Journal of VLSI Signal Processing 30, 71–87, 2002

Deterministic Time-Varying Packet Fair Queueing for Integrated Services Networks*

ANASTASIOS STAMOULIS AND GEORGIOS B. GIANNAKIS
Department of Electrical and Computer Engineering, 200 Union Street. S.E., University of Minnesota, Minneapolis, MN 55455, USA

Received September 13, 2000; Revised June 15, 2001

Abstract. In integrated services networks, the provision of Quality of Service (QoS) guarantees depends critically upon the scheduling algorithm employed at the network layer. In this work we review fundamental results on scheduling, and we focus on Packet Fair Queueing (PFQ) algorithms, which have been proposed for QoS wireline-wireless networking. The basic notion in PFQ is that the bandwidth allocated to a session is proportional to a positive weight ϕ_i. Because of the fixed weight assignment, the inherent in PFQ *delay-bandwidth coupling* imposes limitations on the range of QoS that can be supported. We develop PFQ with deterministic time-varying weight assignments, and we propose a low-overhead algorithm capable of supporting arbitrary piecewise linear service curves which achieve delay-bandwidth decoupling. Unlike existing service-curve based algorithms, our time-varying PFQ scheme does not exhibit the *punishment* phenomenon, and allows sessions to exploit the extra bandwidth in under-loaded networks.

Keywords: packet fair queueing, generalized processor sharing, QoS, service curves, integrated networks, wireless networks

1. Introduction

In integrated services networks, the provision of Quality of Service (QoS) guarantees to individual sessions depends critically upon the scheduling algorithm employed at the network switches. The scheduling algorithm determines the transmission order of packets in outgoing links and, thus, it has a direct impact on the packet delay and achievable throughput, which serve as primary figures of merit of the system performance. In wireline Computer Networks, scheduling is an area of fervent research for nearly two decades. In wireless networks, research of scheduling algorithms is in a relatively nascent form, as existing second generation systems carry mainly only voice and low-rate data traffic. However, the emergence of third generation systems, and their promise of broadband transmissions combined with the indiosyncracies of the wireless channel shed light on the importance of scheduling in wireless environments as well.

The primary mission of the scheduler is to allocate *efficiently* the system *resources* to the users. In computer networks, the term "resources" refers primarily to the bandwidth of the transmission link, and the queueing buffer space in the routers, whereas the adverb "efficiently" refers to making sure that resources are not wasted (for example, in a TDMA system, the scheduler would try to ensure that as many as possible timeslots are used for transmission). Integrated services networks do not only offer the potential of more efficient utilization of resources, but also exhibit billing advantages for both service providers and customers (see,

*Work in this paper was supported by the NSF Wireless Initiative grant no. 99-79443. Parts of this work were presented at the *Globecom'2000 Conference*. Original version was submitted to the *Journal of VLSI Signal Processing* on September 11, 2000. It was revised on June 6, 2001.

e.g., [1]). Hence, it does not come as a surprise that third generation wireless networks are envisioned to support, similar to their wireline counterparts, a plethora of different transmission rates and services. What perhaps comes as a surprise, is that the need for efficient bandwidth allocation is not only prominent in wireless networks (where the RF spectrum is scarce and extremely expensive), but in wireline networks as well: despite physical-layer developments in optical networking (for the backbone network) and HDSL (for the last-mile), Internet access is mainly best-effort (which implies, for example, that it is not known a priori how long it will take for a web page to be downloaded).

Network scheduling algorithms play a central role in how bandwidth is allocated among sessions—flows (in a wireline environment) or users (in a cellular network). From a historical standpoint, there is an extensive body of work in scheduling algorithms conducted by researchers in quite diverse fields, such as Operations Research, and Computer Science. In the emerging broadband multi-service networks, the Generalized Processor Sharing (GPS) [2] discipline and the numerous Packet Fair Queueing (PFQ) algorithms are widely considered as the primary scheduler candidates. This is because GPS has been shown to provide both minimum service rate guarantees and isolation from ill-behaved traffic sources. Not only have GPS-based algorithms been implemented in actual gigabit switches in wired networks, but also they have been studied in the context of the emerging broadband wireless networks (see, e.g., [3] and references therein).

The fundamental notion in GPS-based algorithms is that the amount of service session i receives from the switch (in terms of transmitted packets) is proportional to a positive weight ϕ_i. As a result, GPS (and its numerous PFQ variants) is capable of delivering bandwidth guarantees; the latter translate to delay guarantees as long as there is an upper bound on the amount of incoming traffic (this bound could be either deterministic for leaky-bucket constrained sessions [2], or stochastic, as in, e.g., [4]). One of the major shortcomings of GPS is that the service guarantees provided to a session i are controlled by just one parameter, the weight ϕ_i. Hence, the *delay-bandwidth coupling*, which refers to the mutual dependence between delay and throughput guarantees (i.e., in order to guarantee small delays, a large portion of the bandwidth should be reserved). To appreciate why the delay-bandwidth coupling is a shortcoming, one needs to take into consideration that future networks will support multirate multimedia services with widely diverse delay and bandwidth specifications. For example, video and audio have delay requirements of the same order, but video has an order of magnitude greater bandwidth requirements than audio. Therefore, delay-bandwidth coupling could lead to bandwidth underutilization.

To overcome these problems, [5] introduces the notion of service curves (SC); a SC $S_i(t)$ can be thought of as the minimum amount of service that the switch guarantees to session i in the interval $[0, t]$. SCs dispense with the delay-bandwidth coupling because the shape of $S_i(t)$ could be arbitrary. However, as noted in, e.g., [6], SC algorithms suffer from the *punishment* effect: when session i receives in $[0, t_1]$ more service than $S_i(t_1)$ (for example, this could happen if the system is under-loaded in $[0, t_1]$), and the load increases at t_1, then there is an interval $(t_1, t_2]$ where session i does *not* receive any service at all. Eventually session i will receive service at least equal to $S_i(t_2)$ in $[0, t_2]$, but, nevertheless, it is *penalized* for the extra service it received in $[0, t_1]$. From a practical point of view, the punishment phenomenon is undesirable in wireline networks, because it does not allow sessions to take advantage of potentially available bandwidth in the system. In wireless networks, where the quality of the physical channel is time-varying (i.e., sometimes the channel can be considered "good" and sometimes the channel is "bad"), the punishment effect could potentially not allow a user to transmit, even though the physical channel could be temporarily in a good state. Therefore, in both wireless and wireline environments, the punishment phenomenon is undesirable. As GPS does not suffer from the punishment problem [2], we are motivated to study whether GPS with proper weight assignment is capable of providing the same QoS services as SC-based algorithms, while obviating the punishment phenomenon in both wireline and wireless networks.

In this work we provide a short tutorial on scheduling algorithms and revisit some of the fundamental results on deterministic QoS. Furthermore, we develop a novel scheduling algorithm which combines the strengths of PFQ algorithms and SC-based algorithms. Our algorithm, called Deterministic Time-Varying Packet Fair Queueing (DTV-PFQ) relies on the intuitive idea of time-varying weights (ϕ_i's). Noting that in GPS (and hence in PFQ) the weight assignment is fixed throughout the lifetime of the sessions, herein we develop a time-varying weight assignment which extends PFQ and makes PFQ capable of supporting piecewise linear SCs with minimum overhead. Our contribution lies

in showing how the time-varying weight assignment can be done deterministically for each session and independently of the other sessions, thus preserving the isolation properties of PFQ. As a result, our deterministic time-varying PFQ is capable of combining the strengths of GPS and the service flexibility of SC-based algorithms. Moreover, illustrating that DTV-PFQ can support (modulo some approximation errors) as many multiple leaky-bucket constrained sessions as the Earliest Deadline First (EDF) scheduling algorithm constitutes an important ramification of our work. Noting that in wireless environments a lot is to be gained by the joint design across network layers, we also discuss how our scheduling algorithm can be integrated with the physical layer transmission-reception scheme.

The rest of the paper is organized as follows: in Section 2 we revisit results on GPS, EDF, PFQ, and SCs, and in Section 3 we discuss the punishment effect of SC-based schedulers. In Section 4 we describe our deterministic time-varying weight assignment scheme and illustrate its merits. In Section 5 we discuss implementation details in wireless–wireline networks, and finally, we provide concluding remarks and pointers to future research in Section 6.

2. Model Description

In this section we briefly review results on GPS, EDF, PFQ, and SCs, and describe the deterministic model that we will use in the following sections to develop and study DTV-PFQ. We consider a single network switch which multiplexes data packets sent by various sessions. The scheduler operates either in each of the output links of a network switch (see Fig. 1, [41]), or at the basestation inside a cell of a wireless network (see Fig. 2). The objective of the scheduler is to provide QoS guarantees to sessions (either to individual or to groups of sessions). The QoS guarantees can be expressed in terms of packet delay, achievable throughput, and packet drop probability (in wireline networks packets are dropped mainly because of buffer overflows due to congestion, whereas in wireless networks packets are dropped mainly because of symbol errors at the physical layer).

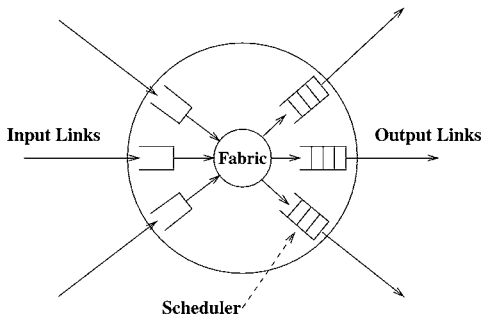

Figure 1. Network switch.

