packets of flow j experienced at the edge shaper is also given by (11), where $\alpha(t) = \max\{\rho^j t + \sigma^j, P^j t + L^{j,max}\}$, $r = r^j$ and $e = 0$ (edge traffic conditioning is a GR server with 0 latency). This gives

$$d_{edge}^j = T_{on}^j \frac{P^j - r^j}{r^j} + \frac{L^{j,max}}{r^j},$$

where $T_{on}^j = (\sigma^j - L^{j,max})/(P^j - \rho^j)$ is the maximum duration that flow j can inject traffic at its peak rate into the network (here the edge traffic conditioner). Hence the end-to-end delay bound for flow j is given by

$$d_{end-to-end}^j = d_{edge}^j + d_{core}^j = T_{on}^j \frac{P^j - r^j}{r^j} + (q+1)\frac{L^{j,max}}{r^j} + (h-q)d^j + \sum_{i=1}^{h}\Psi_i + \sum_{i=1}^{h-1}\pi_i.$$

Observe that the end-to-end delay formula fits in the IETF Guaranteed Service framework. In this sense, the virtual time reference system provides a conceptual *core stateless* framework based on which guaranteed services can be implemented in a scalable manner using the DiffServ paradigm. Under this framework, per-hop behavior (i.e., its ability to support delay guarantees) of a core router is characterized using the notion of error term. This simple abstraction enables us to derive *end-to-end delay bounds* for flows traversing an arbitrary concatenation of core routers and their scheduling mechanisms.

Core Stateless Packet Scheduling. The virtual time reference system does not mandate any *specific* scheduling mechanisms to be implemented in a network domain as long as their abilities to provide delay guarantees can be characterized using the notion of error term. In fact, in [53] it is shown that almost all known scheduling algorithms can thus be characterized, be they *core stateless* or *stateful*. In addition, the virtual time reference system leads to the design of a set of new core stateless scheduling algorithms (both rate-based and delay-based). Two representative examples of such core stateless scheduling algorithms are: the rate-based *core stateless virtual clock* ($C_\$$VC) and delay-based *virtual time earliest deadline first* (VT-EDF) scheduling algorithms.

The core stateless virtual clock ($C_\$$VC) is a work-conserving counterpart of the CJVC scheduling algorithm developed in [55]. It services packets in the order of their virtual finish times, where as defined before, the virtual finish time of packet $p^{j,k}$ is given by $\tilde{\nu}^{j,k} = \tilde{\omega}^{j,k} + L^{j,k}/r^j + \delta^{j,k}$. It is shown in [53] that as long as the total reserved rate of flows traversing a $C_\$$VC scheduler does not exceed its capacity (i.e., $\sum_j r^j \leq C$), then the $C_\$$VC scheduler can guarantee each flow its reserved rate r^j with the minimum error term $\Psi = L^{*,max}/C$, where $L^{*,max}$ is the largest packet size among all flows traversing the $C_\$$VC scheduler.

Unlike the conventional rate-controlled EDF, VT-EDF supports delay guarantees without per-flow rate control, and thus is core stateless. It services packets in the order of their virtual finish times, where as defined before, the virtual finish time of packet $p^{j,k}$ is given by $\tilde{\nu}^{j,k} = \tilde{\omega}^{j,k} + \tilde{d}^j$. It is shown in [53] that the VT-EDF scheduler can guarantee each flow its delay parameter d^j with the minimum error term $\Psi = L^{*,max}/C$, if the following schedulability

condition is satisfied:

$$\sum_{j=1}^{N}[r^j(t-d^j)+L^{j,max}]1_{\{t\geq d^j\}} \leq Ct, \forall t \geq 0,$$

where we assume that there are N flows traversing the VT-EDF scheduler with $0 \leq d^1 \leq d^2 \leq \cdots \leq d^N$. The indicator function $1_{\{t\geq d^j\}} = 1$ if $t \geq d^j$, 0 otherwise.

Lastly, we have seen in Section II-C that the dynamic packet state technique and aggregation were combined by Zhang et al [56] to design a new aggregate packet scheduling algorithm called SETF. Control information is encoded (using a *finite* number of bits) in the packet header for scheduling purpose: packets are stamped with their entry time at the network edge, and they are scheduled in the order of their (network entry) time stamps at a router. In [56] another class of aggregate packet scheduling called *dynamic earliest time first* (DETF) is also defined. It differs from SETF in that packet time stamps may be modified at certain routers as packets traverse them. Using SETF and DETF as well as the simple FIFO, the authors demonstrate the fundamental trade-offs between granularity of control information and achievable network performance in terms of providing deterministic QoS guarantees.

B. Scalable Control Plane Operations

Control plane is an integral part of any QoS provisioning architecture, as support for performance guarantees requires control and management of network resources. Complexity and scalability of control plane operations such as resource management and signaling are closely tied to the data plane architecture as well as the desired QoS provisioning objectives. For example, in the IntServ architecture, per-flow scheduling architecture is used to support fine-grain bandwidth and delay guarantees for individual user flows. Consequently, a signaling protocol, RSVP [57], is designed to convey per-flow resource reservation information to core routers, which in turn need to perform per-flow resource reservation management, thus limiting the scalability of the IntServ architecture. In the DiffServ architecture, as aggregate scheduling is used at core routers to support class of services, a variety of more scalable, and perhaps less complex, approaches to resource management and provisioning are possible. Similarly, the dynamic packet state architecture also allows for more scalable and flexible QoS control plane to be developed. In the following we briefly discuss a number of representative approaches to scalable QoS control plane operations.

Lightweight Signaling. The lightweight signaling approach in general still requires a signaling protocol that conveys resource reservation to core routers. However thanks to control state aggregation, only lightweight processing is necessary at core routers. Examples of this approach include [58], [59], [60], [61], [62], [55]. In [58], [59], [60], QoS state aggregation is proposed to address the scalability of RSVP. By aggregating a large number of individual RSVP requests, e.g., on backbone links, it significantly reduces the number of request messages backbone core routers need to process as well as the granularity of control states they need to manage. In [61], a lightweight signaling protocol, YESSIR, is proposed to address several scalability issues associated with RSVP. It uses a sender-based model and piggybacks QoS reservation messages on top of RTCP [63] to reduce the processing overhead of RSVP. It also extends the all-or-nothing reservation model of RSVP to support partial reservations that improve over the duration of the session.

Scalable Resource Reservation (SRP) is another lightweight signaling protocol developed by Almesberger et al [62]. SRP uses in-band messages ("flags") carried in data packets to signal resource reservation intention from sources. It has the flavor of endpoint admission control approach we will discuss below, but requires routers' active participation in admission control process. It operates as follows: A source wishing to make a reservation starts by sending data packets marked with a *request* flag to destination. These packets are forwarded normally by routers, which also make flow admission decisions on per-packet basis. Based on feedback from destination, the source estimates how much of its reservation has been accepted in the network, and may then send data packets marked with a *reserved* flag at the accepted rate. The accepted rate is computed independently by sources and routers, using a "learn by example" procedure. Using the concept of deterministic effective bandwidth from network calculus, an adaptive estimation algorithm is developed for routers.

Under the core stateless framework proposed in [55], the scalability issue of QoS control plane is addressed by maintaining only *aggregate reservation state* at each router. A novel bandwidth estimation algorithm, which relies on the dynamic reservation information periodically carried in packets, is designed for estimating the amount of bandwidth requested by individual user flows. This estimate provides an upper bound on the aggregate bandwidth that is reserved, and is used to make admission control decisions at core routers.

Endpoint/Edge Admission Control. The end-point/edge admission control approach eliminates the signaling protocols and thus QoS reservation messages. Instead end hosts or edge routers perform admission control based on (noisy) measured resource availability information via *probe* packets. Hence core routers do not need to perform

any QoS control operations, besides simple queueing operations. Examples of this approach are [64], [65], [66], [67], [68], [69], [70]. Most of schemes are designed for DiffServ, and aim to provide some reasonable QoS *assurances* (*not* deterministic guarantees we discussed so far) for *adaptive* applications such as Internet audio and video streaming.

In the endpoint admission control scheme designed by Elek et al [64], an end host sends probe packets at the rate it wishes to reserve. The probe packets are queued at a separate lower priority queue at routers. Based on the drop rate of probe packets, the end host estimates whether sufficient resource is available in the network to accommodate its reservation and makes admission decision accordingly. The scheme proposed in [65] is very similar. It also uses packet drops and probe packet in a lower priority queue to make admission control decisions at end hosts. In [66] the admission control decision is made at *egress* edge routers instead of end hosts. The advantage of such a scheme is that edge routers can passively monitor the network load at an aggregate level, and thus it may provide more accurate load estimates. A more systematic study of endpoint admission control is carried out via simulation in [67]. In this simulation study, architectural issues such as deployability are major motivations in the choice of design options. Hence the authors consider endpoint admission control algorithms that can be implemented in the DiffServ architecture, and study several design issues such as thrashing and robustness in endpoint admission control algorithms. The general conclusion is that, when compared to traditional router-based admission control, endpoint admission control algorithms suffer only modest performance degradation. Hence the endpoint admission control approach may be a viable option in support of "soft" QoS guarantees for real-time adaptive applications.

The distributed admission control framework proposed in [68], [69], [70] is developed for the *best-effort* Internet using "pricing" mechanisms. In this framework, all packets (data or probe, elastic or real-time) are indistinguishable and thus treated equally. Upon congestion, packets are marked (e.g., using the ECN bit). Users must "pay" for marked packets. Based on the willingness of users to pay a certain price, admission control decisions can be made accordingly, either by end hosts or edge routers. In [69] network models are developed for studying the performance of the proposed distributed admission control framework. Fixed point approximations are applied to these models to derive acceptance marking probabilities at routers. A virtual queue mechanism is designed for detecting approaching traffic overload: a router marks packets or not depending on the state of a fictitious queue, of lower capacity than the real queue. Using many source asymptotics the authors show that the critical time scale of the virtual queue is same as the real queue, hence the proposed packet marking scheme is robust.

"Centralized" Bandwidth Broker. The notion of *bandwidth broker* (BB) is first proposed in [71] in the context of the DiffServ architecture for the support of *Premium Service*. In this approach, admission control, resource provisioning and other policy decisions are performed by a centralized bandwidth broker in each network domain. In [72] a two-tier bandwidth broker system is designed and implemented to support coarse-grain QoS provisioning for the DiffServ architecture.

In the context of the dynamic packet state architecture, Zhang et al [73] developed a (*conceptually*) centralized bandwidth broker architecture for scalable support of guaranteed services. This bandwidth broker architecture is built upon the virtual time reference system [53] we introduced in Section III-A. Taking advantage of the QoS abstraction of the data plane enabled by the virtual time reference system, the proposed bandwidth broker architecture *decouples the QoS control plane from the data plane*. More specifically, under this BB architecture, core routers do not maintain any QoS reservation states, whether per-flow or aggregate. Instead, the QoS reservation states are stored at and managed solely by the bandwidth broker(s) in a network domain. Despite this fact, the proposed bandwidth broker architecture is still *capable of providing end-to-end guaranteed services, whether fine-grain per-flow delay guarantees or coarse-grain class-based delay guarantees*.

