Scheduling for Quality of Service Guarantees via Service Curves *

Hanrijanto Sariowan and Rene L. Cruz
Dept. of Electrical & Computer Engineering
University of California, San Diego
La Jolla, CA 92093-0407
{sariowan,cruz}@ece.ucsd.edu

George C. Polyzos
Dept. of Computer Science & Engineering
University of California, San Diego
La Jolla, CA 92093-0114
polyzos@cs.ucsd.edu

Abstract

We propose a new scheduling policy, called SCED (Service Curve-based Earliest Deadline first), which provides guarantees to virtual circuits in packet-switched networks. This scheduling policy is developed for a general framework for service provisioning based on Service Curves proposed by Cruz. Instead of explicitly guaranteeing a specific quality of service measure, such as maximum delay, SCED guarantees the service curve for a connection. Quality of service guarantees for the connection can then be expressed as simple functions of the service curve guarantee and the traffic burstiness constraint of the connection. A simple and convenient condition under which SCED can simultaneously guarantee a set of service curves is proved. The service curve specification gives greater flexibility to a server in allocating its resources to meet diverse delay and throughput requirements. We demonstrate by an example that SCED provides a larger schedulability region than scheduling policies such as VirtualClock and PGPS with rate proportional assignment.

1 Introduction

Future high-speed packet-switched networks are expected to support a wide variety of services. These services will have traffic characteristics and quality of service requirements which may be dramatically different from one another. In ATM networks, each service may be transported via one or more separate virtual circuits, but these circuits or connections may share the same physical links and switching systems. This presents a formidable challenge to the network that has to efficiently allocate limited network resources to many connections by promoting sharing while also providing quality of service for each connection.

The issues of providing guaranteed service for virtual circuits in the networks have drawn considerable interest from the research community. Several new traffic models, both deterministic [2][3] and stochastic [12][17][1], have been proposed which have made end-to-end network analysis tractable and have yielded some bounds on performance measures such as delay, throughput, and backlog. Complementing these new traffic models, several link scheduling policies which provide guarantees in the network have been proposed, such as Delay-EDD [7], Jitter-EDD [16], RCSP [18], Hierarchical Round Robin [11], VirtualClock [19], Stop-and-Go Queueing [10], Fair Queueing [6], PGPS [13], and Leave-in-Time [8].

In order to provide guarantees, most scheduling policies specify some constraints on the input traffic of each connection, such as minimum interarrival time between packets and maximum packet size. The guarantees for a connection are usually expressed as complex functions of the traffic constraints of *all* connections. Hence, when the traffic constraints of a connection change, the guarantees for that connection change as well as those for other connections.

Cruz [5] (see also [4]) introduced the notion of *service curves* and proposed their use as a general framework for service characterization. A key feature of charactectizing service for a connection using a service curve is that the quality of service guarantees can be expressed as simple functions of the service curve and the traffic burstiness constraint of the connection. These functions are *independent* of the service curve guarantees and burstiness constraints of all other connections.

Another key feature of the service curve specification is that it gives greater flexibility to a server in allocating its resources to meet diverse delay and throughput requirements. Many rate based policies

*This work was supported in part by the National Science Foundation under the grants NCR 91-58618, NCR 94-15684, and NCR 91-19473.

0-8186-7180-7/95 $04.00

only provide a lower bound on throughput. Hence, a connection which has bursty traffic but requires a small maximum delay is forced to ask for a large throughput, although the average rate of the traffic may be small. By specifying an appropriate service curve, such a connection connection can inform the server that it needs a small average throughput, but may ocassionally need a large throughput during burst periods in order to meet the delay requirement of the connection. In other words, the service curve specification allows the throughput and delay requirements of a connection to be decoupled, and therefore allows a larger set of connections to be scheduled.

The versatility of the service curve specification motivates us to devise a new scheduling policy called SCED (Service Curve-based Earliest Deadline first) that guarantees the service curves for a set of connections. We show that if a simple sufficient condition is satisfied, the policy can guarantee these service curves independent of the traffic of the connections. We will demonstrate using an example that SCED can schedule a set of connections which are not schedulable by scheduling policies such as VirtualClock and PGPS with rate proportional assignment.