In order to quantify the achievable QoS guarantees, let us introduce some notation. $A_i(\tau, t)$ denotes the amount of traffic that session i transmits in the time interval $(\tau, t]$: $A_i(\tau, t)$ could either be a deterministic function of time (as e.g., in the case of a known MPEG file which is to be transmitted over the network), or a stochastic process [7]. $R_i(\tau, t)$ denotes the amount of service given to session i by the scheduler in the interval $(\tau, t]$; it readily follows that $R_i(\tau, t)/(t - \tau)$ is the average bandwidth allocated to the session i over the time interval $(\tau, t]$. Supposing that session i started transmitting packets at time 0, then the backlog $Q_i(t)$ of session i at time t, and the delay $D_i(t)$ of a packet of session i which arrives at the switch at time t are given respectively by [8]:

$$\begin{aligned} Q_i(t) &= A_i(0, t) - R_i(0, t), \\ D_i(t) &= \inf\{\Delta t > 0 : R_i(0, t + \Delta t) \\ &\quad \geq A_i(0, t)\}. \end{aligned} \tag{1}$$

Herein we focus on *deterministic* QoS guarantees, which refer to the worst-case service that a session can receive from the scheduler: the worst-case service is expressed as the maximum packet delay or the minimum service throughput. Intuitively thinking, the worst-case service scenario for session i is determined by the amount of traffic that session i sends to the switch (this amount could be measured in, e.g., bits or ATM cells), and the amount of service that receives from the switch. In its turn, the amount of service is determined by the available bandwidth and the scheduling algorithm: for some scheduling algorithms, the amount of service provided to session i does not only depend on session i traffic, but it is also a function of the traffic generated by the other active sessions in the network. Hence, if bounds on both the input traffic and the provided service exist, it is possible to derive bounds on the maximum delay and minimum throughput [8]. Next, in order to lay out the framework of our scheduling algorithm, we consider bounds on the amount of traffic generated by sessions, and the bounds on the amount of service provided by the scheduler.

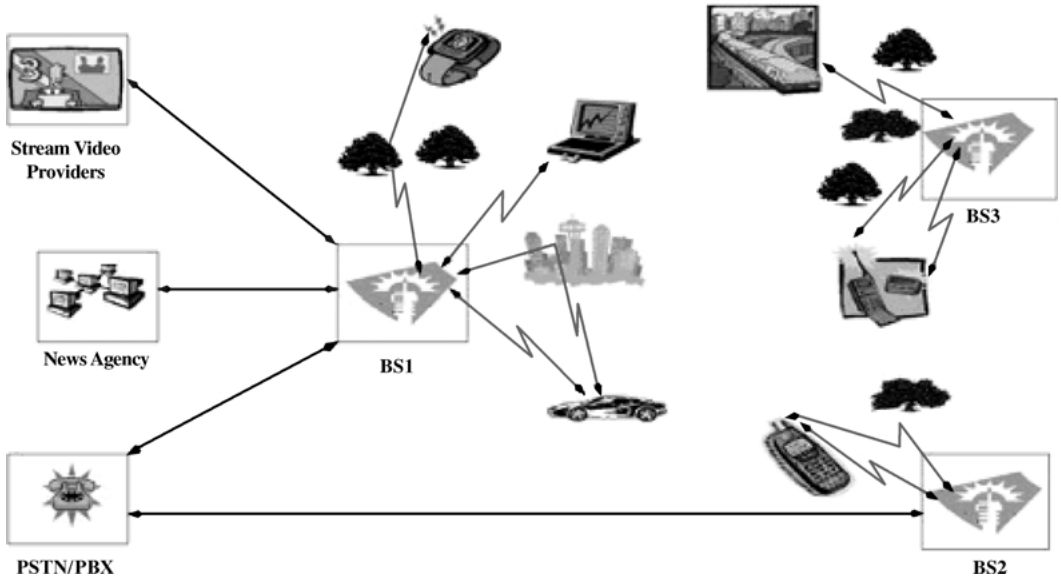

Figure 2. Basestation in a single cell.

2.1. Bound on Incoming Traffic

An upper bound on the amount of traffic generated by session i is given by the traffic envelope $A_i^e(t)$ defined as [8]:

$$A_i^e(t) := \sup_{0\le\tau}\{A_i(\tau, t+\tau)\}.$$

The traffic envelope provides an expression of the worst-case traffic that can be generated by a session (i.e., when the session is *greedy*) in a time interval of length t. In integrated services networks, knowledge of the traffic envelope of a traffic source is not only a prerequisite in determining QoS guarantees, but it can also be used in traffic policing and admission control. As it is naturally expected, $A_i^e(t)$ can have an arbitrary shape; for simplicity of implementation reasons, it is highly desirable that as few as possible scalar parameters describe $A_i^e(t)$. In the case of a *leaky-bucket* constrained session [8], two positive constants, σ_i, ρ_i, determine the affine function $A_i(t) = \sigma_i + \rho_i t$; (σ_i, ρ_i) are chosen such that $A_i^e(t) \le A_i(t)$, and $A_i(t)$ is as "close" to $A_i^e(t)$ as possible. In other words, the leaky bucket serves as an approximation of the traffic envelope of a specific session (note that the term "approximation" is rather loosely defined); the intuition behind the leaky bucket traffic descriptor is that ρ_i indicates approximately how much bandwidth should be allocated to session i, whereas σ_i indicates approximately the buffer space which should be reserved for the session so that no overflows occur. To improve the quality of the approximation (which increases the bandwidth utilization inside the network), a single leaky bucket can be generalized to a multiple leaky-bucket [9]:

$$A_i(t) := \min_{1\le k\le K_i}\{\sigma_{i,k} + \rho_{i,k} t\}.$$

Multiple leaky-buckets capture the notion of multiple transmission rates (over different periods of time), and provide a parsimonious way of describing real-world traffic sources (see, e.g., [9], and [10] for an algorithm to determine the parameters of the leaky buckets). From a practical point of view, leaky buckets are attractive as traffic policing mechanisms because they can implemented by just a few lines of `C` or `assembly` code, and have been incorporated in the ATM standard (in the GCRA algorithm, see, e.g., [11]). From a theoretical point of view, the operation of leaky buckets can be rigorously defined (see [12] and references therein): under the (min, +) algebra, a single leaky bucket can be realized by an IIR filter, whereas multiple leaky buckets can be realized by a filterbank of IIR filters.

2.2. Bounds on Provided Service by GPS

The scheduling algorithm, which is employed at the switch, determines the transmission order of the outgoing packets, and is primarily responsible for the bandwidth allocation to the sessions in the system (note

also that the scheduler can also determine how much queueing buffer space is allocated to incoming packets). It is common to assume that the scheduler operates at a fixed rate r: e.g., the scheduler can take r ATM cells/sec from the incoming queues, and transmit them on the outgoing link. The fixed-rate assumption is valid in most wireline network routers, but in wireless networks (where the physical layer achievable transmission rates fluctuate depending on the channel quality) the service rate remains fixed if the physical layer is not adaptive (for example, the physical layer does not employ any of the recently-proposed adaptive modulation or adaptive coding techniques [13]).

From a network-wide point of view, the scheduler should efficiently utilize the available resources. From a user perspective, the scheduler should guarantee to the sessions that: (i) network resources are allocated irrespective of the behavior of the other sessions (which refers to the *isolation* property of the scheduler), and (ii) whenever network resources become available (e.g., in underloaded scenarios), the extra resources are distributed to active sessions (the *fairness* property of the scheduler). Known for its perfect isolation and perfect fairness properties is the Generalized Processor Sharing (GPS) scheduling algorithm [2].

According to [2], a GPS server operates at a fixed rate r and is work-conserving, i.e., the server is not idle if there are backlogged packets to be transmitted. Each session i is characterized by a positive constant ϕ_i, and the amount of service $R_i(\tau, t)$ session i receives in the interval $(\tau, t]$ is proportional to ϕ_i, provided that the session is continuously backlogged. Formally, under GPS, if session i is continuously backlogged in $(\tau, t]$, then it holds:

$$\frac{R_i(\tau, t)}{R_j(\tau, t)} \geq \frac{\phi_i}{\phi_j},$$

for all sessions j that have also received some service in this time interval. It follows that in the worst case, the minimum guaranteed rate g_i given to session i is $g_i = r\phi_i / \sum_{j=0}^{N-1} \phi_j$, where N is the maximum number of sessions that could be active in the system. Therefore, a lower bound for the amount of service that session i is guaranteed is: $R_i^l(\tau, t) := (t - \tau) r\phi_i / \sum_{j=0}^{N-1} \phi_j$. If session i is (σ_i, ρ_i)-leaky bucket constrained, and the minimum guaranteed rate is such that $g_i \geq \rho_i$, then the maximum delay is $D_i^{\max} \leq \sigma_i / g_i$ (note that this bound could be loose [2]).

Effectively, GPS offers perfect isolation, because every session is guaranteed its portion of the bandwidth irrespective of the behavior of the other sessions. From this point of view, GPS is reminiscent of fixed-assignment TDMA or FDMA physical layer multiplexing techniques. What is radically different about GPS is its perfect fairness property: whenever a session i generates traffic at a rate less than g_i, then the "extra" bandwidth is allocated to other sessions proportionally to their respective weights. Let us clarify the operation of GPS in a wireless network using the following simple example: suppose that in a pico-cell three mobile users are assigned to the base-station. One (high-rate) user has a weight of $\phi_{hr} = 1$, and the two (low-rate) users have a weight of $\phi_{lr} = 0.5$. When all users are active, the high-rate user will take 50% of the bandwidth, and each of the low-rate users 25%. If one of the low-rate users becomes silent, then the extra 25% of the bandwidth will be allocated to the other users: the high-rate will have now 66%, and the low-rate 34%. Note that the extra bandwidth can be used in a multiple of ways: for example, to increase the information rate, or to decrease the transmitted power through the use of a more powerful channel code.