Because of this decoupling of data plane and QoS control plane, the bandwidth broker architecture in [73] is appealing in several aspects. First of all, by maintaining QoS reservation states only in a bandwidth broker (or bandwidth brokers), core routers are relieved of QoS control functions such as admission control, making them potentially more efficient. Second, and perhaps more importantly, a QoS control plane that is decoupled from the data plane allows a network service provider to introduce new (guaranteed) services without necessarily requiring software/hardware upgrades at core routers. Third, with QoS reservation states maintained by a bandwidth broker, it can perform sophisticated QoS provisioning and admission control algorithms to optimize network utilization in a *network-wide* fashion. For example, in [73] the authors demonstrate how admission control can be performed at an entire path level, instead of on a "hop-by-hop" basis. Such an approach can significantly reduce the complexity of the admission control algorithms. Such network-wide optimization is difficult, if not impossible, under the

router-based hop-by-hop signaling approach, nor is it possible under the endpoint/edge admission control approach. Furthermore, under the proposed bandwidth broker approach, the reliability, robustness and scalability issues of QoS control plane (i.e., the bandwidth broker architecture) can be addressed *separately from, and without incurring additional complexity to, the data plane*. In other words, the bandwidth broker architecture is only centralized *conceptually*, with respect to the data plane. Distributed or hierarchical bandwidth broker systems can be designed under the framework proposed in [73] (see, for example, the work in [74]).

To conclude this section, we remark that there are a multidimensional spectrum of numerous possible approaches toward providing QoS guarantees; IntServ and DiffServ are but two point solutions in this spectrum. These approaches vary according to the time scale and granularity (e.g., per-packet, per-flow, or per flow-aggregate) of the control adopted, and the amount of state/complexity required in the end systems and edge and core routers – considerations that all impact the *scalability* of these approaches. Time scale and granularity of QoS controls determine at what levels user traffic can be differentiated and how frequently the network can apply control operations on user traffic. Therefore, they directly affect the fundamental trade-offs in QoS provisioning: the trade-offs among levels of services and performance that can be offered by a QoS solution, the network resource usage it can achieve, and its associated implementation complexity and operational/management costs. Much research is still needed to analyze and quantify these fundamental trade-offs in QoS provisioning.

IV. Statistical Guarantees

Quality of service guarantees may be given with some probability, rather than on a deterministic basis as in Section II-C. Doing so relies on the possibility to (1) model user traffic and (2) estimate probabilities of satisfying some quality of service.

A. Model Based Approaches

A large body of work exists on computing loss and delay probabilities, assuming that user flows satisfy some a priori traffic model, for example: Markov modulated fluid [75], [76], fractional brownian traffic [77]; see also the collective book edited by J. Roberts, U. Mocci and J. Virtamo [78].

Better than Poisson/MTU and Negligible Jitter. A difficulty with the approach mentioned above is to give a convincing model of traffic *inside* the network. A radical solution is proposed by Bonald et al in [79]; it applies to constant rate sources, shaped at network access, that are assumed to be independent, in a stochastic sense. The independence assumption is at network access, not inside the network. Inside the network, all such traffic is served in non-preemptive priority schedulers. This represents a simplified model of EF (see Section II-C). The starting point for the analysis is that that periodic sources are "better than Poisson/MTU", which means that the queue length at a scheduler is majorized, in some sense, by that of the same scheduler fed with a Poisson flow of packets of constant size equal to the maximum transfer unit (MTU). The authors propose four different possible approaches to the majorization, each of them having slightly different mathematical implications.

The better than Poisson/MTU assumption is proved to be formally true for *fresh* traffic. The key observation is then that sources continue to be better than Poisson/MTU *inside* the network; this is also called the "negligible jitter" property. This property is posed as a conjecture; while it is supported by simulations and analysis of special cases, an exact analysis seems to pose a formidable challenge. Accepting the conjecture, every node inside the network can be analyzed as a simple $M/D/1$ queue, the only input parameter being the traffic intensity at this node. Bonald et al further assume that, for the distribution of end-to-end delay, independence at every hop is a worst case assumption; this allows them to compute the distribution of the end-to-end delay as the convolution of the delays at every hop.

B. Approaches Based Only on Independence at Network Access

An alternative approach makes no assumption about the distribution of sources, other than independence of different sources at network access, stationarity, and the fact that fresh sources are shaped at network access. With these weak assumptions, it is possible to find good probabilistic bounds. A first family of results is based on a heuristic which assumes that the worst case traffic is made of on-off sources [80], [81], [82], [83]. In contrast, in the rest of this section, we describe results that are exact bounds.

Hoeffding Bounds. A formally proved bound for a node modeled as a constant rate server is found by Kesidis et al in [84]. Another bound for the same model is found later by C.S. Chang et al [85] who show that their bound is better than the former, and asymptotically tight. These results are extended by Vojnović et al [86] to the case where the node can be modeled with a service curve, instead of being a constant rate server, which better reflects the EF assumptions. More interestingly, Vojnović et al show that all these bounds are application of more generic bounds

found by Hoeffding in 1963 [87], which apply to the sum of a collection of independent (not necessarily identically distributed), bounded random variables, assuming that the expectation of the sum is known.

The generic method for all of the above bounds is (1) to majorize the queue length by a sum of independent processes (2) use network calculus to give deterministic bounds on these independent processes and (3) apply Hoefding bounds. We now re-write the two bounds mentioned above bound using this generic method.

The Bound by Kesidis et al. Consider a node offering a service curve guarantee $\beta(t)$ to an EF aggregate, made of flows $A_i(t)$ that are independent, stationary and individually shaped. Call $\alpha_i(t)$ an arrival curve for flow i (thus $A_i(t) - A_i(s) \leq \alpha_i(t-s)$. For simplicity, we explain the method on the homogeneous case $\alpha_i = \frac{1}{I}\alpha$. By definition of a service curve, the queue length at time t satisfies

$$Q(t) \leq \sup_{s \geq 0}\{\sum_i [A_i(t) - A_i(t-s)] - \beta(s)\} \qquad (22)$$

Here, step (1) is based on $Q(t) \leq \sum_i Q_i(t)$, with

$$Q_i(t) = \sup_{s \geq 0}\{A_i(t) - A_i(t-s) - \frac{1}{I}\beta(s)\}$$

Step (2) is the deterministic backlog bound (6) $Q_i(t) \leq \frac{b_{\text{req}}}{I}$, where b_{req} is the buffer size required for loss free operation, given by Equation (6).

Step (3) consists in applying a Hoeffding bound, here Theorem 1 in [87], formula (2.1). It is valid for a sequence of independent, bounded random variables, here $0 \leq Q_i \leq \frac{b_{\text{req}}}{I}$, assuming that we know $\mu = \frac{1}{I}\mathbb{E}(\sum_{i=1}^{I} Q_i(t))$. We do not know μ, but the Hoeffding bound is increasing with μ, thus it is sufficient to have an upper bound on μ. We obtain one such bound by Little's formula

$$\mathbb{E}(Q_i(t)) \leq \rho_i D_i$$

where ρ_i is a bound on the intensity of arrivals of flow i and D_i is a bound on the delay that would be experienced by any bit if the system would be FIFO. Call $D_{\max}$ the worst case delay in the loss-free system. The deterministic delay bound (6) gives $D_i \leq h(\frac{1}{I}\alpha, \frac{1}{I}\beta) = h(\alpha, \beta) := D_{\max}$. Finally $\mu \leq \frac{\rho}{I} D_{\max}$, where ρ is the total traffic intensity (by stationarity $\rho \leq \inf_{t>0} \frac{\alpha(t)}{t}$ [5]). This gives the bound

$$\mathbb{P}(Q(t) > b) \leq$$
$$\exp\left(-I\frac{b}{b_{\text{req}}}\ln\frac{b}{\rho D_{\max}} + I\left(1 - \frac{b}{b_{\text{req}}}\right)\ln\frac{b_{\text{req}} - \rho D_{\max}}{b_{\text{req}} - b}\right)$$

which is valid for $\rho D_{\max} \leq b \leq b_{\text{req}}$. For $\beta(t) = ct$, this is the bound in [84].

The Bound by Chang et al. Here we assume in addition that β is *super-additive*, which means that $\beta(t+s) \geq \beta(t) + \beta(s)$ [47]. This is not restrictive, as it is true for any convex β, in particular for rate-latency functions assumed for DiffServ. Now assume that τ satisfies $\alpha(\tau) \leq \beta(\tau)$. If we interpret β as a strict service curve (Section II-B), then τ is an upper bound on the duration of any busy period. However, this interpretation is too restrictive, as explained in Section II-B; the general statement that can be made is that (22) can be specialized to

$$Q(t) \leq \sup_{s \leq \tau}\{\sum_i [A_i(t) - A_i(t-s)] - \beta(s)\}$$

which is the starting point for step (1). Now let $\underline{s} = (0 = a_0, a_1, ..., a_K = \tau)$ be a partition of the interval $[0, \tau]$. It follows from the previous equation and the monotonicity of β and A that

$$Q(t) \leq \max_{k=0}^{K-1}\{A(t) - A(t - s_{k+1}) - \beta(s_k)\}$$

thus

$$\mathbb{P}(Q(t) > b) \leq \sum_{k=0}^{K-1} \mathbb{P}(A(t) - A(t - s_{k+1}) > \beta(s_k) + b\} \qquad (23)$$

Fix k and define $Z(t) = A(t) - A(t - s_{k+1})$. Step (1) is concluded by observing that $Z(t) = \sum_{i=1}^{I}\{A_i(t) - A_i(t - s_{k+1}\}$ which is a sum of independent processes.

Step (2) simply consists in the arrival curve bound $0 \leq Z(t) \leq \alpha_i(s_{k+1})$. For step (3), we apply the same Hoeffding bound as in the previous case, which follows from the bound $\mathbb{E}(\sum_{i=1}^{I}(A_i(t) - A_i(t - s_{k+1})) \leq \rho s_{k+1}$. Combining with (23) gives

$$\mathbb{P}(Q(t) > b) \leq \sum_{k=0}^{K-1} \exp(-I g_k)$$

with

$$g_k = \begin{cases} +\infty,\ b > \alpha(s_{k+1}) - \beta(s_k) \\ 0,\ b < \rho s_{k+1} - \beta(s_k) \\ \frac{\beta(s_k)+b}{\alpha(s_{k+1})}\ln\frac{\beta(s_k)+b}{\rho s_{k+1}} + \\ +\left(1 - \frac{\beta(s_k)+b}{\alpha(s_{k+1})}\right)\ln\frac{\alpha(s_{k+1})-\beta(s_k)-b}{\alpha(s_{k+1})-\rho s_{k+1}},\ \text{otherwise} \end{cases}$$

which, for $\beta(t) = ct$ and $s_k = k$ is the bound in [85]. Taking the minimum over a set of partitions $\underline{s}$ gives a better bound [86]. See [88] for the general case where α_i's are not identical.

Application to DiffServ(EF). The bounds can be used for statistical guarantees. First, an EF node can be modeled as a rate-latency service curve. Second, it is necessary to account for traffic *inside* the network. Chang et

al [85] propose the following method, which again uses a deterministic bounds to obtain a stochastic result. The increments $A^n(t) - A^n(s)$ of the EF aggregate that feeds node n are majorized by

$$A^n(t) - A^n(s) \leq A^{0,n}(t) - A^{0,n}(s - \Delta)$$

where $A^{0,n}(t)$ is the aggregate of all *fresh* EF traffic whose path uses node n and has arrived at the network boundary at or before time t, and Δ is a deterministic bound on the delay jitter across $H - 1$ hops (H is the maximum hop count for any flow). Δ can be obtained by formula (14) if buffers are large. If EF buffers are small – a more likely assumption – then the delay-from-backlog bound of PSRG provides the better bound $\Delta = (H-1)\max_m\{B_m/c_m + e_m\}$, where B_m is the buffer size, c_m the service rate, and e_m the latency of the node m. The microflows that constitute the aggregate processes $A^n(t)$ are not independent at node n, but $A^{0,n}(t) - A^{0,n}(s - \Delta)$ may be decomposed from its constituent microflows at network access, which are assumed to be independent. This majorization also accounts for possible packet losses between network access and node n. Then the bounds seen above can be applied. Vojnović et al ([88], Theorem 2) show how this can be used to compute congestion probabilities, given that only aggregate information is available, as is usual with DiffServ.