The remainder of this paper is organized as follows. In Section 2, we describe our model and introduce the notion of service curves. Traffic burstiness constraints and bounds on delay and backlog are briefly reviewed. In Section 3, we present the SCED policy. The condition for feasible allocation of service curves in a server is proved in Section 4. In Section 5, SCED is compared with other scheduling disciplines through an example. We conclude in Section 6 with a summary of our work and suggestions for future research.

2 Service Curves and Performance Guarantees

In this section, we define the notion of service curves and present associated maximum delay and backlog guarantees for a connection when its traffic is constrained. The service curve notion for a single server can be easily extended to the network case [5] which will not be considered in this paper.

Throughout this paper we assume that time is divided into slots, numbered $0, 1, 2, \ldots$ We consider a server which receives (and buffers) packets from M connections and sends up to c packets per slot. We call c the capacity of the server. Withous loss of generality, we assume that the server serves packets in a perfect "cut-through" manner, meaning that a packet which arrives in one slot may depart in the same slot.

Let $R_i^{in}[t]$ ($R_i^{out}[t]$) denote the number of packets from connection i arriving (leaving) the server during slot t, where t is a non-negative integer. Define $R_i^{in}[s,t]$ to be the number of packets arriving in the interval $[s,t]$, i.e. $R_i^{in}[s,t] = \sum_{m=s}^{t} R_i^{in}[m]$. If $s > t$, define $R_i^{in}[s,t] = 0$. Similarly, $R_i^{out}[s,t]$ is defined to be the number of packets leaving the server in the interval $[s,t]$. For simplicity, we will focus our discussion on a given connection and omit the subscript i for the rest of the section.

We assume that there is no packet stored in the server at the end of slot zero. Therefore, the number of packets from the connection which are stored in the server at the end of slot t, called the *backlog* for the connection at the end of slot t, is given by

$$B[t] = R^{in}[1,t] - R^{out}[1,t] \geq 0. \qquad (1)$$

The *virtual delay,* relative to t, suffered by the connection is denoted by $d[t]$ and defined to be:

$$d[t] = \min\{\Delta : \Delta \geq 0 \text{ and } R^{in}[1,t] \leq R^{out}[1,t+\Delta]\}. \qquad (2)$$

Note that if packets depart the server in the same order in which they arrive (FIFO), then $d[t]$ is an upper bound of the delay suffered by a packet that arrives in slot t. Having defined all the necessary terms, we are now ready to define service curves.

Definition 1 (Service Curves) *Let $S(\cdot)$ be a non-decreasing function, with $S(0) = 0$. We say that the server guarantees the service curve $S(\cdot)$ for the connection if for any t, there exists $s \leq t$ such that $B[s] = 0$ and $R^{out}[s+1,t] \geq S(t-s)$.*

Intuitively, $S(t-s)$ specifies the minimum number of packets from the connection that have to depart the server within some specific interval $[s+1,t]$, where t is any given slot and s is some slot, no later than t, in which the backlog of the connection is zero.

The next two theorems [5] show that if the arrival stream of the connection is constrained in the sense of Definition 2 below and the connection is guaranteed a service curve, then the delay and backlog sustained by the connection are bounded. Theorem 2 below is reminiscent of well-known sample path results [14] [15].

Definition 2 (Burstiness Constraints) *Given a non-decreasing function $b(\cdot)$, called an arrival curve, we say that the input traffic R^{in} is b-smooth if $R^{in}[s+1,t] \leq b(t-s)$ for all s and t satisfying $s \leq t$. In the special case where b is affine, i.e. $b(x) = \sigma + \rho x$, we say that R^{in} is (σ, ρ)-smooth.*

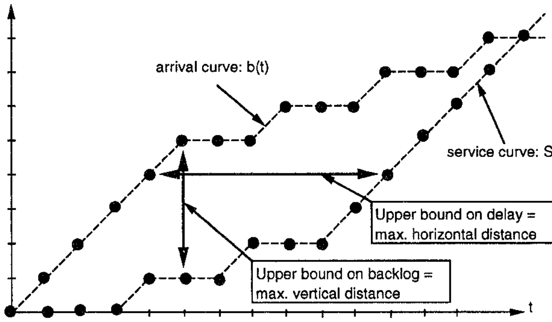

Figure 1. Example of an arrival curve and a service curve

Theorem 1 (Upper Bound on Delay) *Consider a server that guarantees the service curve $S(\cdot)$ for the connection and suppose that the input traffic R^{in} is b-smooth. Then for every t, the virtual delay $d[t]$ is upper bounded by*

$$d[t] \leq \max_{s:s\geq 1} \min\{\Delta : \Delta \geq 0 \text{ and } b(s) \leq S(s+\Delta)\}. \quad (3)$$