GPS belongs to the family of rate-based schedulers [14], which attempt to provide bandwidth guarantees to sessions (recall that bandwidth guarantees yield delay guarantees if the traffic envelope is known). When the sessions have a nominal, long-term average rate (the "sustainable cell rate" (SCR) in ATM terminology), then the allocation of ϕ's appears to be straightforward. The situation becomes more complicated if we consider that network traffic could be bursty or self-similar. Though there has been work on the weight assignment problem (see, e.g., [15]), it is still considered quite challenging (see, e.g., [16]). Apart from the weight assignment problem, another important issue is the implementation of GPS: real-world routers operate at the packet or cell level, whereas GPS assumes a fluid model of traffic. Hence, in practice GPS needs to be approximated by a Packet Fair Queueing (PFQ) algorithm [2]. Starting with Weighted Fair Queueing (WFQ) [17], there has been a lot of work on approximating GPS (see, e.g., [18–22]). We will revisit some of this work in Section 4.2.

2.3. *Bound on Service Provided by EDF and SC-based schedulers*

Despite its perfect isolation and fairness properties, GPS is not necessarily the panacea of the scheduling problem. Let us observe that there are applications

where we are interested more in delay QoS guarantees and less in throughput QoS guarantees: such an application is voice (IP telephony), where it is tolerable if some packets are dropped, but it is not tolerable if packets arrive too late. In such cases, it is meaningful to pursue the design of schedulers which base their operation on criteria other than rate.

The Earliest Deadline First (EDF) algorithm, a member of deadline-based schedulers, attempts to provide maximum delay guarantees to individual sessions. Under EDF, each session i is characterized by a delay bound $d_i > 0$. When a packet of session i arrives at the scheduler at time t, the deadline $(t + d_i)$ is assigned to the packet, and the switch transmits packets in increasing order of their deadlines (thus the name "earliest deadline first"). As long as the following "schedulability condition" holds [23, 24]:

$$\sum_{i=0}^{N-1} A_i^e(t - d_i) \leq tr, \tag{2}$$

and every source conforms to its traffic envelope, then no packet will miss its transmission deadline. It follows from (2), that given a set of traffic envelopes $\{A_0^e(t), \ldots, A_{N-1}^e(t)\}$, there are many vectors $\mathbf{d} = (d_0, \ldots, d_{N-1}) \in \mathbb{R}_+^N$ which satisfy (2). The set

$$\mathcal{R}_{\text{EDF}}\big(A_0^e(t), \ldots, A_{N-1}^e(t)\big) := \{\mathbf{d} \in \mathbb{R}_+^N : (2) \text{ is satisfied}\}$$

is called the *schedulability region*, which intuitively indicates the range of achievable QoS guarantees that can be provided to sessions. The importance of EDF stems (partially) from its optimality with respect to the schedulability region [23]: for an arbitrary scheduling algorithm with corresponding schedulability region $\mathcal{R}\big(A_0^e(t), \ldots, A_{N-1}^e(t)\big)$, it holds

$$\mathcal{R}\big(A_0^e(t), \ldots, A_{N-1}^e(t)\big) \subseteq \mathcal{R}_{\text{EDF}}\big(A_0^e(t), \ldots, A_{N-1}^e(t)\big).$$

From a practical point of view, if delay bounds are the only desirable QoS aspect, then EDF supports the maximum number of users than *any* other scheduling algorithm. As a result, EDF has been proposed as the basis of a network-wide deterministic QoS architecture (see e.g., [25]). Note also that recently, there has been work on statistical QoS guarantees using EDF (see e.g., [26] and references therein).

Going a step beyond rate-based (such as GPS) or deadline-based (such as EDF) schedulers, a SC-based scheduler attempts in $[0, t]$ to provide service to session i greater or equal to $S_i(t)$ [5, 6]. Formally, let the service curve $S(\cdot)$ be a nonnegative nondecreasing real function with $S_i(0) = 0$. Then a session i which starts transmitting at time t_o is guaranteed the service curve $S_i(\cdot)$, if for any time $t \geq t_o$, there exists $t_o \leq s \leq t$ such that

$$R_i(t_o, t) - A_i(t_o, s) \geq S_i(t - s). \tag{3}$$

It is straightforward to check whether the switch is capable of satisfying all SCs by performing the following test [5]:

$$\sum_{i=0}^{N-1} S_i(t) \leq tr, \quad \forall t \geq 0. \tag{4}$$

Equation (3) indicates that a SC-based scheduler guarantees a relatively more "relaxed" type of service than GPS or EDF: GPS attempts at every moment to achieve proportional bandwidth distribution, and EDF strives to transmit every packet before its deadline elapses. On the other hand, a SC-based scheduler only guarantees that eventually every session will receive its guaranteed amount of service. For example, if $S_i(t) = g_i\, t$, a SC-based scheduler imposes a much weaker condition that of GPS. Implicitly, this condition leads to short-term unfairness and the punishment phenomenon as we explain later on. On the other hand, a SC-based scheduler is very versatile with respect to the QoS guarantees that it can provide to sessions: because a SC can have an arbitrary shape (as long as it is a non-decreasing real function of t and (4) holds), the SC can be used to provide delay or bandwidth guarantees [5].

From a practical point of view, SC-based schedulers are attractive in wireless/wireline networks, because there could be low-rate users, high-rate users, and applications which could demand different service rates during various periods of time (for example, a video session). In such cases, allocating a fixed portion of the bandwidth to a session could result in considerable waste of network resources. A SC-based scheduler could lead to more efficient bandwidth utilization, because it allows time-varying service guarantees, which have the potential to meet the needs of multirate users and applications. From a theoretical point of view, SCs possess two important properties: (i) rate-based schedulers can be cast into the SC framework [10], and (ii) the Earliest Deadline First (EDF) algorithm can be modeled using service curves (recall that the importance of property (ii) stems from the proven optimality of EDF

in admitting the maximum number of sessions given delay bounds and traffic characterization [23]).

To implement SCs in packet switched networks, [5] proposes the SC-based Earliest Deadline first policy (SCED), where every packet upon its arrival is assigned a deadline; the deadline is basically a function of $S_i(t)$ and the number of packets session i has transmitted up to time t. The packets are transmitted in increasing order of their deadlines. Apart from "resetting" [5], the assignment of deadlines to packets of a particular session in SCED does not take into consideration the behavior of the other sessions. Thus, the punishment feature of SCED, which is our main motivation for the development of the Deterministic Time-Varying PFQ (DTV-PFQ).

3. Punishment in SCED

As acknowledged in [5] and mentioned in [6], SCED exhibits the punishment property, which does not appear in GPS. Let us illustrate this with the following example (similar to an example in [2]): suppose that we are interested in providing minimum rate guarantees to a system with two sessions "1" and "2". Both of them are to receive 50% of the service rate. Using SCs, we could have $S_1(t) = S_2(t) = rt/2$, whereas under GPS we would have $\phi_1 = \phi_2 = 0.5$. We make the assumption that the scheduler operates in discrete time slots, and we let session "1" start transmitting at slot 0, whereas session "2" starts at slot 10. In the interval [0, 9], session "1" receives 100% of the available bandwidth: normally, source "1" starts transmitting packets at a rate no greater than $r/2$, because this is the rate which is guaranteed to the source. However, based on feedback information from the receiver (e.g., "packets arrived sooner than expected"), session "1" could decide to increase its rate. Figures 3 and 4 show the bandwidth which is allocated to sessions "1" and "2" in the interval [0, 30] under SCED and WFQ. To study how bandwidth is allocated, we make both sessions continuously backlogged by having them transmit at rate $1.5r$. It is clearly illustrated that under WFQ, session "1" is allocated 100% of the bandwidth in [0, 9] and 50% of the bandwidth in [10, 30]. On the other hand, under SCED, session "1" receives 100% of the bandwidth, but in [10, 14] session "1" does not receive any service at all. Eventually, in [0, 30], session "1" receives at least 50% of the bandwidth, as it was advertised. Nevertheless, session "1" is punished for being greedy in $[0, t_1]$.