Loss ratios may differ from congestion probabilities, because packet losses do not necessarily affect all microflows in the same way. In [88], the loss ratio is estimated from the congestion probabilities; this is based on a deterministic bound on loss found by Chuang et al [89], combined with a stochastic analysis by Likhanov et al [90]. In practice though, dimensioning a network on a small congestion probability is normally sufficient [91].

Delay distributions can be obtained from the delay-from-backlog property of PSRG (Section II-C), if buffers are small. Indeed, in that case, the delay at every hop is bounded deterministically, as mentioned above. Else a well known method consists in first computing $\mathbb{P}^0(Q(t) > b)$, the distribution of backlog seen by an arriving packet. This differs from the bounds given above in that the probability is conditional on arrival (this is called a Palm probability). Call c_n the rate of the EF node number n (modeled as a PSRG node); it is shown in [88] that

$$\mathbb{P}^0(Q(t) > b) \leq \frac{c_n}{\rho_n}\mathbb{P}(Q(t) > b)$$

where ρ_n is the total EF traffic intensity at this node. This generalizes a result by Konstantopoulos et al [92] who showed that equality holds in the previous equation if the node is a constant rate server with rate c_n. It follows from the delay-from-backlog property that

$$\mathbb{P}(D_n > u) \leq \frac{c_n}{\rho_n}\mathbb{P}\{Q(t) > c_n(u - e_n)\} \qquad (24)$$

for $u \geq e_n$, where D_n is the delay of an arbitrary packet at node n. The end-to-end delay distribution can then be bounded as follows

$$\mathbb{P}(D > u) \leq \sum_{n=1}^{H} \mathbb{P}(D_n > \frac{u}{H})$$

This last bound holds without any assumption on the dependence of delays at consecutive nodes. It is argued in [79] that assuming these delays are independent gives a conservative approximation; thus an approximation to the distribution of D would be the convolution of the distributions of D_n, as estimated by (24).

In [88] this method is compared to the approximation based on the method "better than Poisson/MTU". For large number of sources, the bounds converge to the approximation; for small number of sources with large burstiness, the results suggest that the approximation may be too optimistic.

Related Approaches. Boorstyn et al [93] define a concept of effective envelope, which captures statistical multiplexing between independent flows. It is based on Chernov bounds and the central limit theorem; the approach can be re-written using Hoeffding bounds, as above. The effective envelope is then is used to evaluate the amount of multiplexing that can be achieved in constant rate servers. The concept is further developed in [94], which introduces the idea of effective service curve; this allows application to network scenarios such as EF. The end result is similar to the previous method; however, the method of effective envelope and effective service curve does not give closed form expression, unlike the method based on the bounds presented above. A related method is exponentially bounded burstiness [95], [96] and the generalization in [97], which considers some restricted forms of effective envelopes.

V. QoS Guarantees for TCP-dominated Traffic

In Section II, we presented a deterministic network calculus whereby deterministic guarantees (bounds) on service characteristics such as delay and throughput were derived based on traffic and service bounds. In Section IV, we presented some forms of "stochastic network calculus" that derive stochastic service guarantees from stochastic bounds on traffic combined with deterministic or stochastic bounds on service. In this section, we survey recent

advances related to "elastic network calculus", where traffic is transported by the Transmission Control Protocol (TCP) [98]) which is subject to a closed-loop congestion control algorithm (see also [99], [100]). The term "elastic" refers to the ability of TCP to adjust its sending rate as a function of network conditions.

The primary motivation for modeling elastic traffic and using the models to provide QoS guarantees stems from the fact that the vast majority of Internet traffic is transported over the TCP protocol. Early measurements on the MCI-operated vBNS network were reported in Thompson et al [101] showing TCP at 95% of total byte traffic, 85%-95% of packet traffic and 70%-85% of flows, of which 70%-75% was Web traffic. More recently, McCreary et al [102] reported measurements at NASA Ames Internet Exchange showing TCP at 80%-85% of total packet traffic.

TCP congestion control has evolved through several variants in the last decade including Tahoe [103], Reno [99], SACK [104], FACK [105]. Currently, the majority of implementations are based on Reno or SACK. Briefly, the TCP sender maintains a window of packets "in flight" (i.e., sent and not yet acknowledged (ACKed)). TCP congestion control follows an "Additive-Increase-Multiplicative-Decrease" (AIMD) algorithm, where the window is increased linearly in time as transmission progresses without errors, and the window is halved (congestion avoidance algorithm) when a missing ACK condition is detected (a.k.a. "Triple-Duplicate ACK" or TD for short). The additive increase is resumed after the error condition is removed. The cause of the error is usually a packet drop by a network element (router, switch) due to congestion. Frequent packet drops can cause a TCP sender to stop sending for a while (time-out or TO). Transmission eventually resumes after a timeout with a window of one packet, and if this is successful, AIMD is resumed, otherwise another time-out occurs with double duration.

The TCP congestion control algorithm provides a certain sending rate T depending on the network conditions such as the packet drop probability p and round trip time R (the time between a packet is sent and its corresponding ACK is received at the TCP sender). Many recent works have proposed models for the stationary "long-term" TCP sending rate as well as for the short term and transient behavior of TCP flows. In the following we review several representative models and their application to predicting performance and providing QoS guarantees to "elastic" traffic.

A. Models for Expected Rate, Delay and Loss of TCP Traffic

Early models for the average TCP sending rate[5] assumed long-lived TCP flows with stationary network conditions, and ignored timeouts. They yielded the following expression for sending rate,

$$T = \frac{aM}{R\sqrt{p}} \tag{25}$$

where p is the average packet drop probability, R is the average round trip time, M is the average size of TCP packets and a is a small constant. Ott et al [106] adopt a continuous time fluid flow model of the TCP window size and describe its behavior by a stochastic differential equation. Their model yields $a = 1.3/\sqrt{b}$ if packet losses are assumed independent, where b is the number of packets confirmed by one ACK (usually $b = 2$). A model with periodic losses is proposed by Mathis et al [107] where they derive the above expression for sending rate T with $a = \sqrt{3/2b}$. It is shown to be close to simulation experiments and Internet measurements provided time-outs are rare.

A major shortcoming of these models is that they diverge significantly from measurements when packet loss probability is above 0.02 due to a significant occurrence of timeout events. Through extensive Internet measurements, Padhye, Firoiu, Towsley and Kurose [108] show that the majority of traffic is subject to loss probabilities above that threshold and that TCP time-outs have a significant impact on TCP sending rate. They propose a model (sometimes known as "PFTK") for the stationary TCP sending rate that includes the effect of time-outs and TCP receiver's advertised maximum window size W_m. The result of analyzing this model is an expression that accounts for timeouts and maximum window size. The fact that this model is comprehensive (includes most aspects of TCP congestion control) and is proved to be fairly accurate for the full range of drop probability and practical round trip times, has made it a model of choice for applications such as modeling network performance (presented next), multicast congestion control (see [109]) and TCP-friendly congestion control (presented later in this section).

So far, all models assumed long-lived TCP flows, i.e., flows that transfer a large amount of data such that the transient, first part can be ignored. For short TCP flows, the transient known as "slow start" represents a large part of the session. In slow start, the TCP window approximately doubles at each consecutive round trip time until

[5]In the following we consider models for TCP sending rate, reserving the term "throughput" for the data rate seen at the TCP receiver. Throughput is smaller than sending rate due to packet losses.

a first loss event is detected. This event ends slow start and begins the congestion avoidance algorithm described earlier. Modeling short TCP flows is important since most Web traffic falls in this category. (Internet measurements [110], [101], [102] reported Web transactions with an average of 8–16KB). Cardwell et all [111] extend the PFTK model [108] by adding a model for TCP slow start. They derive the expected duration for transferring d bytes

$$E[t] = E[t_{ss}] + E[t_{loss}] + E[t_{ca}] + E[t_{delack}]$$

where $E[t_{ss}], E[t_{loss}], E[t_{ca}], E[t_{delack}]$ are the expected time intervals spent in slow start, first loss, congestion avoidance and delayed ACK respectively, where $E[t_{ca}]$ follows from the PFTK model. The average TCP sending rate is approximated by $T = d/E[t]$. It is important to note that in newer HTTP implementations (such as HTTPv1.1) consecutive small transactions are carried over persistent TCP connections (if they are sufficiently close in time) and thus slow start has less impact on TCP sending rate than in the Cardwell model.

The models for expected TCP sending rate can be used for modeling network performance. Firoiu et al [112] (and independently Misra et al [113]) presents a model for a set of N TCP flows sharing a single congested link. Assuming an average queue size q and drop probability p at that link and using one of the TCP models above for the N flows, $T_i(p, R_i)$, we can state that the link's capacity c is fully utilized by the N flows:

$$\sum_{i=1}^{N} T_i(p, R_i^0 + q/c) = c,$$

where R_i^0 is the round trip propagation delay for flow i and q/c is the average queue wait time. Given that all expressions derived for T are strictly decreasing in R and P, the above equation implies that q is a decreasing function of p:

$$q = G(p). \tag{26}$$

This "queue law" imposes a direct relationship between the average queue size and drop probability at a link congested by a given set of TCP flows, independent of how the packets are selected to be dropped. [112] shows a good match between the model and simulation experiments, despite the fact that the PFTK model assumes correlated losses whereas the experiments implemented independent losses.

An active queue management policy determines what packet and when it is dropped. It is described by a "queue control function" H. The steady state values of q and p are determined by the queue law, (26) and the following equation

$$p = H(q).$$

For example, the Random Early Detection (RED) algorithm [114] implements a simple, two segment, increasing control function H. The characteristics of the queue law and the control function are such that there is a unique solution (q_s, p_s) to satisfy both equations. This corresponds to the steady-state operating point of the system of N TCP flows coupled with the active queue management and has been shown to match well with experiments. Once q and p are determined, the average TCP sending rate of flow i is $T_i(p_s, R_i + q_s/c)$. The average rate guarantee of useful data at the TCP receiver (goodput) g_i can further be computed from T_i by accounting for packet drops and retransmissions. Thus, the QoS guarantee provided by a TCP flow is characterized by an average goodput, loss and delay $(g_i, p_s, R_i^f + q_s/c)$, where R_i^f is the forward propagation delay.