Theorem 2 (Upper Bound on Backlog) *Consider a server that guarantees the service curve $S(\cdot)$ for the connection and suppose that the input traffic R^{in} is b-smooth. Then for every t, the backlog $B[t]$ is upper bounded by*

$$B[t] \leq \max_{s:s\geq 0}\{b(s) - S(s)\}. \quad (4)$$

Figure 1 illustrates a service curve $S(\cdot)$ and an arrival curve $b(\cdot)$. Note that the upper bound on backlog can be graphically interpreted as the maximum *vertical* distance between the curves $S(\cdot)$ and $b(\cdot)$, while the upper bound on delay as the maximum *horizontal* distance between $S(\cdot)$ and $b(\cdot)$, roughly speaking.

Suppose the connection generates traffic that is b-smooth for some b and requires that the delay suffered through the server be at most d^{max} slots. Then, using Theorem 1, this delay guarantee can be made if the connection is guaranteed the service curve $S(\cdot)$, where $S(\cdot)$ is equal to $b(\cdot)$ shifted d^{max} units to the right, i.e.:

$$S(t) = \begin{cases} 0 & \text{, if } 0 \leq t \leq d^{max} - 1 \\ b(t - d^{max}) & \text{, if } t \geq d^{max}. \end{cases} \quad (5)$$

It is important to remark that although traffic characteristics of the connection are needed to make an intelligent choice of the appropriate service curve for the connection, they neither have to be enforced nor are required by the server in order to guarantee the service curve. The traffic characteristics of the connection may change while the server continues to guarantee the same service curve, and the new delay and backlog bounds can be easily recalculated using (3) and (4).

3 The SCED Policy

We consider a server which serves M connections. Suppose connection i requires that the service curve $S_i(\cdot)$ be guaranteed by the server for $i = 1, \ldots, M$. We would like to find a scheduling algorithm that guarantees the service curve for each connection. We would also like to identify a condition under which it is feasible for the server to simultaneously guarantee the service curve for each connection.

For simplicity of exposition, we assume that $S_i(\cdot)$ is integer valued. A server is said to be *empty* in slot t if all connections have zero backlog at the end of slot t, i.e. $B_i[t] = 0$ for $i = 1, \ldots, M$. For a non-negative integer t, define $\tau(t)$ as the last slot no later than t in which the server is empty, i.e.:

$$\tau(t) = \max\{s : s \leq t \text{ and } B_i[s] = 0 \text{ for } i = 1, \ldots, M\}. \quad (6)$$

We say that the server guarantees the *target output curve* $Z_i(\cdot)$ for connection i if for every t,

$$R_i^{out}[\tau(t) + 1, t] \geq Z_i(t) \quad (7)$$

where

$$Z_i(t) = \min_{\substack{s:\tau(t)\leq s\leq t \\ \text{and } B_i[s]=0}} \{R_i^{out}[\tau(t) + 1, s] + S_i(t - s)\}. \quad (8)$$

Lemma 1 *For connection i to be guaranteed the service curve $S_i(\cdot)$, it is sufficient for connection i to be guaranteed the target output curve $Z_i(\cdot)$.*

Proof: Since connection i is guaranteed the target output curve $Z_i(\cdot)$, then by (7) and (8), for every t, we have $R_i^{out}[\tau(t) + 1, t] \geq R_i^{out}[\tau(t) + 1, s^*] + S_i(t - s^*)$ where s^* denotes the slot in which the minimum in (8) is achieved. Subtracting $R_i^{out}[\tau(t) + 1, s^*]$ from both sides of the inequality, we obtain $R_i^{out}[s^* + 1, t] \geq S_i(t - s^*)$. Since $B_i[s^*] = 0$, by Definition 1 connection i is guaranteed the service curve $S_i(\cdot)$. □

Lemma 1 provides a motivation for the SCED scheduling policy. Consider a policy which assigns deadlines to arriving packets and sends them in the order of the deadlines such that (7) is satisfied. The target output $Z_i(t)$ is not known prior to time t but can be estimated by assuming that the backlog is not cleared.