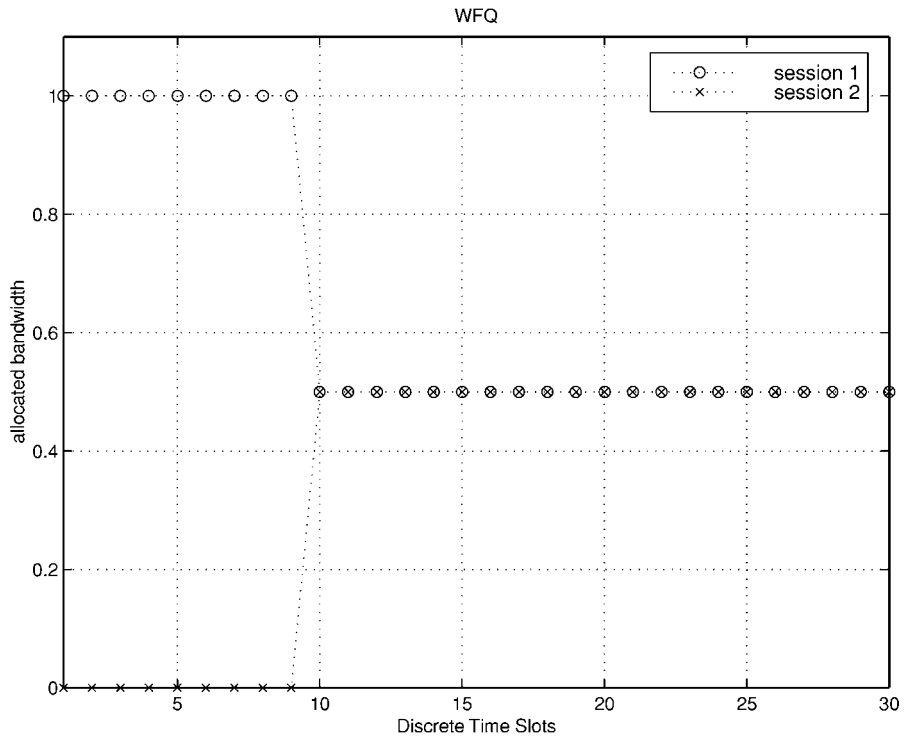

Figure 3. Bandwidth allocation under GPS.

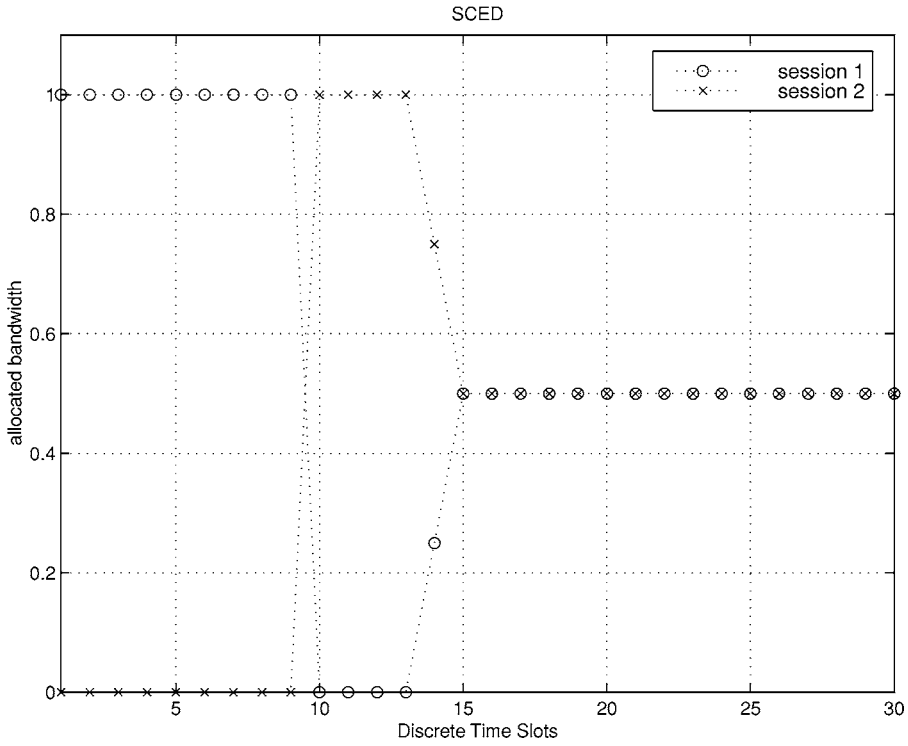

Figure 4. Bandwidth allocation under SCED.

Given the multirate capabilities of SC-based schedulers, and the potential of more efficient bandwidth utilization, the punishment phenomenon is our motivation for the design of DTV-PFQ. In wireless networks, where the bandwidth is so scarce and expensive, the punishment phenomenon is very undesirable: perhaps it makes little sense not to let a user transmit in future transmission rounds just because in some of the previous rounds, some of the other users in the cell were not active. Moreover, the wireless channel is time-varying; occasionally users experience "bad" or "good" channels. It is counter-intuitive not to let a user transmit packets, especially when the channel is "good". On the other hand, in wireline networks, the punishment phenomenon does not encourage sessions to take advantage of extra bandwidth that may be available in the system. Before we present our designs on DTV-PFQ, we remark that considering the same scheduling algorithm for both wireless and wireline environments facilitates seamless integration between wireless-wireline network components, and allows us to capitalize on the extensive published work on scheduling in wireline networks.

4. Deterministic Time-Varying PFQ

To overcome the performance limitations caused by the delay-bandwidth coupling of GPS, herein we propose a time-varying assignment of weights, which provides us with more degrees of freedom than the non rate-proportional weighting of [15]. In our scheme, the weight ϕ_i which is assigned to a session i is a function of time $\phi_i(t)$. In particular, we focus in the case where the variations in $\phi_i(t)$ are carried out in a deterministic fashion and unlike [27, 28], we develop a framework with minimum overhead. In this paper we study the single-node case, leaving the study of the multiple-node node case for future work.

In this section, first we discuss why in theory a Time-Varying GPS is capable of implementing SCs of arbitrary shapes, but it is difficult to realize them in practice. Then, we focus on piecewise linear SCs having in mind that in integrated services networks, piecewise linear SCs can be used to provide multirate services [29], and delay guarantees to sessions constrained by multiple leaky buckets (recall from Section 2.1 that multiple leaky-buckets have been proposed to model

real-life applications and shown to result in improved network utilization [9]). We design a practically realizable weight assignment algorithm which guarantees piecewise linear SCs while obviating the punishment phenomenon. Moreover, we discuss how prescribed delay bounds can be met by DTV-PFQ (as long as the delay bounds are EDF-feasible), and show that our newly introduced scheme is optimal in the schedulability-region sense (as long as the traffic envelopes are piecewise concave linear functions).

4.1. Time-Varying GPS, in Theory

Let us recall that every scheduling algorithm has to face two issues: (i) allocation of reserved bandwidth (which leads to isolation among sessions, and provision of worst-case guarantees), and (ii) distribution of extra (or available) bandwidth to active sessions (which defines the fairness properties of the scheduler). The SC framework generalizes GPS with respect to the first issue, as it allows a session to demand different service rates over different periods in time. However, a SC-based scheduler does not provide any analytical guarantees-results about how the extra bandwidth is allocated to sessions. On the other hand, GPS strives to achieve perfect fairness by providing the same normalized service to all backlogged sessions at any time interval [30]: if sessions i, j are continuously backlogged in $(\tau, t]$, then $R_i(\tau, t)/R_j(\tau, t) = \phi_i/\phi_j$.

In theory, a time-varying GPS system is capable of accommodating arbitrarily shaped SCs $S_i(t)$ provided that (4) is satisfied. By setting $\phi_i(t) = \partial S(t)/\partial t$, we obtain $\sum_{i=0}^{N-1} \phi_i(t) \leq 1$ at any time t, and as a result, $\int_0^t \phi_i(\tau)\, d\tau \geq S_i(t)$, $\forall t \geq 0$, provided that $S_i(t)$ is differentiable (we will address the case when the derivative does not exist later on). Hence, the deterministic assignment $\phi_i(t) = \partial S(t)/\partial t$ allows GPS to provide services similar to a SC-based scheduler. Intuitively thinking, the "equivalence" between a time-varying GPS system and a SC-based system should not come as a surprise. However, what perhaps comes as a surprise is the difficulty of implementing an arbitrary weight assignment in a packet-by-packet practically realizable system.

In a real system, the GPS scheduler assigns deadlines to incoming packets; these deadlines, as we will explain in Section 4.2, are given as a function of a quantity termed the "virtual time" of the system and the weight of the session. Let us consider a packet of session i that arrives at time t_1. This packet will be transmitted by the system at a later time $t_2 \geq t_1$. At time t_2, the weight of the session is $\phi_i(t_2) = \frac{\partial S_i(t)}{\partial t}|_{t=t_2}$. However, the time instant t_2 is not known upon the packet arrival at t_1. The time t_2 does not only depend on the backlog of the session i at time t_1, but also on the backlog and future packet arrivals of the other sessions. Therefore, unless the SC has a constant slope (which corresponds to fixed weight assignments), it is quite challenging to assign deadlines to packets upon their arrival. A possible solution is the computationally expensive algorithm of [27], which uses a vector $\mathbf{V} \in \mathbb{R}^3$ as virtual time in the system. However, in high-speed (gigabit) networks, the implementation overhead of the scheduler should be kept as small as possible.

Though we have not solved the problem of GPS supporting arbitrarily shaped SCs, we will show in Section 4.3 that fortunately in the practically appealing case of piecewise linear SCs, it is possible to implement DTV-PFQ with a computationally efficient deadline-weight assignment procedure. Before we present our approach in Section 4.3, let us revisit some known results on the packet-by-packet implementation of GPS and SC-based schedulers (as our DTV-PFQ algorithm capitalizes on these results).