The model for N TCP flows and one congested link can be extended to networks with arbitrary topology. Firoiu, Yang and Zhang [115] and Bu and Towsley [116] have independently proposed models for arbitrary networks with TCP and non-TCP flows and RED queue management. They model a network as a set of links K, a set of TCP flows F and a set of UDP flows U where a UDP flow $f \in U$ has average rate u_f. Each flow $f \in F \cup U$ traverses a path P_f (ordered set of links in K) within the network. The first part of P_f from source up to and including link k is denoted by $P_{f,k}$. For each link $k \in K$, its bandwidth c_k is given along with its propagation delay d_k and queue control function $p_k = H_k(q_k)$. The model's unknowns are the average queue sizes q_k and link drop probabilities p_k at each link $k \in K$ and the average sending rate T_f for each TCP flow $f \in F$. The network model is given by the following set of equations

$$\begin{aligned}
p_k &= H_k(q_k), k \in K && (27)\\
T_f &= T_f(p_f, R_f), f \in F && (28)\\
p_f &= 1 - \prod_{k \in P_f} (1 - p_k) && (29)\\
R_f &= \sum_{k \in P_f} (d_k + q_k/c_k) && (30)\\
T_f &= u_f, f \in U && (31)\\
T_{f,k} &= T_f(1 - \prod_{k \in P_{f,k}} (1 - p_k)) && (32)\\
\sum_{f \in F_k} T_{f,k} &\leq c_k. && (33)
\end{aligned}$$

The last equation constrains the sum of expected flow throughputs at each link by the capacity of that link. Flow

throughputs at link k are "thinned" versions (32) of the average sending rates of UDP flows (31) or TCP flows (28), where the end-end drop probability and round trip delay are compositions of per-link values, (29) and (30) respectively.

The system can be solved numerically using various fixed point methods. [115] proposed an algorithm that exhibits quick convergence properties for $(q_k)_{k\in K}$ while [116] used a standard algorithm from MATLAB. Also, [115] adapts results from [117] for the TCP sending rate in the context of Differentiated Services, and [116] includes a more elaborate model for UDP flows. The models were confirmed by simulation experiments with various network topologies. As an open problem, neither existence nor uniqueness of solutions have been established for the system of equations (27)-(33); however a unique solution was quickly reached in all examples tried, including degenerate topologies. Gibbens et al [118] proposed a similar model for a random number of TCP flows in an arbitrary network with two drop priority classes, but with $M/M/1/K$ queueing models and without experimental verification.

B. Models for the Dynamic Behavior of TCP Traffic

The models presented so far provide steady-state averages of sending rate, queue size and drop probability. They do not provide any indication of their variability in time or conditions under which convergence to steady state occurs. For example, [112] showed through simulation that the queue size oscillates when a RED control function is discontinuous or the linear segments having large slopes.

Recently, fluid models have been proposed for studying the dynamic behavior of TCP sending rate, queue size, and loss, their stability and their higher moments. An early model for the dynamics of TCP sending rate in a network with constant drop probability and RTT was proposed by Ott et all [106]. They model the evolution of the TCP window size $W(t)$ at time t through a stochastic differential equation (SDE) where loss indications are described by a time varying Poisson process $\{N(t)\}$ with intensity $\lambda(t) = W(t)p$. The SDE is

$$dW(t) = R^{-1}dt - W(t)/2dN(t) \tag{34}$$

Here R is the round trip time. Note that this equation captures the AIMD behavior of TCP but not the timeout behavior. By rescaling the process in time, the authors transform it into one where losses are governed by a time invariant Poisson process. They then analyze this process to obtain the stationary distribution and higher moments of $W(t)$.

Based on a statistical analysis of network traces, Misra et al [119] conclude that the loss process is well modeled by a time invariant Poisson process with intensity $\lambda(t) = \lambda$. They then derive the following differential equation describing the behavior of $\overline{W}(t)$

$$\frac{d\overline{W}}{dt} = \frac{1}{R} - \frac{\lambda\overline{W}}{2} \tag{35}$$

By setting $d\overline{W}/dt = 0$, it is possible to retrieve the square root p formula for the stationary sending rate. They also extend the analysis to account for a single timeout. This model shows a good match with Internet measurements reported in [108]. As a tradeoff, it does not model the intensity of loss events λ_{TD} and λ_{TO} as a function of network drop probability p (they are taken directly from experimental traces) and does not model multiple time-outs. This work has been extended to account for any losses described by any stationary ergodic process, [120].

The model is extended in Misra et al [121] to include active queue management such as RED. The model considers a set of N TCP flows sharing a congested link of capacity c and queue size $q(t)$ that varies over time. The underlying behavior is described by a set of SDEs analogous to (34) that can be used to obtain a set of differential equations describing the behavior of the average window sizes. Let $\overline{W}_i(t)$ for flow $i = 1, \ldots, N$ similar to (35)

$$\frac{d\overline{W}_i}{dt} = \frac{1}{R_i(t)} - \frac{\overline{W}_i(t)}{2}\lambda_i(t) \tag{36}$$

Observe that here the round trip time R_i and packet loss intensities λ_{TD} can vary over time. The round trip time is a combination of round trip propagation delay R_i^0 and queueing delay

$$R_i(t) = R_i^0 + q(t)/c \tag{37}$$

The packet loss intensities are proportional to the flow's sending rate T_i and drop probability

$$\lambda_i(t) = \overline{W}_i(t)p(q(t)) \tag{38}$$

The expected TCP sending rate is

$$T_i(t) = \frac{\overline{W}_i(t)}{R_i(t)} \tag{39}$$

The RED control function determines the drop probability at the link

$$p(t) = H(\hat{q}(t)) \tag{40}$$

In RED, $\hat{q}$ is an estimate of the queue size q computed from samples taken every δ seconds and combined in an exponential moving average with parameter α

$$\hat{q}((k+1)\delta) = (1-\alpha)\hat{q}(k\delta) + \alpha\hat{q}(k\delta), k = 1, 2, ...$$

Thus, the evolution of $\hat{q}$ can be approximately described by

$$\frac{d\hat{q}}{dt} = \frac{\ln(1-\alpha)}{\delta}(\hat{q}(t) - q(t)) \quad (41)$$

Finally, the balance of flow rates in and out of the queue (Lindely's equation) states

$$\frac{dq}{dt} = -c + \sum_{i=1}^{N} T_i(t) \quad (42)$$

The system of equations (36)-(42) is determined and can be solved numerically. The result is an estimate of the TCP sending rate evolution in time for all N flows. The model was verified with simulations and was also extended to networks with arbitrary topologies.

This model of transient TCP behavior also reveals instability and oscillations under certain conditions. The main cause of instability is identified to be RED's control mechanism (the combination of control function and queue estimation). Based on (36)-(42) this problem is further analyzed by Hollot et al [122] using arguments of feedback control theory. They show that RED becomes less stable as the number of sessions decreases and the average session round trip time increases. They then provide conditions on the RED control function H and queue estimation parameters α, δ sufficient for the stability of a system with $N > N^-$ TCP flows and average round trip time $R < R^+$. In order to stabilize the queue size at a certain value, the RED control function needs to have a high slope. In this case, the TCP+RED system is stable only if the RED queue estimator has a long memory. This in turn entails a slow closed-loop response of the RED feedback control, which is unable to adjust to normal traffic changes.

The sluggishness of the RED control system is due to the queue estimator using the exponential moving average that acts as a low-pass filter. A faster response can be achieved by a different queue controller using both proportional and integral feedback without compromising stability. The proportional-integral controller (PI) is a classic solution in Feedback Control Theory and a variant is proposed in Hollot et al [123]. The PI controller is designed with the objective of stabilizing the queue size $q(t)$ at or near a reference value q_{ref}. The PI controller generates a loss probability p proportional to the "error" $q_e(t) = q(t) - q_{ref}$ and to the error's integral. In a discrete time system where the queue is sampled at intervals of δ seconds, an implementation of PI controller can be

$$p(k\delta) = aq_e(k\delta) - q_e((k-1)\delta) + p(k-1)\delta).$$

Besides responding more quickly to perturbations while being stable, PI also has the ability to set a reference objective queue size independent of the steady-state drop probability p. This improves the ability to provide low queueing delays for a wider range of traffic load (number of TCP flows and round trip times). This comes with a cost, namely that for a given traffic load N, a smaller queue size implies a larger drop probability p, as stated by the queue law (26) that is always a decreasing function. A higher drop probability is detrimental to the efficiency of TCP transfers, decreasing their goodput and predictability (see [124] Section III for more details). Therefore, most benefits of PI are reaped only if the loss events are signaled to TCP senders through Explicit Congestion Notification (ECN [125]) and not through actual packet drops.

Last, recent AQM algorithms associated with random early marking (REM) [126] and adaptive virtual queue control [127] have adopted PI controllers for similar reasons. The latter mechanism is interesting because its goal is to reduce delays by maintaining the link utilization at a reference utilization below 100%.

C. Providing Service Guarantees with TCP Flows

In general, providing service guarantees corresponds to a network service provider offering a data transport service between two or more end-points with a certain set of QoS levels (lower bound on rate, higher bound on end-end delay and loss) and a certain degree of assurance (probability or proportion of violation of the above bounds lying below some threshold). In order to ensure such QoS guarantees, the providing network is managed through admission control of service requests and path selection of admitted flows.

The models described earlier in this section have led to an "elastic network calculus" whereby QoS levels of all flows in a network can be predicted given the traffic load and network characteristics. For example, both the average model in (27)-(33) and the dynamic model in (36)-(42) (extended to a network) result in predictions for QoS levels (average rate, end-end delay and drop probability) given a network topology, routing of flows, capacities and queue management for all links. This can be used by network management and traffic engineering to design a network (topology, capacities, queue management, routing) given a set of load matrix (source, destination, number and QoS levels of flows). It can also be used on-line to assist the admission control of new flows in the network by checking if the addition of a new flow would provide it with the requested QoS level while not compromising the QoS levels of all other flows.

The TCP congestion control has been designed to reduce congestion in a network while giving all flows the opportunity to make use of all available capacity in a "fair"

way. While the concept of fairness has had many definitions and research, there is no commonly agreed definition that can be applied to data networks. TCP congestion control provides equal average sending rate for each of a set of flows that have the same round trip time, drop probability, packet size, maximum window size, as can be seen from (25) and the more accurate PFTK model [108].

While the rationale behind this "TCP-fairness" is outside the scope of this article, we note that a fair amount of work has concentrated on it. Floyd et al [128] argues that, in a network with non-differentiated service (also known as Best Effort service) such as the current Internet, all traffic has to be "TCP-fair" irrespective to the transport protocol used (TCP or not) in order to avoid a congestion collapse. The "TCP-fairness" can be implemented in a rate control algorithm using TCP's AIMD algorithm or a TCP model such as PFTK as proposed in [129] and also studied in [130]. A variety of window-based mechanisms using increase rules other than additive increase and decrease rules other than multiplicative decrease have recently been developed and judged to be TCP fair [131]. All of these mechanisms tend to produce smoother flows than TCP does; a trait that is considered desirable in the transport of multimedia. A mechanism for assessing "TCP-fairness" in existing implementation is proposed in [132].

In general, there are many cases where QoS levels required by applications are significantly different from the "fair/equal" levels provided by undifferentiated networks. To provide such QoS levels, parts of the traffic needs to be treated with discrimination. This can be achieved by giving each flow a specific treatment through such mechanisms as differentiated bandwidth reservation, scheduling and queue management, as defined by IntServ [133], [20]. A simplification that reduces complexity and increase scalability is defined by DiffServ [2] whereby differentiated traffic treatment is applied to groups of flows with the tradeoff of decreased assurance of QoS levels for individual flows. The DiffServ Assured Forwarding Per-Hop Behavior (AF PHB) [134] guarantees the forwarding of a traffic sending below a committed rate and forwards without guarantees traffic above that rate. Many research works have modeled the behavior of TCP traffic under these two types of treatment giving predictions for QoS levels under various network settings.