Let the *arrival count* of a packet from connection i which arrives in slot u be a unique number arbitrarily selected from

$$\left[R_i^{in}[\tau(u-1)+1, u-1]+1, R_i^{in}[\tau(u-1)+1, u]\right].$$

Note that if only up to one packet arrives in a slot, the arrival count is simply the number of packets from connection i which arrive since the last time the server was empty. The complete scheduling policy is described as follows:

Definition 3 (SCED Policy) *Each packet is assigned a deadline and packets are served earliest deadline first in a work-conserving manner. Specifically, in a given slot u there are $\sum_{i=1}^{M}(B_i[u-1]+R_i^{in}[u])$ packets available for departure, and up to c packets with the smallest deadlines are selected for departure in slot u. The deadline D_i assigned to a packet from connection i which arrives in slot u is given by*

$$D_i = \min\{t : t \geq u \text{ and } Z_i(t; u-1) \geq n_i\} \tag{9}$$

where

$$Z_i(t;u) = \min_{\substack{s:\tau(u)\leq s\leq u \\ \text{and } B_i[s]=0}} \{R_i^{out}[\tau(u)+1, s] + S_i(t-s)\} \tag{10}$$

and n_i is the arrival count of the packet.

Note that $Z_i(u;u) = Z_i(u)$ where $Z_i(u)$ is the target output curve defined in (8). Also, $Z_i(t;u)$ is the "estimated value" of $Z_i(t)$ at the end of slot u.

We would like to remark here that although the deadline computation for a general service curve may seem rather complicated, there are many curves for which this computation is very simple and even almost identical to the deadline computation of other scheduling policies. For example, let $S_i(t) = \lfloor t \ Rate_i \rfloor$, $i = 1, \ldots, M$, where $1/Rate_i$ is a positive integer. The deadlines corresponding to these service curves can be computed as follows. For each connection i, the server maintains a variable VT_i which is initialized with zero and reset to zero every time the server is empty. When a new packet from connection i arrives in slot AT_i, the variable VT_i is updated as follows: $VT_i \leftarrow \max\{VT_i + 1/Rate_i, AT_i + 1/Rate_i - 1\}$. The arriving packet is then stamped with VT_i and packets are served in increasing order of their stamps. The computation of VT is very similar to that of $auxVC$ in VirtualClock [19], except for the resetting of the variable to zero every time the server is empty and the subtraction of one which accounts for the "cut-through" nature of our model. We believe that many scheduling policies can be characterized using service curves and corresponding SCED policies can be designed to emulate these policies.

Using the next two lemmas, we will show that if the server schedules packets according to the SCED policy and no deadlines are missed, then connection i is guaranteed the service curve of $S_i(\cdot)$ for $i = 1, \ldots, M$. A packet is said to *miss* its deadline if the packet departs *after* the deadline.

Lemma 2 *Assume that the server implements the SCED scheduling policy and deadlines are never missed. Then for any non-negative integer t, the number of packets from connection i which arrive and are assigned deadlines within the interval $[\tau(t)+1, t]$ is at least $Z_i(t)$.*

Proof: Define n^{max} as the number of packets from connection i that arrive and are assigned deadlines in $[\tau(t)+1, t]$. Note that n^{max} is also the largest arrival count of a connection i's packet which arrives and is assigned a deadline in $[\tau(t)+1, t]$. If $n^{max} = 0$, then $n^{max} = Z_i(t) = 0$ and the lemma is proved. Assume then that $n^{max} > 0$, and let t_D be the slot during which the packet from connection i with the arrival count n^{max} arrives. We have

$$n^{max} \leq Z_i(t; t_D - 1).$$

We consider two cases:

Case 1: $n^{max} = Z_i(t; t_D - 1)$

Since $Z_i(t; t_D - 1) \geq Z_i(t; t)$, the lemma is proved.

Case 2: $n^{max} < Z_i(t; t_D - 1)$

We need to consider two subcases:

Subcase 2a: Some packet from connection i arrives in $[t_D, t]$ after the n^{max}-th packet.

Suppose the packet with the arrival count $n^{max}+1$ arrives in slot t_1. By definition of t_D, this packet has a deadline greater than t. Furthermore, we have $t_1 \leq t$ and hence

$$\begin{aligned} n^{max} + 1 &> Z_i(t; t_1 - 1) \\ &\geq Z_i(t; t) \\ &= Z_i(t). \end{aligned}$$

If follows that $n^{max} \geq Z_i(t)$.

Subcase 2b: No packets from connection i arrive in $[t_D, t]$ after the n^{max}-th packet.