4.2. PFQ and SCED, in Practice

Perhaps one of the most renowned shortcomings of GPS is that it relies on a fluid model of traffic-service: the traffic sent by the users is considered to be infinitely divisible, and the router is supposed to be able to service all active sessions simultaneously. However, non-cut-through switches operate at the packet or cell level (i.e., they transmit one packet of a single session at a time), a fact which implies that GPS needs to be approximated by a Packet Fair Queueing algorithm [2]. A PFQ algorithm attempts to allocate bandwidth proportionally to session weights, i.e., it attempts to minimize the difference

$$\mathcal{F}_{ij}(\tau, t) = \left| \frac{R_i(\tau, t)}{\phi_i} - \frac{R_j(\tau, t)}{\phi_j} \right|,$$

which serves as the "fairness index" of the algorithm (note that in GPS $\mathcal{F}_{ij}(\tau, t) = 0$ for sessions which are continuously backlogged—see, e.g., [31]). Many PFQ algorithms have been proposed in the literature, and their merits can be partially judged by the maximum value of $\mathcal{F}_{ij}(\tau, t)$, the latency (which indicates how

long it takes for a new session to start receiving service), and the implementation complexity (see, e.g., [30, 31], and references therein).

All PFQ algorithms need to keep track of the amount of service that has been provided to the active sessions in the system. Central to almost all PFQ algorithms is the notion of "virtual time" or "system potential" (see [31] for a unifying framework) which is used for the assignment of deadlines: when the n-th packet of session i arrives to the switch at time $t_i^{(n)}$, the packet is time-stamped with a deadline $F_i^{(n)}$, which is a function of the virtual time and ϕ_i. In [2], the virtual time $V(t)$ is a scalar quantity, which is incremented every time a packet arrives or departs from the scheduler ($L_i^{(n)}$ is the packet length):

$$V(t_k) = V(t_{k-1}) + \frac{t_k - t_{k-1}}{\sum_{j\in B(t)} \phi_j}, \quad V(0) := 0, \tag{5}$$

$$F_i^{(n)} = \max\left\{F_i^{(n-1)}, V\left(t_i^{(n)}\right)\right\} + \frac{L_i^{(n)}}{\phi_i r}, \quad F_i^{(0)} := 0. \tag{6}$$

In (5) the sequence $\{t_k\}$ represents the times that a packet arrives to or departs from the switch, and $B(t)$ denotes the set of backlogged sessions at time t. The packets are transmitted by the switch in increasing order of their time-stamps, and the virtual time measures the progress of the work in the system. Thus, the virtual time is primarily responsible for the absence of the *punishment* phenomenon in PFQ algorithms [2], as the amount of service assigned to a session is represented by a "session potential function" [30] (in [2], the state of session i is represented by the sequence $\{F_i^{(n)}\}$).

Equations (5) and (6) indicate that the implementation of PFQ relies on the knowledge of the ϕ_i's. Our basic intuition is to utilize the basic results on GPS, and extend the service guarantees that can be provided to sessions by using time-varying ϕ_i's. In this case, given a way to determine $\phi_i(t)$ for every session i, fairness-delay-rate guarantee results for PFQ can be translated to DTV-PFQ. In Section IV-C we will describe how $\phi_i(t)$ can be defined deterministically, and how DTV-PFQ can be implemented in the case of piecewise linear SCs. Before we describe our approach, let us define formally the SCs which can be provided to individual sessions.

References [5, 29] have provided computationally efficient scheduling algorithms for piecewise linear SCs $S_i(t)$ defined as:

$$S_i(t) = \max\left\{0, \min_{k=1,\ldots,K_i}\{a_{i,k} + b_{i,k}t\}\right\},$$

where $a_{i,k}$, $b_{i,k}$ $(1 \le i \le N,\ 1 \le k \le K_i)$ are real constants satisfying:

$$b_1 > b_2 > \cdots > b_{K_i} > 0, \quad 1 \le i \le N$$
$$\text{and} \quad 1 \le \frac{a_2 - a_1}{b_1 - b_2} < \cdots < \frac{a_K - a_{K-1}}{b_{K_i} - b_{K_i-1}}.$$

Details on the selection of leaky bucket parameters can be found in, e.g., [9, 10]. Herein, we allow the *piecewise linear* SC to be zero in the interval $[0, T_i)$ and have an initial "burst":

$$S_i(t) = \begin{cases} \max\left\{0, \min\limits_{k=1,\ldots,K_i}\left\{a_{i,k} + b_{i,k}t\right\}\right\} : & t \ge T_i \\ 0 : & 0 \le t < T_i, \end{cases} \tag{7}$$

We note that the introduction of the constant T_i (which is zero in [5, 29]) allows our DTV-PFQ scheme to model an EDF scheduler for multiple-leaky buckets. Indeed, if session i has traffic envelope:

$$A_i^e(t) = \min_{1\le k\le K_i}\{\sigma_{i,k} + \rho_{i,k}t\},$$

then the allocation of the SC $S_i(t) \leftarrow A_i^e(t - d_i)$ guarantees the delay bound d_i as long as (4) holds (Fig. 5). It readily follows that this allocation of the SC amounts to setting: $T_i \leftarrow d_i$, $b_{i,k} \leftarrow \rho_{i,k}$, $a_{i,k} \leftarrow \sigma_{i,k} - \rho_{i,k}d_i$. Therefore, modulo approximation errors induced by any virtual-time based implementation [2], our DTV-PFQ scheme is capable of achieving approximately the schedulability region of an EDF scheduler (for multiple leaky-bucket constrained sessions).

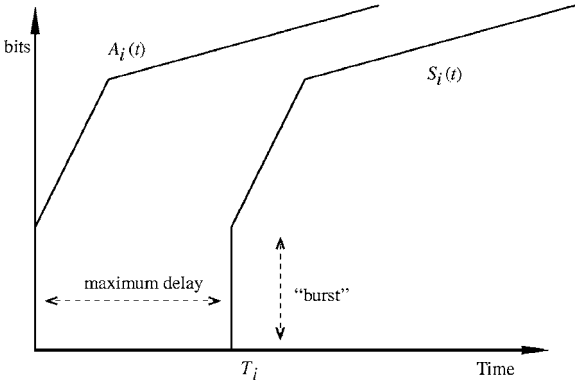

Figure 5. Envelope and service curve.

4.3. DTV-PFQ, in Practice

Out DTV-PFQ algorithm implements piecewise linear service curves as given by (7). In order to obviate the punishment phenomenon, we rely on the virtual time implementation of GPS and, in order to provide multiple rates to the same session, we rely on the algorithms of [5, 29]. However, the combination of GPS and [5, 29] is not straightforward as we encounter 3 issues:

(1) the weight of session i assumes K_i discrete values $\phi_{i,k} := b_{i,k}/r$, $1 \le k \le K_i$, which correspond to the K_i different service rates that are to be given to the session.
(2) Eq. (6) should be modified so that the delay factor T_i is taken into account in the assignment of the deadline of the first packet of a session.
(3) the term $a_{i,1}$ (the "burst") in the service curve at time T_i corresponds to an infinite slope ($\phi_i = \infty$), and it should be handled in an appropriate way.

Issue (1) is not very difficult to handle: [5, 29] provide mechanisms to determine the service curve slope $b_{i,k}$ which corresponds to each incoming packet of session i. Therefore, when a packet of session i arrives, we can use the method of [5, 29] to determine what is the slope $b_{i,k}$ of $S_i(t)$ which corresponds to that packet. Then, we set ϕ_i equal to $b_{i,k}/r$.

Issue (2) is taken care of by adding to the deadline of the first packet of the session the term VirtualOffset_i, which is defined as:

$$\text{VirtualOffset}_i := \int_0^{T_i} \frac{I_i(t)}{\sum_{j=1, j\neq i}^{N} \tilde{\phi}_j(t)}\, dt, \quad (8)$$

where $I_i(t)$ is an indicator function defined by:

$$I_i(t) := \begin{cases} 1: & \forall j\ S_j(t) \text{ is differentiable at } t \\ 0: & \text{otherwise,} \end{cases} \quad (9)$$

and $\tilde{\phi}_j(t)$ is the slope of $S_j(t)$:

$$\tilde{\phi}_j(t) := \begin{cases} \dfrac{\partial S_j(t)}{\partial t}\ \dfrac{1}{r}: & S_j(t) \text{ is differentiable} \\ 0: & \text{otherwise.} \end{cases} \quad (10)$$

Note that we adopt the convention that if no $S_j(t)$ is differentiable at a specific t, then the overall value of the integrated quantity is 0 (at that specific t).

As (8) indicates, VirtualOffset_i accounts for the work-load that the scheduler should give to other sessions: in the worst-case all the other sessions are continuously active, and by integrating (5) we obtain a (conservative) estimate of the value of the virtual time at which session i should start receiving service from the scheduler. We remark that the term VirtualOffset_i enables our DTV-PFQ scheduler to emulate the EDF-scheduler; the intuition is that packets of a newly arrived session may be forced to wait a little bit longer in the scheduler. Interestingly enough, [30] indicates that when the potential of a new session is set higher than that of the sessions currently serviced, then the new session may have to wait before it can be serviced. In our scheme, we set the potential (i.e., the virtual finishing time) of a new session higher than the current virtual time on purpose.

Finally, issue (3) is handled by setting "informally" $\phi_i = \infty$ for the packets of the session which correspond to the sudden "burst". As (5) and (6) suggest, if $\phi_i = \infty$, the virtual time and the finishing time of the session are not updated. Note that the number of packets which belong to the initial burst is bounded. Therefore, if the admission test (4) is satisfied, then isolation among sessions is preserved.