Yeom and Reddy [117] consider a TCP flow using AF PHB with a committed rate g. Traffic rate is measured at the sender (using for example a sliding window mechanism) and packets within the committed rate are marked "in-profile" and the rest as "out-of-profile". If the committed rate g is reserved in the network then the service is said to be under-subscribed and it is assumed that only OUT packets are dropped with probability p_{out}. Otherwise the service is over-subscribed and it is assumed that all OUT packets are dropped and IN packets are dropped with probability p_{in}. They propose a model for average TCP sending rate as a function of subscription status, p_{in}, p_{out} using similar arguments as the PFTK model [108]. A simplified expression for the average excess sending rate (above the committed rate g) of a TCP flow experiencing drop probability p_{out} is

$$T_e = \begin{cases} \frac{3M}{4R}\sqrt{\frac{2}{p_{out}}} - \frac{g}{4} & \text{if } g \geq \frac{M}{R}\sqrt{\frac{2}{p_{out}}} \\ \sqrt{\frac{g^2}{4} + \frac{3M^2}{2Rp_{out}}} - \frac{g}{2} & \text{otherwise} \end{cases}$$

where M, R are packet size and round trip time. The expression shows that, for high committed rate or narrow under-subscription ($p_{out} \rightarrow 1$), the excess rate is small or negative (i.e., TCP flow cannot achieve its committed rate). Usage of excess bandwidth is biased toward flows with small committed rates.

The main difficulty in achieving a desired rate with TCP flows in the context of AF (marking and differentiated dropping) is that TCP congestion control is unaware of the cause (marking) of dropped packets. Sahu et al [135] determine the parameters of a leaky-bucket marking necessary for guaranteeing a given rate, if at all possible. They reach similar conclusions as above such as that in the under-subscribed case, when the committed rate is small, the marking has no influence on the achieved rate.

In a recent work, Chait et al [136] propose adding adaptive rate mechanisms (ARMs) to the leaky-bucket markers. An ARM monitors the sending rate attained by an aggregate and sets the token rate so that the aggregate achieves a minimum sending rate. Simulations demonstrate that ARMs coupled with a multilevel AQM policy provide these minimum sending rates, provided that they sum to less than the bandwidth available in the network. The latter inequality can be guaranteed through call admission.

We conclude here our survey of providing QoS guarantees to elastic traffic dominated by the TCP congestion control. We have seen that both steady-state and dynamic models can be formulated for arbitrary network and traffic conditions which result in fairly accurate predictions for QoS levels. The major obstacle identified is that guaranteeing different rates for different flows or groups of flows is difficult or sometimes impossible if the set of guarantees is far from the undifferentiated, best-effort, rate. The opportunity of using TCP congestion control for providing differentiated QoS levels is under question, and changing TCP congestion control or replacing it with other mechanism in the context of differentiated QoS is currently an

open research area.

VI. Service Differentiation within Best-Effort

With DiffServ and IntServ, quality of service is given to some data, as a form of better service. We have described in Section II the standard DiffServ and IntServ mechanisms. A common feature is that low delay is usually linked to a form of priority, thus to more throughput in case of congestion. Kilkki proposes a different approach (SIMA [137]) by which a priority level (0-7) is set for an entire flow as a function of how the flow deviates from its contractual rate; if a flow exceeds its rate, its priority level is low). As a result, it is optimal for a flow to be adaptive. Bandwidth is then shared on a best effort basis between low delay and other flows. Real time flows that conform to their rate are able to obtain a low delay, but do not get throughput priority. However, all these methods need some form of admission control.

In contrast, a number of authors have proposed service differentiation without admission control. The main motivation is to retain the best effort, flat rate type of commercial agreements that are believed to be the basis for the rapid deployment of Internet in the 90s. Dovrolis et al [138] propose a proportional differentiation model where the quality between classes of traffic is proportional and thus can be performed independently of the load within each class. Central to their work is the use of two packet schedulers BPR (Backlog Proportional Rate) and WTP (Waiting-Time Priority) to approximate the behavior of the proportional differentiation model. Moret and Fdida [139] also describe a two-class proportional differentiation model called Proportional Queue Control Mechanism (PQCM). Both studies propose controlling the relative queueing delays between classes.

A simpler service alternative is proposed by Hurley et al under the name Alternative Best Effort (ABE) [140] or by Guo et al under the name Best Effort Differentiated Service (BEDS) [124]. Both propose to associate a priority for delay with a negative priority for throughput (or loss). A packet that is marked as low delay (called "green" in ABE) has more risk of being dropped (or marked with explicit congestion indication). If the relative values of dropping probabilities and delays are well set by the router implementations, then it is advantageous for an application that uses TCP *not* to mark packets with the low delay bit; in contrast, it is advantageous for an Internet telephony application to mark its packets with the low delay bit, as long as the throughput it receives is not too low. The key feature of the service is that an application marking some of its packets with the low delay bit does not impact other applications that would not mark their packets. This would allow an incremental deployment and satisfy the requirement that a flat rate service be maintained. It is shown in [141] how an audio application can use such a service.

VII. Application-based QoS Control

In the preceding sections we reviewed the state of the art regarding the provision of QoS within the network. However, in spite of recent advances in the design and evaluation of QoS mechanisms for open loop and closed network traffic, little has been deployed within the Internet. This is due to a number of economical and technical reasons that are beyond the scope of this paper to explore. One consequence of this is that many ISPs find it easier and more economical to over provision their backbone networks in order to provide QoS. A second consequence, which is the topic of this section, has been the development and deployment of a wide array of *application-level* mechanisms outside of the network core for providing QoS. This array of mechanisms rely on one or both of the following simple ideas:

- the introduction of application-level routing and caching within the network,
- the introduction of redundancy and quality adaptation to deal with end-to-end loss and delay variations

We review these techniques, paying particular attention to the use of redundancy and quality adaptation in the context of networked audio applications.

A. Application-level Caching and Routing

One method for dealing with delays due to congested end-to-end paths between servers and clients is to cache web objects close to the client [142], [143]. This is the primary rationale for the establishment of content distribution networks (CDNs) such as the Akamai network. Such a network can consist of 100s or even thousands of servers that cache web objects close to the clients. These servers create a logical topology and establish routes within this topology to avoid congested links in the network.

Caching is also useful for the delivery of video streams. Unlike traditional web objects it is unnecessary to cache the entire video near the client [144]. For example, it may suffice to *cache a prefix of the video (first several seconds)* locally. This can produce a low startup latency while providing sufficient time to initiate streaming the remainder of the video from the server and the opportunity to handle poor network connectivity between the server and cache. This idea was first studied in [145]

More recently, there have been proposals to establish application-level networks for other applications such as teleconferencing, video streaming, etc. Underlying these

efforts is the recognition that the Internet interdomain routing algorithm, BGP (Border Gateway Protocol) [146], is not always able to provide good quality routes between domains. This can be due to policy reasons or because of the inability of BGP to account for performance when establishing routes. In addition, due to the size of the Internet as measured by the number of domains, BGP is not able to quickly recover from a router/link failure. The establishment of a route following such an event can take 10s to 100s of seconds [147]. These problems have motivated the commercial development of application-level overlay networks such as that deployed by InterNap as well as academic research into such networks [148].

A third impetus behind the development of application-level routing is the lack of a widely deployed multicast infrastructure. This has motivated the the development of a variety of application-level multicast algorithms [149], [150].

B. Redundancy and Quality Adaptation

In the mid 90s it was not uncommon for audio and video applications to encounter end-to-end path loss rates on the order of 10-40%. This stimulated the introduction of redundancy for the purpose of reducing the loss rate seen by the application. One scheme, PET (priority encoding transmission), explored the use of block codes for improving the quality of an MPEG-1 stream [151]. Briefly, different levels of protection were provided to I, P, and B frames in accordance to their importance to the application. Experiments reported in [151] demonstrated that the loss rate seen by an application can be significantly reduced. There are, however, a couple of problems with this approach. The reduced loss rate to the application comes at the cost of increased bandwidth. Thus, in the case where a single video application uses this technique, other applications sharing the network with it suffer a performance degradation. If all applications traversing the congested part of the network use this technique, then they will all observe higher loss rates. Because of the different levels of protection given to different parts of the video stream, this might or might not result in degraded quality as perceived by the application.

These problems have been resolved in an approach first proposed in [152] and refined in [153] in the context of networked audio. The basic idea is to systematically introduce redundancy for the purpose of improving audio quality while satisfying a bandwidth constraint. We describe this approach in the next subsection.

Before proceeding to the problem of adaptive quality enhancement for audio applications, we point out that the addition of redundancy can reduce bandwidth usage for a multicast application, i.e., one where one node (source) sends data to two or more other nodes (receivers) Studies have shown that the use of block erasure codes (e.g., Reed-Solomon codes) in a multicast setting is very effective in reducing bandwidth usage. This is easy to understand. A block code groups packets into groups of size n. The encoder adds an additional k parity packets to each group. The receiver can decode all n data packets provided that it receives any n of the combined $n + k$ data/parity packets. Consider a setting where several receivers each lose one data packet. In this case a single parity packet is sufficient to allow the receivers to recover all of the data packets, even when they have lost different packets. In the absence of parity packets, a retransmission of all of the missing data packet would have been required. More detailed treatments in the case of reliable delay insensitive data transmission and delay sensitive transmission can be found in [154] and [155] respectively.

C. Adaptive Redundancy and Quality for Audio Applications

We describe an approach that relies on the ability to encode audio and video at different qualities and different bandwidths. This provides the opportunity for an audio/video application to trade off encoding quality with level of redundancy while satisfying a bandwidth constraint. The basic paradigm is as follows:

- monitor network behavior (e.g., loss rate, delay jitter). This could result in periodic reports to the application or reports triggered by notable changes in network conditions.
- increase/decrease redundancy level as a consequence of changes in network behavior. This would include changes in encoding qualities.

We will make this concrete in the context of an FEC scheme recently standardized by the IETF for IP telephony [156], [153].

Consider an audio source that constructs samples spanning intervals of time of length L, encodes them and places them into packets that are periodically transmitted with period L. Suppose that the source can encode a sample at a rate $x \in [r_0, \infty)$ and that the quality of the encoded sample is given by a function $f : \mathbb{R}^+ \to \mathbb{R}$, which is increasing and concave. [152] proposed that each audio sample be encoded multiple times, each encoding at a different rate from the others, and transmitted to the receiver. In the case that K encoded versions of the sample are created, each packet would contain one version of each of K distinct audio samples. These include a version of the most recently generated sample along with versions of the preceding $K - 1$ samples (see Figure 9 for an example

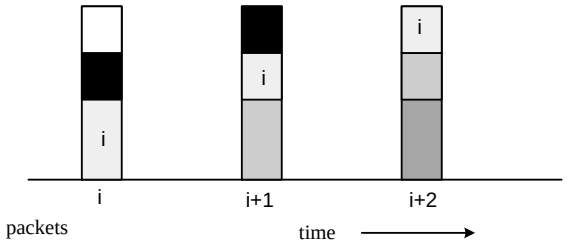


Fig. 9. Enhanced QoS through transmission of three copies of audio sample encoded at different rates.

with $k = 3$). This redundancy can significantly reduce the loss rate seen by the audio application. In order to ensure that an adaptive audio application not impact the quality of other applications, [152], [153] proposed that the encoding rate satisfy a bandwidth constraint R. This constraint can be imposed in any number of ways. For example, it could be imposed statically at session establishment or dynamically in response to congestion.

Two important questions need to be addressed with this scheme. The first regards the level of redundancy and the second the encoding rate for the K versions. A simple design rule for the level of redundancy is to use the maximum delay, D, that can be tolerated by the audio application along with the sample length L, $K \approx D/L$. We focus now on the second question.