We need to establish the following claim for this subcase.

Claim *In Subcase 2b, if no deadlines are missed within the interval $[\tau(t)+1, t]$, there must exist a slot $u \in [t_D, t]$ in which $B_i[u] = 0$.*

Proof of claim: By contradiction: Suppose $B_i[u] > 0$ for all $u \in [t_D, t]$ and no deadlines are missed. By definition of *Subcase 2b*, there are no packets with an arrival count greater than n^{max} arrive in $[t_D, t]$. Thus

$$\begin{aligned} R_i^{out}[\tau(t)+1, t] &< R_i^{in}[\tau(t)+1, t] \\ &= R_i^{in}[\tau(t)+1, t_D] \\ &= n^{max}. \end{aligned}$$

Hence, there must be at least one packet missing its deadline in $[\tau(t)+1, t]$ since the number of packets departing in this interval is less than the number of packets which arrive and are assigned deadlines within the interval. This is a contradiction to the assumption that no deadlines are missed and thus the claim is established. ◇

We now proceed with the proof for *Subcase 2b*. By the Claim, $B_i[u] = 0$ for some $u \in [t_D, t]$. Furthermore, since there are no packets arriving in $[t_D, t]$ after the n^{max}-th packet, it follows that $B_i[t] = 0$ and hence

$$\begin{aligned} R_i^{out}[\tau(t)+1, t] &= R_i^{in}[\tau(t)+1, t] \\ &= R_i^{in}[\tau(t)+1, u] \\ &= n^{max}. \end{aligned}$$

Using the definition of $Z_i(t)$ and noting that $B_i[t] = 0$, we have

$$\begin{aligned} Z_i(t) &= \min_{\substack{s:\tau(t)\le s\le t \\ \text{and } B_i[s]=0}} \{R_i^{out}[\tau(t)+1, s] + S_i(t-s)\} \\ &\le R_i^{out}[\tau(t)+1, t] + S_i(0) \\ &= n^{max}, \end{aligned}$$

The lemma is proved for *Subcase 2b*. □

Lemma 3 *Assume that the server implements the SCED scheduling policy and deadlines are never missed. Then, connection i is guaranteed the service curve $S_i(\cdot)$ for $i = 1, \ldots, M$.*

Proof: Let $N_i^a[s, t]$ be the number of packets from connection i which arrive and are assigned deadlines within $[s, t]$. If deadlines are never missed, then for each i and every interval $[\tau(t)+1, t]$,

$$\begin{aligned} R_i^{out}[\tau(t)+1, t] &\ge N_i^a[\tau(t)+1, t] \\ &\ge Z_i(t). \end{aligned}$$

The second inequality follows from Lemma 2. Hence, connection i is guaranteed the target output curve $Z_i(\cdot)$ and by Lemma 1 is therefore guaranteed the service curve $S_i(\cdot)$. □

4 Feasible Allocation of Service Curves

In this section, we present and prove a condition on which a set of service curves can be simultaneously guaranteed by a server.

Theorem 3 (Feasible Allocation) *Given a set of M connections served by a server with capacity c, if $\sum_{i=1}^{M} S_i(t) \le ct$ for all non-negative integers t, then the SCED policy described in Definition 3 guarantees connection i the service curve $S_i(\cdot)$ for all $i = 1, \ldots, M$.*

We need the following additional lemma to prove the theorem.

Lemma 4 *Assume the server implements the SCED policy and at least one packet from connection i is assigned a deadline D. Then the number of packets from connection i which arrive and are assigned deadlines in $[\tau(D)+1, D]$ is at most $Z_i(D)$.*

Proof: (Lemma 4) Define n^{max} as the number of packets from connection i that arrive and are assigned deadlines in $[\tau(t)+1, t]$. Let t_D be the slot during which the packet from connection i with the arrival count n^{max} arrives.