Our DTV-PFQ system can be implemented using the algorithm in Fig. 6. Our algorithm extends the algorithms of [2, 5] by addressing (1), (2), and (3), while maintaining the same complexity as the algorithm of [5]. We remark that our scheme provides an exact algorithm for the weight assignment, unlike [15] which provides only a heuristic way of ϕ-assignment. Also, different from [16] which looks at the feasible region of weights for only two sessions, our scheme supports an arbitrary number of sessions. An approach different that ours is taken in [28] which proposes the adaptive modification of the session weights: the dynamic programming algorithm of [28] attempts to minimize the queuing delays, but it sacrifices closed-form individual session guarantees, and requires traffic policing (because otherwise, greedy sessions would take over all the bandwidth).

4.4. DTV-PFQ, Approximation Errors and Fairness Guarantees

In DTV-PFQ approximation errors are caused by two factors: (i) the inability to assign the proper weight for the (rare) case of packets which cross the boundary of two leaky buckets, and (ii) the $L_{\max}/r$ approximation

Initialize: $\delta_{i,k} \leftarrow 0, \delta^o_{i,k} \leftarrow -a_{i,k}/b_{i,k}$,
$1 \le i \le N, 1 \le k \le K_i$.

In each slot u during which packets have arrived:
$\phi_i \leftarrow 0, i = 1 \ldots N$
for $i = 1 \ldots N$ do
  if $A_i^{\text{in}}[u] \neq 0$ /* packets from i arrived in slot u */
    $\delta_{i,k} \leftarrow \max\{\delta_{i,k}, u - 1 + \delta^o_{i,k}\}$
    for $l = 1 \ldots A_i^{\text{in}}[u]$ do
      $\delta_{i,k} \leftarrow \delta_{i,k} + 1/b_{i,k}, 1 \le k \le K_i$
      $k = \arg \max_{1 \le k \le K_i} \delta_{i,k}$
      if $u \le \delta_{i,k}$
        then $\phi_i \leftarrow 0$
        else $\phi_i \leftarrow b_{i,k}/r$
      endif
      if $l = 1$ and session was not backlogged
        then $F_i = V + \text{VirtualOffset}_i$
        else if $\phi_i \neq 0$
          then $F_i = \max\{F_i, V\} + 1/\phi_i$
        endif
      endif
      assign the deadline F_i to the packet
    endfor
  endif
endfor
sum $= \sum_{i=1}^{N} \phi_i$
if sum $\neq 0$
  then $V \leftarrow V + \frac{1}{\text{sum}}$
endif

Figure 6. Algorithm for deadline assignment.

error which is associated with PFQ algorithms [2] ($L_{\max}$ is the maximum packet length). As a result, the actual SC which can be guaranteed in the packetized system to a session is equal to:

$$\underbrace{S_i(t)}_{\text{fluid model}} - \underbrace{L_{\max}}_{\text{GPS aproximation}} - \underbrace{\sum_{k=1}^{K_i}(-b_k + b_{k-1})\frac{L_{\max}}{r}}_{\text{leaky bucket boundary}}, \quad \text{with } b_0 := 0\,.$$

With respect to the fairness guarantees of DTV-PFQ, in the fluid model, the extra bandwidth in the system is distributed according to the $\phi_i(t)$'s (as a result, perfect instantaneous fairness is provided to sessions). In the packetized version, the service that a session receives is subject to the approximations induced by the virtual time implementation. Having in mind that in the packetized system the service of one session can lag at most by $L_{\max}$ or be ahead by $(N-1)L_{\max}$ with respect to the fluid system, then over an interval $(\tau, t]$ where both sessions i, j are continuously backlogged, we could have:

$$\frac{R_i(\tau,t)}{R_j(\tau,t)} \ge \frac{-\frac{L_{\max}}{r} + \frac{1}{t-\tau}\int_\tau^t \phi_i(t)\,dt}{(N-1)\frac{L_{\max}}{r} + \frac{1}{t-\tau}\int_\tau^t \phi_i(t)\,dt}$$

To illustrate the operation of our algorithm, we assume a system which supports two sessions ("1" and "2"). We simulate the system for 20 slots, and we make both sessions transmit at rate $2r$ (to keep them both continuously backlogged, which allows us to study the bandwidth allocation). Session "1" starts transmitting at slot 0, whereas session "2" starts transmitting at slot 4. Figures 7 and 8 illustrate the bandwidth allocation under SCED and under DTV-PFQ. We assume that $b_1 = 25\%$, $b_2 = 75\%$, $T_1 = 10$, $T_2 = 5$. As Fig. 7 depicts, session "1" does not receive any service at all in [4, 11], being penalized for the extra bandwidth it received in [0, 9]. On the other hand, DTV-PFQ does not penalize session "1" (Fig. 8), because session "1" still receives service in [4, 11]. Under both schedulers, we observe that in the long run sessions "1" and "2" receive respectively 25% and 75% of the bandwidth; but it is in the transient that DTV-PFQ performs better than SCED.

5. Scheduler Implementation

In Section 4.2 we briefly mentioned the problems associated with implementing the fluid traffic-service model of GPS in a packet-by-packet system. However, in both wireline and wireless networks, when it comes to the practical implementation of schedulers, there is more than meets the eye.

In wireline networks, the switch is either output-buffered or input-buffered depending on where the incoming packets are queued; if the switching fabric of the router is N times faster than the speed of the N input links, then the server is fast enough to route packets to the same outgoing link. Though there are some existing 5 Gb/s routers with switching fabric fast enough to keep pace with the input links (see, e.g., [32]), this might not be true for all existing routers: given the recent advances in gigabit optical networking, the switching fabric may not be fast enough, and input-buffering may be employed. Unfortunately, in such a case, tracking a fluid scheduling policy (such as GPS) with a non-anticipative packet scheduling policy is still an open problem (see, e.g., [33] and references therein for a link to the multi-periodic TDMA satellite scheduling problem). The importance of input-output buffering notwithstanding, the implementation cost of PFQ

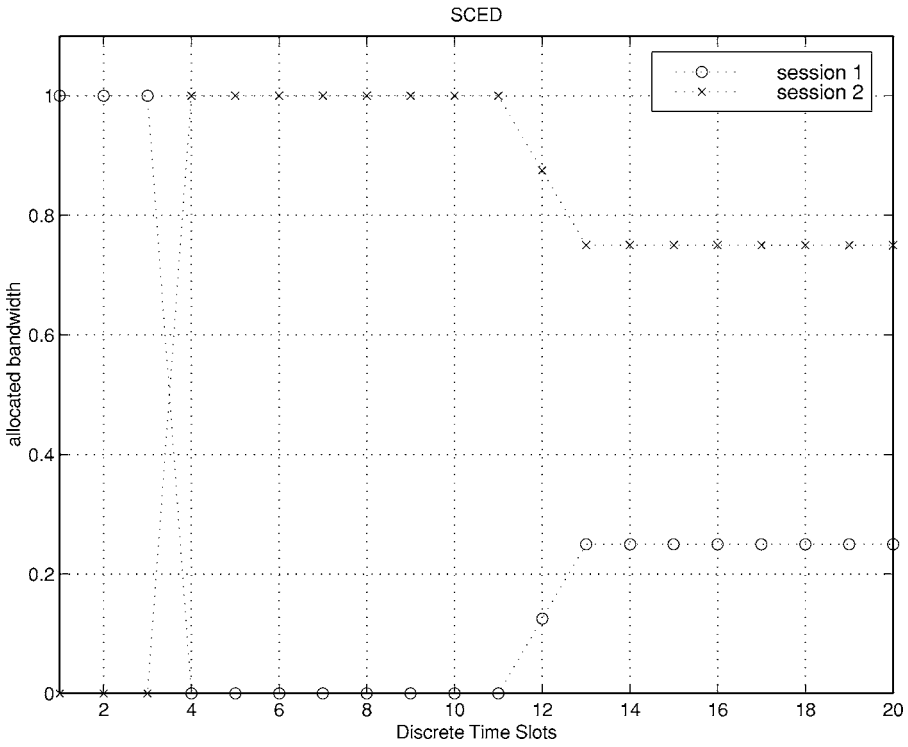

Figure 7. Bandwidth allocation under SCED.

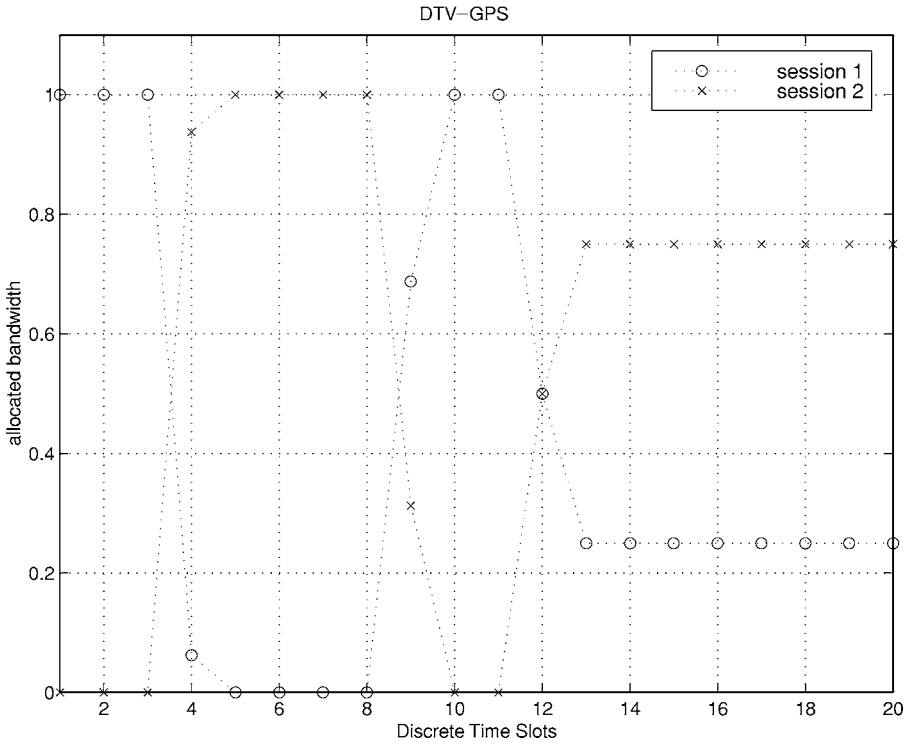

Figure 8. Bandwidth allocation under DTV-PFQ.

algorithms is a function of the following three factors (see, e.g., [34] and references therein): (i) the cost associated with keeping track of the system-potential function, (ii) the computational complexity of timestamp sorting, (iii) the storage and scalability issues related to recording the state of every session (see also [35] and references therein).