Consider a source that transmits voice packets to a receiver over an unreliable network characterized by a stationary loss process (as might be described by a two-state Markov chain). Consider a typical audio sample. Let $Y = (Y_1, \ldots, Y_K)$ be transmission outcome vector for the sample, i.e., $Y_k = 0$ denotes the loss of the k-th version, $Y_k = 1$ otherwise. The design problem is

$$\begin{aligned} \text{Maximize} \quad & \textstyle\sum_{y \in \{0,1\}^K} P(Y = y) \max_{k \in \mathcal{S}} y_k f(x_k), \\ \text{s.t.} \quad & x_k \geq r_0, \qquad k = 1, \ldots, K \\ & \textstyle\sum_{k=1}^{K} x_i \leq R \end{aligned}$$

where x_k is the rate at which the k-th version is encoded and $P(Y = y)$ is the stationary probability that the transmission outcome vector for the sample is $y \in \{0, 1\}^K$. In general, this is a hard problem. However, for the case of a Markovian two state loss process, [153] was able to establish ordering relationships among potential solutions. These relationships were exploited in the cases of $K = 2, 3, 4, 5$ to derive a simple algorithm for obtaining the optimal solution based on the parameters of the loss process. An interesting property exhibited by the solution is that the first and last versions of the audio sample should always be encoded at higher qualities than those of the remaining versions.

In practice, there are only a finite number of encoding rates available, i.e., $x_k \in \{r_0, r_1, \ldots, r_N\}$. These would correspond to the rates possible using various audio encoders such as LPC, GSM, PCM, etc. The optimal solutions obtained for the previously described problem can be used to generate near optimal solutions to the true problem. An extension of the approach consists in incorporating delay in the quality function function $f(\cdot)$, which is useful in trading off the the quality improvement obtained with FEC versus the delay penalty [141].

D. Summary

The search for application-level solutions to QoS has been driven by necessity. Unlike the work on network-based QoS, it has been pursued in an *ad hoc* manner. Although some very clever techniques have been developed, there is considerable room for improvement.

VIII. Conclusions and Challenges for the Future

In this paper we have surveyed recent advances in several selective areas of Internet QoS provisioning: various network calculi and theories for deterministic, stochastic and elastic services, architecture and solutions for scalable QoS support, service differentiation within best effort and adaptive application QoS control. Where appropriate, we have also pointed out further research issues in these areas. There are a large number of other important research areas related to Internet QoS we did not cover. Examples are QoS pricing, in particular, congestion pricing, QoS and constraint-based routing, MPLS and traffic engineering. Some of these areas are nascent and still developing. In any case, reporting advances in these areas probably will require another survey paper.

It is evident from the research results we surveyed here that overall and collectively we have made great strides, in both theory and practice, toward building a QoS-capable Internet. We have gained fundamental understanding of what is achievable; we have also developed many required solutions and technologies. Despite all this progress, however, we have not, *as yet*, seen wide-spread deployment of QoS services. There are probably a variety of factors that hinder the deployment of Internet QoS, many of which are *not* technical *but* economic and political. Nonetheless, this "under-achievement" of Internet QoS should prompt us to re-think some of the fundamental challenges in Internet QoS and adjust our research focuses accordingly. As an initial effort to induce further discussion and debate on this critical subject, we conclude this paper with a short list of research challenges for the future that the authors *personally* think are important to Internet QoS but have not

been adequately researched.

A first category of challenges for implementing the theories and calculi presented here is their complexity: computational and informational. Their computational complexity refers to the amount of computation needed to predict performance of new or existing traffic, and is especially critical for short-term decisions, such as admission control of a flow or a service level agreement. The informational complexity refers to the fact that the network models need a potentially large amount of up-to-date information such as scheduling and queue management configurations at all network elements, characterization of all flows, routing of flows, many of them having frequent changes. Therefore, the management system for providing QoS guarantees in a sizable network is likely to be complex, expensive to build and to manage. Other network models that provide looser bounds on QoS levels may be able to trade off network efficiency or level of QoS assurance for a simpler QoS management system. The tradeoff between complexity and efficiency in network models is an open research area.

In this survey, we have presented many significant advances in the theory and mechanisms mostly related to the performance aspects of network data plane. Less researched is the control plane aspect of QoS provisioning such as signaling and bandwidth broker, briefly described in Section III-B. Any solution incurs a certain complexity of operation (such as volume of control traffic and processing overhead) that can be traded off with precision of reservations and network efficiency. Other critical aspects of QoS management systems such as accounting and billing were also not covered. The scalability and efficiency of such systems are also open for research.

Complementary to the performance aspects of Internet QoS provisioning are availability, reliability and security. Techniques for redundant provisioning of resources have been well studied in other contexts such as circuit switching networks, but not as much in the context of data networks. While data confidentiality, integrity and protection against denial of service attacks are security issues for both best-effort and QoS enabled networks, other security issues are specific to QoS networks, such as protection against service theft.

The business and economic aspects of QoS services require special consideration and research. The issue of recovering the cost of QoS provisioning (cost of reserved resources and associated complex network management) has been frequently invoked by network operators as a major hurdle in front of QoS deployment. There is a fundamental trade-off between service performance and its associated complexity and cost, and research is ongoing for finding the balance between the cost and acceptable price. The problem is further complicated by the need for inter-operator agreements on dividing the costs and benefits for services spanning multiple network domains.

Last but not least, new service paradigms have recently emerged that may have implications on the methods to provide QoS services. For example, content distribution networks and "application-level" service overlay networks attempt to improve service offering via techniques such as data replication, load balancing and routing using application-level mechanisms. Combining known techniques for QoS provisioning with such overlaid networks is a challenging area for research, but may have the benefit of bypassing the complicated inter-domain issues [157], [158].

REFERENCES

[1] R. Braden, D. Clark, and S. Shenker, "Integrated services in the internet architecture: an overview," RFC1633, June 1994.

[2] S. Blake et al, "An architecture for differentiated services," RFC2475, Dec 1998.

[3] P. Ferguson and G. Houston, *Quality of Service: Delivering QoS on the Internet and in Corporate Networks*, John Wiley & Sons, 1998.

[4] Z. Wang, *Internet QoS: Architectures and Mechanisms for Quality of Service*, Morgan Kaufmann, 2001.

[5] C.S. Chang, *Performance Guarantees in Communication Networks*, Springer-Verlag, New York, 2000.

[6] J.-Y. Le Boudec and P. Thiran, *Network Calculus*, Springer Verlag Lecture Notes in Computer Science volume 2050 (available online at http://icawww.epfl.ch), July 2001.

[7] A. 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. Networking, vol 1-3*, pp. 344–357, June 1993.

[8] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The multiple node case," *IEEE/ACM Trans. Networking, vol 2-2*, pp. 137–150, April 1994.

[9] P. Goyal and H. Vin, "Generalized guaranteed rate scheduling algorithms: a framework," *IEEE/ACM Trans. Networking, vol 5-4*, pp. 561–572, August 1997.

[10] C Kalmanek, H. Kanakia, and R. Restrick, "Rate controlled servers for very high speed networks," in *IEEE Globecom'90, vol 1*, 1990, pp. 12–20.

[11] H. Zhang and D. Ferrari, "Rate controlled service disciplines," *Journal of High Speed Networks*, vol. 3 No 4, pp. 389–412, August 1994.

[12] L. Georgiadis, R. Guérin, V. Peris, and R. Rajan, "Efficient support of delay and rate guarantees in an internet," in *Proceedings of Sigcomm'96*, August 1996, pp. 106–116.

[13] H. Zhang., "Service disciplines for guaranteed performance service in packet switching networks," *Proceedings of the IEEE*, 1996.

[14] C. S. Chang, "Stability, queue length and delay, Part I: deterministic queuing networks," Tech. Rep. Technical Report RC 17708, IBM, 1992.

[15] R.L. Cruz, "A calculus for network delay, Part I: network ele-

ments in isolation," *IEEE Trans. Inform. Theory, vol 37-1*, pp. 114–131, January 1991.

[16] R.L. Cruz, "A calculus for network delay, Part II: network analysis," *IEEE Trans. Inform. Theory, vol 37-1*, pp. 132–141, January 1991.

[17] C.S. Chang, "On deterministic traffic regulation and service guarantee: A systematic approach by filtering," *IEEE Transactions on Information Theory*, vol. 44, pp. 1096–1107, August 1998.

[18] J.-Y. Le Boudec, "Application of network calculus to guaranteed service networks," *IEEE Transactions on Information Theory*, vol. 44, pp. 1087–1096, May 1998.

[19] R. Agrawal, R. L. Cruz, C. Okino, and R. Rajan, "Performance bounds for flow control protocols," *IEEE/ACM Transactions on Networking (7) 3*, pp. 310–323, June 1999.

[20] S. Shenker, C. Partridge, and R. Guerin, "Specification of guaranteed quality of service," RFC2212, Sep 1997.

[21] P. E. Boyer, M. J. Servel, and F. P. Guillemin, "The spacer-controller: an efficient upc/npc for atm networks," in *ISS '92, Session A9.3, volume 2*, October 1992.

[22] J.-Y. Le Boudec, "Some properties of variable length packet shapers," in *Proc ACM Sigmetrics / Performance '01*, 2001.

[23] R.L. Cruz, "Quality of service guarantees in virtual circuit switched networks," *IEEE JSAC*, pp. 1048–1056, August 1995.

[24] F. Baccelli, G. Cohen, G.J. Olsder, and J.-P. Quadrat, *Synchronization and Linearity, An Algebra for Discrete Event Systems*, John Wiley and Sons, 1992.

[25] S. Giordano J.-Y. Le Boudec, P. Thiran, "A short tutorial on network calculus II: min-plus system theory applied to communication networks," in *Proceedings of ISCAS 2000*, Geneva, May 2000, pp. IV–97 – IV–100.

[26] L. Georgiadis, Gurin R., and Peris V., "Efficient network provisioning based on per node traffic shaping," *IEEE/ACM Transactions on Networking*, vol. 4, pp. 482–501, 1996.

[27] Agrawal R. and Rajan R., "A general framework for analyzing schedulers and regulators in integrated services network," in *34th Allerton Conf of Comm., Cont., and Comp. Monticello, IL*, Oct 1996, pp. 239–248.

[28] R. L. Cruz, "Lecture notes on quality of service guarantees," 1998.

[29] D. Stiliadis and A. Varma, "Rate latency servers: a general model for analysis of traffic scheduling algorithms," in *IEEE Infocom '96*, 1991, pp. 647–654.

[30] R. Guérin and V. Peris, "Quality-of-service in packet networks - basic mechanisms and directions," *Computer Networks and ISDN, Special issue on multimedia communications over packet-based networks*, 1998.

[31] H. Sariowan, "A service curve approach to performance guarantees in integrated service networks," 1996, Ph.D. Dissertation, UCSD.

[32] H. Sariowan, R.L. Cruz, and G.C. Polyzos, "Scheduling for quality of service guarantees via service curves," in *Proceedings ICCCN'95*, Sept 1995, pp. 512–520.

[33] S. Giordano and J.-Y. Le Boudec, "On a class of time varying shapers with application to the renegotiable variable bit rate service," *Journal on High Speed Networks*, vol. 9, no. 2, pp. 101–138, June 2000.