We can express $Z_i(D)$ as follows

$$\begin{aligned} Z_i(D) = \min\{&Z_i(D; t_D - 1), \\ &\min_{\substack{s:t_D\le s\le D \\ \text{and } B_i[s]=0}} \{R_i^{out}[\tau(t)+1, s] + S_i(D-s)\}\} \end{aligned} \tag{11}$$

Since the n^{max}-th packet arrives in slot t_D and is assigned the deadline D, the following must hold

$$Z_i(D; t_D - 1) \ge n^{max}. \tag{12}$$

If there exists $s^* \in [t_D, D]$ in which the inner minimum in (11) is achieved, we have

$$\begin{aligned} R_i^{out}[\tau(t)+1, s^*] &= R_i^{in}[\tau(t)+1, s^*] \\ &\ge n^{max} \end{aligned}$$

since by definitions of t_D and n^{max}, $R_i^{in}[\tau(t)+1, s] \ge n^{max}$ for all $s \ge t_D$. If there is no such s^*, the inner minimum in (11) can be assumed to be infinite. Hence,

$$\min_{\substack{s:t_D\le s\le D \\ \text{and } B_i[s]=0}} \{R_i^{out}[\tau(t)+1, s] + S_i(D-s)\} \ge n^{max}, \tag{13}$$

and from (11), (12), (13), we obtain $Z_i(D) \ge n^{max}$. □

Proof: (Theorem 3) By Lemma 3, it suffices to show that if the condition $\sum_{i=1}^{N} S_i(t) \le ct$ is satisfied, no deadlines are missed. We argue by contradiction. Suppose the condition is satisfied and a deadline is missed.

Let t_m be the first deadline missed. Let t_s be the last slot in $[\tau(t_m)+1, t_m]$ in which a packet with a deadline greater than t_m is served. If there is no such slot, define $t_s = \tau(t_m)$. Define T to be the number of packets which *have to* depart in $[t_s+1, t_m]$, i.e. those which arrive and are assigned deadlines in $[\tau(t_m)+1, t_m]$ but do *not* depart in $[\tau(t_m)+1, t_s]$. We first will show that T must be greater than $c(t_m - t_s)$.

During $[\tau(t_m)+1, t_m]$, the server is not empty. Since SCED is work conserving, in this interval the server serves c packets per slot and hence in $[t_s+1, t_m]$ there are $c(t_m - t_s)$ packets served.

Clearly, all packets served in $[t_s+1, t_m]$ arrive in $[\tau(t_m)+1, t_m]$ and they are not served in $[\tau(t_m)+1, t_s]$. Furthermore, they should have deadlines at most t_m by the definition of t_s, and at least t_s+1 since otherwise a deadline smaller than t_m would have been missed. Hence, all packets served in $[t_s+1, t_m]$ contribute toward T.

Finally, by definition of T, the packet which misses its deadline in t_m is also counted in T, and hence

$$\begin{aligned} T &\geq c(t_m - t_s) + 1 \\ &> c(t_m - t_s). \end{aligned} \tag{14}$$

We now proceed to show that T is upper-bounded by $\sum_{i=1}^{M} S_i(t_m - t_s)$. Define $\mathcal{A}$ to be the set of connections which have zero backlog in t_s *and* have no packets with deadlines greater than t_m departing in $[\tau(t_m)+1, t_s]$. Define $\mathcal{B}$ to be the *complement* of $\mathcal{A}$.

We claim that only connections in $\mathcal{A}$, and none in $\mathcal{B}$, contribute packets toward T. In other words, all packets from connections in $\mathcal{B}$ which are assigned deadlines within $[\tau(t_m)+1, t_m]$ depart in $[\tau(t_m)+1, t_s]$ and hence none of them needs to depart in $[t_s+1, t_m]$. To see this, recall that in a given slot the server serves packets which have the smallest deadlines among all packets available for departure. Since in slot t_s a packet with a deadline greater than t_m departs, all backlogged packets in t_s must have deadlines greater than t_m. This implies that if a connection has positive backlog in t_s, all packets from the connection which have deadlines t_m or smaller have departed in $[\tau(t_m)+1, t_s]$.

Next, in the interval $[\tau(t_m)+1, t_m]$ packets from a given connection are assigned non-decreasing deadlines and served in non-decreasing order of their deadlines. This implies that if a connection has a packet with a deadline greater than t_m departing in $[\tau(t_m)+1, t_s]$, all packets from the connection which have deadlines t_m or smaller have departed in $[\tau(t_m)+1, t_s]$.