On the other hand, in wireless networks, implementing PFQ poses a multitude of challenges. First of all, the wireless channel exhibits time-varying capacity and induces location-dependent errors which raise issues with respect to short-versus long-term fairness guarantees (see, e.g., [36–39]. Second, unlike wireline networks, in the uplink scenario the scheduler (which is located at the basestation) does not know a priori the queue status of the mobile users; it is up to the MAC to communicate the bandwidth requests of the mobile users to the scheduler (see, e.g., [3] and references therein): essentially, the network and the physical layer can be tied together using a two-phase demand assignment MAC protocol. During the first phase, each user m notifies the base station about its intention to transmit; the base station calculates the ϕ's, and notifies each user about the corresponding bandwidth assignments. During the second phase, users transmit (at possibly different rates) multimedia information. Note that (i) the duration of the reservation phase can be reduced if users piggy-back their queue-lengths in prespecified intervals), and that (ii) the overall scheme becomes much simpler in the downlink case, as the basestation is aware of the queue lengths of all data streams. Third, though in wireline networks the reliability of the physical medium allows the independent design of physical/data-link/network layers, in wireless networks a lot is to be gained by a joint design approach across network layers.

A very interesting ramification of the joint design approach across layers is that in a multicode CDMA wireless network, the implementation of PFQ does not necessarily have to be based on virtual time. To make this notion concrete, let us focus on a multicode CDMA transmission/reception scheme with C available codes[1] (these codes could be, e.g., Pseudo-Noise or Walsh-Hadamard), which can be allocated to mobile users. Each user m is allocated c_m codes, and splits the information stream into c_m substreams which are transmitted simultaneously using each of the c_m codes: it readily follows that if user m has Q_m data symbols to transmit, then Q_m/c_m yields a measure of the time it takes to transmit them. Assuming that all C codes can be successfully used (an issue which hinges upon channel conditions, power control, and the detection mechanism), c_m/C essentially denotes the bandwidth which is allocated to user m, and GPS is implemented by setting

$$c_m = \left\lfloor \frac{\phi_m}{\sum_{\mu \in \mathcal{A}} \phi_\mu} C \right\rfloor, \tag{11}$$

where $\mathcal{A}$ is the set of active users (note that with C sufficiently large and frequent code re-assignments, the approximation error in implementing GPS using (11) can be made very small). As an illustrative example, we simulate a pico-cell where 3 mobile users communicate with the base-station. We assume $C = 32$, and that the traffic generated by each of the mobile users is Poisson with corresponding normalized rates $\lambda_1 = 1/2$, $\lambda_2 = 3/8$, $\lambda_3 = 1/8$. The weight assignment is $\phi_1 = 0.5$, $\phi_2 = 0.375$, $\phi_3 = 0.125$ (under which, if all three users have data to transmit, users 1, 2, 3 are assigned 16, 12, and 4 codes respectively), and we model the user 3 as an on/off source in order to study the bandwidth re-assignments. We simulate the system for 100 transmission rounds, and Fig. 9 depicts the number of queued packets and the number of allocated codes per user for a range of the transmission rounds. we can clearly see that users 1, 2 are allocated more CDMA codes whenever user 3 is silent.

Finally, let us comment on the need for exact implementation of the transmission schedule: it is well known that accurate approximation of GPS leads to smaller latency, decreases the burstiness of the outgoing traffic, and lowers the buffer requirements inside the network. To get a feel of how important scheduling becomes for real-time applications such as IP telephony, [40] reports that a maximum delay of 150 ms is tolerable for voice communication (using a PC): the 150 ms-limit presents a *delay budget* which is to be distributed over the propagation delay (measured at 95 ms for the longest path in the Continental United States, from Seattle, WA to Orlando, FL), and the queuing/processing delay. In the worst case, from the available 55 ms, 25 ms are to be allocated to the speech encoder, and the speech enhancement and silence suppression operations. Allowing for variable delay factors, only 10 ms are left for queueing delay in the backbone network. Hence, the scheduler does not only need to be fair, it also needs to be fast!

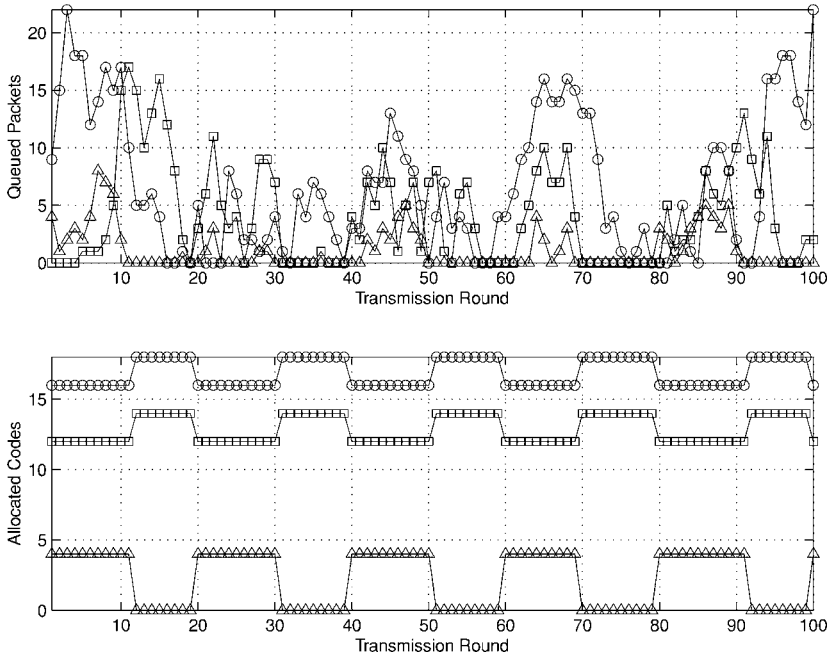

Figure 9. Code allocation and queue lengths in multirate CDMA.

6. Conclusions

In this paper we have discussed issues related to network scheduling in both wireline and wireless environments, and we have presented a deterministic time-varying weight assignment procedure for PFQ-based switching systems. By supporting piecewise linear SCs, our scheme dispenses with the *delay-bandwidth coupling* and targets integrated services networks. Unlike existing SC based algorithms, our time-varying PFQ scheme does not exhibit the *punishment* phenomenon and allows sessions to exploit the extra bandwidth in under-loaded networks. Future research avenues include the study of stochastic time-varying weight assignment procedures which take into account the probabilistic description of incoming traffic (for wired networks), as significant savings of statistical over deterministic services (inherently conservative in admitting sessions) have been reported in, e.g., [26]. On the other hand, an adaptive SC could model the time-varying channel conditions in wireless networks, and yield improved bandwidth utilization.

Note

1. Note that the capacity C is "soft" as it depends on channel conditions, power control, etc.

References

1. J. Chuang and N. Sollenberger, "Beyond 3G: Wideband Wireless Data Access Based on OFDM and Dynamic Packet Assignment," *IEEE Communications Magazine*, vol. 7, no. 38, 2000, pp. 78–87.
2. 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 Transactions on Networking*, vol. 1, no. 3, 1993, pp. 344–357.
3. A. Stamoulis and G.B. Giannakis, "Packet Fair Queueing Scheduling Based on Multirate Multipath-Transparent CDMA for Wireless Networks," in *Proc. of INFOCOM'2000*, Tel Aviv, Israel, March 2000, pp. 1067–1076.
4. Z.-L. Zhang, D. Towsley, and J. Kurose, "Statistical Analysis of Generalized Processor Sharing Scheduling Discipline," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 6, 1995, pp. 1071–1080.
5. H. Sariowan, R.L. Cruz, and G.G. Polyzos, "SCED: A Generalized Scheduling Policy for Guaranteeing Quality-of-Service," *IEEE Transactions on Networking*, vol. 7, no. 5, 1999, pp. 669–684.
6. I. Stoica, H. Zhang, and T.S.E. Ng, "A Hiearchical Fair Service Curve Algorithm for Link-Sharing, Real-Time and Priority Services," in *Proc. ACM Sigcomm'97*, 1997, pp. 249–262.
7. W. Willinger, M.S. Taqqu, R. Sherman, and D.V. Wilson, "Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN traffic at the Source Level," *IEEE/ACM Transactions on Networking*, vol. 5, no. 1, 1997, pp. 71–86.
8. R.L. Cruz, "A Calculus for Network Delay, Part I: Network Elements in Isolation," *IEEE Transactions on Information Theory*, vol. 37, no. 1, 1991, pp. 114–131.