[34] J.-Y. Le Boudec and A. Charny, "Packet scale rate guarantee for non-FIFO nodes," Tech. Rep. DSC200138, EPFL-DSC, http://dscwww.epfl.ch./EN/publications/documents/tr01_038.pdf, July 2001.

[35] V. Jacobson, K. Nichols, and K. Poduri, "An expedited forwarding PHB," June 1999, RFC 2598, IETF.

[36] I. Chlamtac, A. Faragó, H. Zhang, and A. Fumagalli, "A deterministic approach to the end-to-end analysis of packet flows in connection oriented networks," *IEEE/ACM transactions on networking*, vol. (6)4, pp. 422–431, 08 1998.

[37] J.-Y. Le Boudec and G. Hebuterne, "Comment on a deterministic approach to the end-to-end analysis of packet flows in connection oriented network," *IEEE/ACM Transactions on Networking*, February 2000.

[38] H. Zhang, "A note on deterministic end-to-end delay analysis in connection oriented networks," in *Proc of IEEE ICC'99, Vancouver, pp 1223–1227*, 1999.

[39] J. C. R. Bennett, K. Benson, A. Charny, W. F. Courtney, and J.-Y. Le Boudec, "Delay jitter bounds and packet scale rate guarantee for expedited forwarding," in *Proceedings of Infocom*, April 2001.

[40] A. Charny and J.-Y. Le Boudec, "Delay bounds in a network with aggregate scheduling," in *First International Workshop on Quality of future Internet Services*, Berlin, Germany, September 2000.

[41] J.-Y. Le Boudec and F. Farkas, "A delay bound for a network with aggregate scheduling," in *Proceedings of the Sixteenth UK Teletraffic Symposium on Management of Quality of Service*, Harlow, UK, May 2000, p. 5.

[42] R. Agrawal, R. L. Cruz, C. Okino, and R. Rajan, "A framework for adapative service guarantees," in *Proc. Allerton Conf on Comm, Control and Comp, Monticello, IL*, Sept 1998.

[43] D. Verma, H. Zhang, and D. Ferrari, "Guaranteeing delay jitter bounds in packet switching networks," in *Proceedings of Tricomm '91, Chapel Hill*, April 1991, pp. 35–46.

[44] R. L. Cruz, "SCED+: efficient management of quality of service guarantees," in *IEEE Infocom'98, San Francisco*, March 1998.

[45] Z.-L. Zhang, Z. Duan, and T.Y. Hou, "Fundamental trade-offs in aggregate packet scheduling (extended abstract)," in *SPIE Vol. 4526*, August 2001.

[46] J.-Y. Le Boudec and Olivier Verscheure, "Optimal smoothing for guaranteed service," *IEEE/ACM Transactions on Networking*, vol. 8, no. 6, pp. 689–696, December 2000.

[47] Lothar Thiele, Samarjit Chakraborty, and Martin Naedele, "Real-time calculus for scheduling hard real-time systems," in *ISCAS*, Geneva, May 2000.

[48] J. Salehi, Z.-L. Zhang, J. Kurose, and D. Towsley, "Optimal smoothing of stored video and its imapct on network resource requirements," *IEEE/ACM Transaction on Network*, vol. 6, no. 4, pp. 397–410, August 1998.

[49] J. Rexford and D. Towsley, "Smoothing variable bit rate video in an internetwork," *IEEE/ACM transactions on networking*, vol. (7)23, pp. 202–215, 04 1999.

[50] Olivier Verscheure, Pascal Frossard, and J.-Y. Le Boudec, "Joint smoothing and source rate selection for guaranteed service networks," in *Proceedings of Infocom 2001*, Anchorage, USA, April 2001.

[51] S. Sen, D. Towsley, Z.-L. Zhang, and J. Dey, "Optimal multicast smoothing of streaming video over the internet," *IEEE Journal on Selective Areas in Communications, Special Issue on Internet Proxy Services*, 2002, Accepted for Publication. An earlier abridged version can be found in *Proc. IEEE INFOCOM'99*.

[52] I. Stoica, H. Zhang, S. Shenker, R. Yavatkar, D. Stephens, A. Malis, Y. Bernet, Z. Wang, F. Baker, J. Wroclawski, C. Song, and R. Wilder, "Per hop behaviors based on dynamic packet states," Internet Draft, February 1999, Work in Progress.

[53] Z.-L. Zhang, Z. Duan, and Y. T. Hou, "Virtual time reference system: A unifying scheduling framework for scalable support

of guaranteed services," *IEEE Journal on Selected Areas in Communication*, Special Issue on Internet QoS, December 2000.

[54] I. Stoica and H. Zhang, "Core-stateless fair queueing: Achieving approximately fair bandwidth allocations in high speed networks," in *SIGCOMM Symposium on Communications Architectures and Protocols*, BC, Cananda, August 1998.

[55] I. Stoica and H. Zhang, "Providing guaranteed services without per flow management," in *SIGCOMM Symposium on Communications Architectures and Protocols*, Boston, MA, August 1999.

[56] Z.-L. Zhang, Z. Duan, and Y. T. Hou, "Fundamental trade-offs in aggregate packet scheduling," in *Proceedings of IEEE ICNP*, Riverside, CA, November 2001.

[57] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource reservation protocol (RSVP) – version 1 functional specification," RFC 2205, Sept. 1997.

[58] R. Guerin, S. Blake, and S. Herzog, "Aggregating RSVP-based QoS requests," Internet Draft, November 1997, Work in Progress.

[59] F. Baker, B. Davie, F. L. Faucheur, and C. Iturralde, "RSVP reservations aggregation," Internet Draft, Mar. 2000, Work in Progress.

[60] O. Schelen and S. Pink, "Aggregating resource reservation over multiple routing domains," in *Proc. of Fifth IFIP International Workshop on Quality of Service (IWQoS'99)*, Cambridge, Englang, June 1999.

[61] P. Pan and H. Schulzrinne, "YESSIR: a simple reservation mechanism for the internet," in *Proc. of International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSSDAV'98)*, Cambridge, Englang, July 1998.

[62] W. Almesberger, T. Ferrari, and J.-Y. Le Boudec, "SRP: a scalable resource reservation protocol for the internet," in *Proc. of Fifth IFIP International Workshop on Quality of Service (IWQoS'98)*, Napa, CA, May 1998.

[63] H. Schulzrinne, S. Casner, R. Frederick, and S. McCane, "RTP: A transport protocol for real-time applications," Request for Comments RFC 1889, Network Working Group, 1994.

[64] V. Elek, G. Karlsson, and R. Ronngre, "Admission control based on end-to-end measurements," in *Proc. of the IEEE INFOCOM*, Tel Aviv, Israel, Mar. 2000.

[65] G. Bianchi, A. Capone, and C. Petrioli, "Throughput analysis of end-to-end measurement-based admission control," in *Proc. of the IEEE INFOCOM*, Tel Aviv, Israel, Mar. 2000.

[66] C. Cetinkaya and E. Knightly, "Egress admission control," in *Proc. of the IEEE INFOCOM*, Tel Aviv, Israel, Mar. 2000.

[67] L. Breslau, E. Knightly, S. Shenker, I. Stoica, and H. Zhang, "Endpoint admission control: Architectural issues and performance," in *SIGCOMM Symposium on Communications Architectures and Protocols*, Stockholm,Sweden, August 2000.

[68] R. Gibbens and F. P. Kelly, "Distributed connection acceptance control for a connectioness network," in *Teletraffic Engineering in a Competitive World, Proceedings ITC 16*, P. Key and D. Smith, Eds. June 1999, pp. 941–952, Elsevier.

[69] F. Kelly, P. Key, and S. Zachary, "Distributed admission control," *IEEE Journal on Selected Areas in Communication*, Special Issue on Internet QoS, December 2000.

[70] R. Gibbens and P. Key, "Distributed control and resource marking using best-effort routers," *IEEE Network Magazing*, pp. 54–59, May/June 2001.

[71] K. Nichols, V. Jacobson, and L. Zhang, "A two-bit differentiated services architecture for the internet," RFC 2638, July 1999.

[72] A. Terzis, L. Wang, J. Ogawa, and L. Zhang, "A two-tier resource management model for the internet," in *Global Internet 99*, December 1999.

[73] Z.-L. Zhang, Z. Duan, L. Gao, and Y. T. Hou, "Decoupling QoS control from core routers: A novel bandwidth broker architecture for scalable support of guaranteed services," in *SIGCOMM Symposium on Communications Architectures and Protocols*, Stockholm,Sweden, August 2000.

[74] Z.-L. Zhang, Z. Duan, and Y. T. Hou, "On scalable design of bandwidth brokers," *IEICE Transaction on Communications*, vol. E84-B, no. 8, August 2001.

[75] D. Anick, D. Mitra, and M. M. Sondhi, "Stochastic theory of a data handling system with multiple sources," *Bell System Technical Journal*, vol. 61, no. 8, pp. 1871–1894, 1982.

[76] A. Lombardo, G. Morabito, and G. Schembra, "A novel analytical framework compounding statistical traffic modeling and aggregate-level service curve disciplines: Network performance and efficiency implications," .

[77] I. Norros, "On the use of fractional brownian motion in the theory of connectionless networks," *IEEE Journal on Selected Areas of Communications*, pp. 953–962, August 1995.

[78] J. Roberts, Mocci U., and Virtamo J. (Eds), *Broadband Network Traffic, Final report of action COST 242*, Springer Verlag – Lecture Notes in Computer Science volume 1155, 1996.

[79] T. Bonald, A. Proutiére, and J. Roberts, "Statistical performance guarantees for streaming flows using expedited forwarding," in *Proc. of IEEE INFOCOM'2001*, March 2001.

[80] A. Elwalid, D. Mitra, and R. Wenworth, "A new approach for allocating buffers and bandwidth to heterogeneous, regulated traffic in ATM node," *IEEE Journal of Selected Areas in Communications*, vol. 13, pp. 1048–1056, August 1995.

[81] F. Lo Presti, Z.-L. Zhang, D. Towsley, and J. Kurose, "Source time scale and optimal buffer/bandwidth trade-off for regulated traffic in a traffic node," *IEEE/ACM Transactions on Networking*, vol. 7, pp. 490–501, August 1999.

[82] J.-Y Le Boudec and A. Ziedins, "A CAC algorithm for VBR connections over a VBR trunk," in *Proceedings of ITC '97*, June 1997.

[83] K. Kumaran and M. Mandjes, "Multiplexing regulated traffic streams: Design and performance," in *Proc. of IEEE INFOCOM'01*, 2001.

[84] G. Kesidis and T. Konstantopoulos, "Worst-case performance of a buffer with independent shaped arrival processes," *IEEE Communication Letters*, 2000.

[85] C.-S. Chang, W. Song, and Y. m. Chiu, "On the performance of multiplexing independent regulated inputs," in *Proc. of Sigmetrics 2001*, Massachusetts, USA, May 2001.

[86] M. Vojnovic and J.-Y. Le Boudec, "Bounds for independent regulated inputs multiplexed in a service curve network element," in *Proc. Globecom 2001*, November 2001.

[87] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *American Statistical Association Journal*, pp. 13–30, March 1963.

[88] M. Vojnovic and J.-Y. Le Boudec, "Stochastic analysis of some expedited forwarding networks," Tech. Rep. DSC200139, EPFL-DSC, http://dscwww.epfl.ch./EN/publications/documents/tr01_039.pdf, July 2001.

[89] C. M. Chuang and J. F. Chang, "Deterministic loss ratio quality of service guarantees for high speed networks," *IEEE Communication Letters*, vol. 4, no. 7, pp. 236–238, July 2000.