Finally, by definition of $\mathcal{A}$, all packets from a connection in $\mathcal{A}$ which depart in $[\tau(t_m)+1, t_s]$ have deadlines t_m or smaller. Hence, for a connection in $\mathcal{A}$, the number of packets which have to depart in $[t_s+1, t_m]$ is equal to the number of packets which are assigned deadlines within $[\tau(t_m)+1, t_m]$ less the number of packets which have departed in $[\tau(t_m)+1, t_s]$. Thus T can be written as

$$\begin{aligned} T &= \sum_{i\in\mathcal{A}} N_i^a[\tau(t_m)+1, t_m] - R_i^{out}[\tau(t_m)+1, t_s] \\ &\leq \sum_{i\in\mathcal{A}} Z_i(t_m) - R_i^{out}[\tau(t_m)+1, t_s] && (15) \\ &= \sum_{i\in\mathcal{A}} \min_{\substack{s:\tau(t_m)\leq s\leq t_m \\ \text{and } B_i[s]=0}} \{R_i^{out}[\tau(t_m)+1, s] + \\ &\qquad S_i(t_m - s)\} - R_i^{out}[\tau(t_m)+1, t_s] && (16) \\ &\leq \sum_{i\in\mathcal{A}} R_i^{out}[\tau(t_m)+1, t_s] + \\ &\qquad S_i(t_m - t_s) - R_i^{out}[\tau(t_m)+1, t_s] && (17) \\ &= \sum_{i\in\mathcal{A}} S_i(t_m - t_s) \\ &\leq \sum_{i=1}^{M} S_i(t_m - t_s) && (18) \end{aligned}$$

where (15) follows from Lemma 4, (16) from the definition of $Z_i(\cdot)$, and (17) from the fact that $B_i[t_s] = 0$ for $i \in \mathcal{A}$. Combining (14) and (18), we obtain $\sum_{i=1}^{M} S_i(t_m - t_s) > c(t_m - t_s)$, which is a contradiction to the hypothesis that $\sum_{i=1}^{M} S_i(t) \leq ct$ for all t. The theorem is proved. □

5 Comparing SCED To Other Scheduling Policies

In this section we will compare SCED to VirtualClock, PGPS, and EDF policies using an example. Consider two connections, C1 and C2, which are to be served by a server with capacity 1. The traffic of C1 and C2 is b_1-smooth and b_2-smooth, respectively. The arrival curves $b_1(\cdot)$ and $b_2(\cdot)$ are plotted in Figure 2. It can be observed from the figure that C1 and C2 have average rates of at most two-thirds and one-third per slot, respectively. Suppose C1 and C2 have different maximum delay requirements: C1 3 slots and C2 1 slot. Here C1 illustrates a connection that requires a large average throughput but can tolerate some delay. C2 illustrates a connection that only needs a small average throughput, but it is bursty and requires a

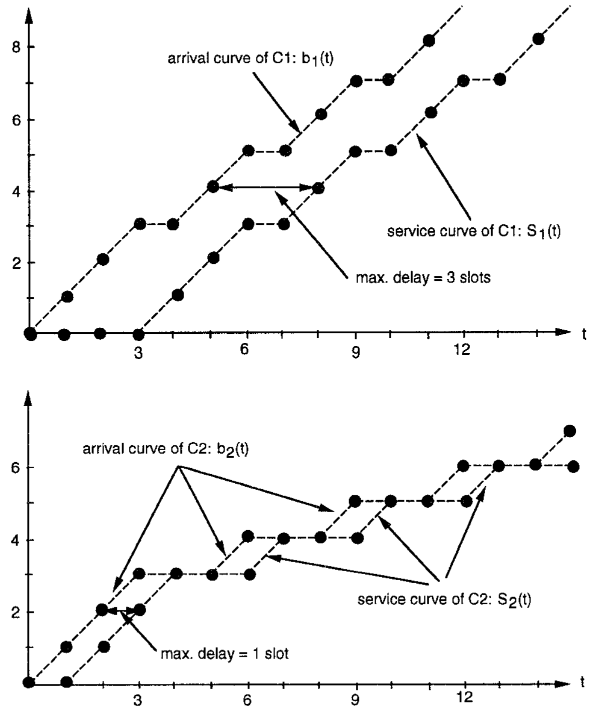

Figure 2. The arrival and service curves of (a) C1 (b) C2

small maximum delay. In Table 1, an example of arrival patterns of C1 and C2 is given. Each number in the Arrival Slots row represents the index of the slot in which a packet arrives.