9. D.E. Wrege, E.W. Knightly, H. Zhang, and J. Liebeherr, "Deterministic Delay Bounds for vbr Video in Packet-Switching Networks: Fundamental Limits and Practical Tradeoffs," *IEEE Transactions on Networking*, vol. 4, 1996, pp. 352–362.
10. J.Y. Le Boudec, "Application of Network Calculus to Guaranteed Service Networks," *IEEE Trans. on Information Theory*, vol. 44, no. 3, 1998, pp. 1087–1096.
11. `http://www.atmforum.com/atmforum/market_awareness/whitepapers/6.html`
12. C.-S. Chang, "On Deterministic Traffic Regulation and Service Guarantees: A Systematic Approach by Filtering," *IEEE Trans. on Information Theory*, vol. 44, no. 3, 1998, pp. 1097–1110.
13. A. Goldsmith, "Adaptive Modulation and Coding for Fading Channels," in *Proceedings of the 1999 IEEE Information Theory and Communications Workshop*, 1999, pp. 24–26.
14. H. Zhang, "Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks," *Proceedings of the IEEE*, vol. 83, no. 10, 1995, pp. 1374–1396.
15. R. Szabo, P. Barta, J. Biro, F. Nemeth, and C.-G. Perntz, "Non Rate-Proportional Weighting of Generalized Processor Sharing Schedulers," in *Proc. of GLOBECOM*, Rio de Janeiro, Brazil, Dec. 1999, pp. 1334–1339.
16. A. Elwalid and D. Mitra, "Design of Generalized Processor Sharing Schedulers Which Statistically Multiplex Heterogeneous QoS Classes," in *Proc. of INFOCOM'99*, Piscataway, NJ, USA, 1999, pp. 1220–1230.
17. A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," in *Proc. ACM Sigcomm'89*, 1989, pp. 1–12.
18. S.J. Golestani, "A Self-Clocked Fair Queueing Scheme for Broadband Applications," in *Proc. IEEE Infocom '94*, 1994, pp. 636–646.
19. P. Goyal, H.M. Vin, and H. Cheng, "Start-Time Fair Queueing: A Scheduling Algorithm for Integrated Services Packet Switching Networks," *IEEE Transactions on Networking*, vol. 5, no. 5, 1997, pp. 690–704.
20. D. Saha, S. Mukherjee, and S.K. Tripathi, "Carry-Over Round Robin: A Simple Cell Scheduling Mechanism for ATM Networks," *IEEE Transactions on Networking*, vol. 6, no. 6, 1998, pp. 779–796.
21. S. Shreedhar and G. Varghese, "Efficient Fair Queueing Using Deficit Round Robin," *IEEE Transactions on Networking*, vol. 4, no. 3, 1996, pp. 375–385.
22. D. Stiliadis and A. Varma, "Efficient Fair Queueing Algorithms for Packet Switched Networks," *IEEE Transactions on Networking*, vol. 6, no. 2, 1998, pp. 175–185.
23. L. Georgiadis, R. Guerin, and A. Parekh, "Optimal Multiplexing on a Single Link: Delay and Buffer Requirements," *IEEE Trans. on Information Theory*, vol. 43, 1997, pp. 1518–1535.
24. J. Liebeherr, D. Wrege, and D. Ferrari, "Exact Admission Control for Networks with a Bounded Delay Service," *IEEE Transactions on Networking*, vol. 4, 1996, pp. 885–901.
25. L. Georgiadis, R. Guerin, V. Peris, and K.N. Sivarajan, "Efficient Network QoS Provisioning Based on Per Node Traffic Shaping," *IEEE/ACM Transactions on Networking*, vol. 4, no. 4, 1996, pp. 482–501.
26. V. Sivaraman and F. Chiussi, "End-to-End Statistical Delay Guarantees Using Earliest Deadline First (EDF) Packet Scheduling," in *Proc. of GLOBECOM*, Rio de Janeiro, Brazil, Dec. 1999, pp. 1307–1312.
27. C-S. Chang and K.-C. Chen, "Service Curve Proportional Sharing Algorithm for Service-Guaranteed Multiaccess in Integrated-Service Distributed Networks," in *Proc. of GLOBECOM*, Rio de Janeiro, Brazil, Dec. 1999, pp. 1340–1344.
28. H-T Ngin, C-K Tham, and W-S Sho, "Generalized Minimum Queueing Delay: An Adaptive Multi-Rate Service Discipline for ATM Networks," in *Proc. of INFOCOM'99*, Piscataway, NJ, USA, 1999, pp. 398–404.
29. D. Saha, S. Mukherjee, and S.K. Tripathi, "Multirate Scheduling of vbr Video Traffic in ATM Networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 6, 1997, pp. 1132–1147.
30. D. Stiliadis and A. Varma, "Rate-Proportional Servers: A Design Methodology for Fair Queueing Algorithms," *IEEE Transactions on Networking*, vol. 6, no. 2, 1998, pp. 164–174.
31. D. Stiliadis and A. Varma, "Latency-Rate Servers: A General Model for Analysis of Traffic Scheduling Algorithms," *IEEE Transactions on Networking*, vol. 6, no. 5, 1998, pp. 611–624.
32. DC. Stephens and H. Zhang, "Implementing Distributed Packet Fair Queueing in a Scalable Switch Architecture," in *Proc. of INFOCOM98*, 1998, pp. 282–290.
33. V. Tabatabaee, L. Georgiadis, and L. Tassiulas, "QoS Provisioning and Tracking Fluid Policies in Input Queueing Switches," in *Proc. of INFOCOM'2000*, Tel Aviv, Israel, March 2000, vol. 3, pp. 1624–1633.
34. F.M. Chiussi and A. Francini, Implementing Fair Queueing in ATM Switches: The Discrete-Rate Approach," in *Proceedings of INFOCOM'98*, San Francisco, CA, USA, 1998, vol. 1, pp. 272–281.
35. 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 Communications*, vol. 18, no. 12, 2000, pp. 2684–2695.
36. D.A. Eckhardt and P. Steenkiste, "Effort-Limited Fair (ELF) Scheduling for Wireless Networks," in *Proc. of INFOCOM2000*, Tel Aviv, Israel, March 2000, pp. 1097–1106.
37. S. Lu, V. Bharghavan, and R. Srikant, "Fair Scheduling in Wireless Packet Networks," *IEEE Transactions on Networking*, vol. 7, no. 4, 1999, pp. 473–489.
38. P. Ramanathan and P. Agrawal, "Adapting Packet Fair Queueing Algorithms to Wireless Networks," in *Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'98)*, New York, NY, 1998, pp. 1–9.
39. N. Tse, I. Stoica, and H. Zhang, "Packet Fair Queueing Algorithms for Wireless Networks with Location-Dependent Errors," in *Proc. IEEE INFOCOM '98*, New York, NY, 1998, vol. 3, pp. 1103–1101.
40. P. Goyal, A. Greenberg, C.R. Kalmanek, W.T. Marshall, P. Mishra, D. Nortz, and K.K. Ramakrishnan, "Integration of Call Signaling and Resource Management for IP Telephony," *IEEE Network*, vol. 13, no. 3, 1999, pp. 24–32.
41. D.E. Wrege, "Multimedia Networks with Deterministic Quality-of-Service Guarantees," Ph.D. Thesis, University of Virginia, Aug. 1996.

A. Stamoulis holds degrees in Computer Engineering (Diploma, University of Patras, 1995), Computer Science (Master, University of Virginia, 1997), and Electrical Engineering (Ph.D., University of Minnesota, 2000). Three days after receiving his Ph.D. degree, he joined the AT&T Shannon Laboratory as a Senior Technical Staff Member.
stamouli@ece.umn.edu

G.B. Giannakis received his Diploma in Electrical Engineering from the National Technical University of Athens, Greece, 1981. From September 1982 to July 1986 he was with the University of Southern California (USC), where he received his MSc. in Electrical Engineering, 1983, MSc. in Mathematics, 1986, and Ph.D. in Electrical Engineering, 1986. After lecturing for one year at USC, he joined the University of Virginia in 1987, where he became a professor of Electrical Engineering in 1997. Since 1999 he has been a professor with the Department of Electrical and Computer Engineering at the University of Minnesota, where he now holds an ADC Chair in Wireless Telecommunications.

His general interests span the areas of communications and signal processing, estimation and detection theory, time-series analysis, and system identification—subjects on which he has published more than 125 journal papers, 250 conference papers and two edited books. Current research topics focus on transmitter and receiver diversity techniques for single- and multi-user fading communication channels, redundant precoding and space-time coding for block transmissions, multicarrier, and wide-band wireless communication systems.

G.B. Giannakis is the (co-) recipient of three best paper awards from the IEEE Signal Processing (SP) Society (1992, 1998, 2000). He also received the Society's Technical Achievement Award in 2000. He co-organized three IEEE-SP Workshops (HOS in 1993, SSAP in 1996 and SPAWC in 1997) and guest (co-) edited four special issues. He has served as an Associate Editor for the *IEEE Trans. on Signal Proc*. and the *IEEE SP Letters*, a secretary of the SP Conference Board, a member of the SP Publications Board and a member and vice-chair of the Statistical Signal and Array Processing Committee. He is a member of the Editorial Board for the *Proceedings of the IEEE*, he chairs the SP for Communications Technical Committee and serves as the Editor in Chief for the *IEEE Signal Processing Letters*. He is a Fellow of the IEEE, a member of the IEEE Fellows Election Committee, the IEEE-SP Society's Board of Governors, and a frequent consultant for the telecommunications industry.
georgios@ece.umn.edu