[90] Nikolay Likhanov and Ravi R. Mazumdar, "Cell loss asymptotics in buffers fed with a large number of independent stationary sources," in *Proc. of IEEE INFOCOM'98*, 1998.

[91] I. Norros and J. Virtamo, "Who loses cells in the case of burst scale congestion," 1991.

[92] T. Konstantopoulos and G. Last, "On the dynamics and performance of stochastic fluid systems," *Journal of Applied Probability*, vol. 37, pp. 652–667, 2000.

[93] R. Boorstyn, A. Burchard, J. Liebeherr, and C. Oottamakorn, "Statistical service assurances for traffic scheduling algorithms," *IEEE Journal on Selected Areas in Communications, Special Issue on Internet QoS*, 2000.

[94] J. Liebeherr, S. D. Patek, and A. Burchard, "A calculus for end-to-end statistical service guarantees," Tech. Rep. CS-2001-19, University of Virginia, Department of Computer Science, August 2001.

[95] O. Yaron and M. Sidi, "Calculating performance bounds in communication networks," in *Proc. of the IEEE INFOCOM*, Apr. 1993, pp. 539–546.

[96] Z.-L. Zhang, D. Towsley, and J.F. Kurose, "Statistical network performance guarantees with generalized processor sharing scheduling," *IEEEJournal on Selected Areas in Communications (JSAC)*, August 1995.

[97] D. Starobinski and M. Sidi, "Stochastically bounded burstiness for communication networks," *IEEE Transactions on Information Theory*, vol. 46, no. 1, pp. 206–212, January 2000.

[98] W. Stevens, *TCP/IP Illustrated, Vol.1 The Protocols*, Addison-Wesley, 1994.

[99] W. Stevens, "TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms," RFC2001, Jan 1997.

[100] M. Allman, V. Paxson, and W. Stevens, "TCP Congestion Control," RFC2581, April 1999.

[101] K. Thompson, G. Miller, and M. Wilder, "Wide-area internet traffic patterns and charateristics," *IEEE Network*, vol. 11, no. 6, November-December 1997, Extended version at http://www.vbns.org/presentations/papers/MCItraffic.ps.

[102] S. McCreary and K. Claffy, "Trends in wide area IP traffic patterns - a view from Ames internet exchange," in *13th ITC Specialist Seminar on Internet Traffic Measurement and Modelling*, 2000, http://www.caida.org/outreach/papers/AIX0005/.

[103] V. Jacobson and M. J. Karels, "Congestion Avoidance and Control," in *SIGCOMM'88*, 1988.

[104] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgements options," RFC2018, Oct 1996.

[105] M. Mathis and J. Mahdavi, "Forward acknowledgement: Refining TCP congestion control," in *ACM Computer Communication Review*, Oct 1996.

[106] T. Ott, J. Kemperman, and M. Mathis, "The stationary behavior of ideal TCP congestion avoidance," 1996, Unpublished. http://www.argreenhouse.com/papers/tjo/TCPwindow.pdf.

[107] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *Comupter Communication Review*, July 1997.

[108] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Reno performance: A simple model and its empirical validation," *IEEE/ACM Transactions on Networking*, vol. 8, no. 2, pp. 133–145, April 2000.

[109] J. Widmer and M. Handley, "Extending equation-based congestion control to multicast applications," in *ACM SIGCOMM*, August 2001.

[110] C. Cunha, A. Bestavros, and M. Crovella, "Characteristics of WWW client-based traces," Tech. Rep. BU-CS-95-010, Boston University, July 1995.

[111] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency," in *IEEE INFOCOM*, 2000, pp. 1742–1751.

[112] V. Firoiu and M. Borden, "A study of active queue management for congestion control," in *IEEE INFOCOM*, 2000, pp. 1435–1444.

[113] A. Misra, J. Baras, and T. Ott, "The window distribution of multiple TCPs with random loss queues," in *GLOBECOM'99*, December 1999.

[114] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, August 1997.

[115] V. Firoiu, I. Yeom, and X. Zhang, "A framework for practical performance evaluation and traffic engineering in IP networks," in *IEEE ICT*, June 2001, http://www-net.cs.umass.edu/~vfiroiu/papers/temodel-conf.pdf.

[116] T. Bu and D. Towsley, "Fixed point approximations for TCP behavior in an AQM network," in *ACM SIGMETRICS*, June 2001.

[117] I. Yeom and A. Reddy, "Modeling TCP behavior in a differentiated-services network," *IEEE/ACM Transactions on Networking*, vol. 9, no. 1, pp. 31–46, April 2001.

[118] R. Gibbens, S. Sargood, C. Van Eijl, F. Kelly, H. Azmoodeh, R. Macfadyen, and N. Macfadyen, "Fixed-point models for the end-to-end performance analysis of IP networks," in *13th ITC Special Seminar: IP Traffic Management, Modeling and Management*, 2000.

[119] V. Misra, W. Gong, and D. Towsley, "Stochastic differential equation modeling and analysis of TCP-windowsize behavior," in *Performance*, 1999.

[120] E. Altman, K. Avrachenkov, and C. Barakat, "A stochastic model of TCP/IP with stationary random losses," in *ACM SIGCOMM'00*, August 2000.

[121] V. Misra, W. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," in *ACM SIGCOMM*, 2000, pp. 151–160.

[122] C. Hollot, V. Misra, D. Towsley, and W. Gong, "A control theoretic analysis of RED," in *IEEE INFOCOM*, 2000.

[123] C. Hollot, V. Misra, D. Towsley, and W. Gong, "On designing improved controllers for AQM routers supporting TCP flows," in *IEEE INFOCOM*, 2000.

[124] V. Firoiu, X. Zhang, and Y. Guo, "Best effort differentiated services: Trade-off service differentiation for elastic applications," in *IEEE ICT*, June 2001, http://www-net.cs.umass.edu/~vfiroiu/papers/beds-conf.pdf.

[125] K. K. Ramakrishnan and S. Floyd, "A proposal to add explicit congestion notification (ECN) to IP," RFC2481, Jan 1999.

[126] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin, "REM: Active queue management," *IEEE Network*, May-June 2001.

[127] S. Kunniyur and R. Srikant, "Analysis and design of an adaptive virtual queue algorithm for active queue management," in *Proc. ACM Sigcomm 2001*, 2001.

[128] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," *IEEE/ACM Transactions on Networking*, vol. 7, no. 4, pp. 458–472, 1999.

[129] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *SIGCOMM'2000*, August 2000.

[130] M. Vojnovic and J.-Y. Le Boudec, "Some observations on equation-based rate control," in *Proceedings of ITC-17, Seventeenth International Teletraffic Congress*, Salvador da Bahia, Brazil, September 2001.

[131] D. Bansal and H. Balakrishnan, "Binomial congestion control algorithms," in *Proc. INFOCOM 2001*, 2001.

[132] J. Padhye and S. Floyd, "On inferring TCP behavior," in *ACM SIGCOMM*, 2001.

[133] J. Wrocklawski, "Specification of the controlled-load network element service," RFC2211, Sep 1997.

[134] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured forwarding PHB group," June 1999, RFC 2597, IETF.

[135] S. Sahu, P. Nain, C. Diot, V. Firoiu, and D. Towsley, "On achievable service differentiation with token bucket marking for TCP," in *Measurement and Modeling of Computer Systems*, 2000, pp. 23–33.

[136] Y. Chait, C. Hollot, V. Misra, D. Towsley, and H. Zhang, "Providing throughput differentiation for TCP flows using adaptive two color marking and multi-level AQM," Tech. Rep., University of Massachussetts, Amherst, 2001, submitted for publication.

[137] K. Kilkki and J. Ruutu, "Simple integrated media access - an internet service based on priorities," in *6th International Conference on Telecommunication Systems*, 1998.

[138] C. Dovrolis, D. Stiliadia, and P. Ramanathan, "Proportional differentiated services: Delay differentiation and packet scheduling," in *ACM SIGCOMM*, 1999.

[139] Y. Moret and S. Fdida, "A proportional queue control mechanism to provide differentiated services," in *International Symposium on Computer Systems*, October 1998.

[140] P. Hurley, M. Kara, J.-Y. Le Boudec, and P. Thiran, "Abe: Providing a low-delay service within best effort," *IEEE Network Magazine*, vol. 15, no. 3, May 2001.

[141] J. Y. Le Boudec C. Boutremans, "Adaptive delay aware error control for internet telephony," in *Proceedings of 2nd IP-Telephony workshop*, Columbia University, New York, April 2001, pp. 81–92.

[142] J. Gwertzman and M. Seltzer, "The case for geographical push-caching," in *HotOS Conference*, 1994.

[143] A. Bestavros, R.L. Carter, M.E. Crovella, C.R. Cunha, A. Heddaya, and S.A. Mirdad, "Application-level document caching in the internet," in *SDNE'95*, 1995.

[144] Z.-L. Zhang, Y. Wang, David H.C. Du, and D. Su, "Video staging: A proxy-server-based approach to end-to-end video delivery over wide-area networks," *IEEE/ACM Transaction on Network*, vol. 8, no. 4, pp. 429–442, August 2000.

[145] S.Sen, J. Rexford, and D. Towsley, "Proxy prefix caching for multimedia streams," in *IEEE INFOCOM'99*, March 1999.

[146] J.W. Stewart, *BGP4: Inter-Domain Routing in the Internet*, Addison-Wesley, 1998.

[147] C. Labovitz, A. Ahuja, A. Abose, and F. Jahanian, "An experimental study of delayed internet routing convergence," in *ACM SIGCOMM'00*, Stockholm, Sweden, August 2000.

[148] D. Anderson, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *18th ACM SOSP*, Banff, Canada, October 2001.

[149] P. Francis, "Yoid: extending the interent multicast architecture," *preprint available from http://www.isis.edu/div7/yoid/*, April 2000.

[150] Y. Chu, S. Rao, and H. Zhang, "A case for end system multicast," in *ACM SIGMETRICS'00*, Santa Clara, June 2000.

[151] B. Lamparter, A. Albanese, M. Kalfane, and M. Luby, "PET - priority encoded transmission," Tech. Rep. TR-95-046, ICSI, August 1995.

[152] J.-C. Bolot and A.V. Garcia, "Control mechanisms for packet audio in the internet," in *IEEE INFOCOM'96*, San Francisco, CA, April 1996.

[153] J. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive fec-based error control for interactive audio in the internet," in *IEEE INFOCOM'99*, March 1999.

[154] J. Nonnenmacher, E.W. Biersack, and D. Towsley, "Parity-based loss recovery for reliable multicast transmission," *IEEE/ACM Transactions on Networking*, vol. 6, no. 4, pp. 349–361, August 1998.

[155] D. Rubenstein, J. Kurose, and D. Towsley, "A study of proactive hybrid FEC/ARQ and sclable feedback techniques for reliable, real-time multicast," *Computer Communications Journal*, vol. 24, pp. 563–574, 2001.

[156] C. Perkins et al., "RTP payload for redundant audio data," *Internet RFC 2198*, September 1997.

[157] Z.-L. Zhang, Z. Duan, and Y. T. Hou, "Service overlay networks: SLAs, QoS and bandwidth provisioning," Tech. Rep., University of Minnesota, July 2001, submitted for publication.

[158] Z. Duan, Z.-L. Zhang, and Y. T. Hou, "Bandwidth provisioning for service overlay networks," in *Proceedings of SPIE ITCOM Conference on Scalability and Traffic Control in IP Networks II*, July 2002.