Using equation (5), we can easily calculate the service curves $S_1(\cdot)$ and $S_2(\cdot)$ which are sufficient to meet the delay requirements of C1 and C2, respectively. The curves $S_1(\cdot)$ and $S_2(\cdot)$ are also plotted in Figure 2. Using Theorem 3, we can easily verify that simultaneous allocation of these two service curves is feasible. The SCED deadlines and departure slots corresponding to the example's arrival pattern are given in Table 1, from which we can see that the delay requirements of C1 and C2 are satisfied.

The VirtualClock [19] policy schedules packets in the following manner. For each connection i, the server maintains a variable $auxVC_i$ which is initialized with zero. When a packet from connection i arrives at time AT_i, $auxVC_i$ is updated as follows: $auxVC_i \leftarrow \max\{AT_i + VTick_i, auxVC_i + VTick_i\}$ where $VTick_i$ is the average packet interarrival time. Packets are stamped with $auxVC$ and served in increasing order of the stamps. For our example, $VTick$ for C1 and C2 is $\frac{3}{2}$ and 3, respectively. In Table 1 the deadlines and departures using VirtualClock are given and we can see that C1 suffers a maximum delay of 1 slot as required, but C2 has a maximum delay of 6 slots, a violation of the delay requirement of C2.

Generalized Processor Sharing (GPS) [13] specifies a non-negative parameter ϕ for each connection which determines the minimum fraction of bandwidth given to the connection during backlogged periods. At any time t in which connection i is backlogged, a packet from connection i is served at the instantaneous rate of $\phi_i / \sum_{j \in B_t} \phi_j$, where B_t denotes the set of connections which are backlogged at time t. PGPS tries to emulate GPS by serving packets in (approximately) increasing order of packet transmission finishing times, assuming that they are served by a GPS server.

The parameter ϕ can be assigned in several ways. One possible assignment is called rate proportional (RPPS) in which ϕ_i is assigned such that $c\phi_i / \sum_{j=1}^{M} \phi_j$ is no smaller than connection i's average rate. The RPPS assignment of ϕ's for our example is $\phi_1 = \frac{2}{3}$ and $\phi_2 = \frac{1}{3}$. The deadlines and departure slots for RPPS corresponding to the example's arrival pattern are identical to those for VirtualClock. Hence, the connections are not schedulable.

The assignment of ϕ however, is not constrained to the rate proportional assignment only. In fact, any assignment of ϕ results in bounded delay and backlog if the sum of the average rates of all connections is smaller than the capacity of the server. A connection which has a small average rate, but requires a small maximum delay may be assigned a large ϕ in order to meet its delay requirement. A procedure for assigning ϕ's for a set of connections with different burstiness and delay constraints was given in [13]. However, due to differences between their model and ours, we resort to an ad-hoc method. Constraining $\phi_1 + \phi_2 = 1$, we assign ϕ_2 the smallest value possible which can still provide a maximum delay of 1 slot for C1 for the arrival pattern in Table 1. It can be inspected that if a value of $\frac{3}{5}$ or smaller is assigned to ϕ_2, the maximum delay of 1 slot requested by C2 may be violated. Hence, a value slightly greater than $\frac{3}{5}$, denoted as $\frac{3}{5}^+$, is assigned to ϕ_2 and a value slightly smaller than $\frac{2}{5}$, denoted as $\frac{2}{5}^-$, to ϕ_1. Applying this assignment on the example's arrival pattern, C2 suffers a maximum delay of 1 slot as required. C1 suffers maximum delay of 4 slots, a violation of its delay requirement. So, even with this alternative assignment, PGPS is not able to schedule C1 and C2. We conjecture however that the difference in schedulability between SCED and PGPS with non-proportional assignment is small and maybe non-existent in some cases.

It is important to remark however, that with non-

		Connection C1						Connection C2				
	Arrival Slots	1	2	3	5	6	8	1	2	3	7	9
SCED	Deadlines	4	5	6	8	9	11	2	3	4	8	10
	Departure Slots	4(3)	5	6	8(7)	9	11	1	2	3(4)	7(8)	10
VirtualClock	auxVC & GPS Finish T.	$2\frac{1}{2}$	4	$5\frac{1}{2}$	7	$8\frac{1}{2}$	10	4	7	10	13	16
PGPS (rate prop.)	Departure Slots	1	2(3)	4	5(6)	7	8(9)	$\underline{3}(2)$	$\underline{6}(\underline{5})$	$\underline{9}(\underline{8})$	$\underline{10}$	$\underline{11}$
PGPS	GPS Finish Times	$3\frac{1}{2}^{+}$	6^{+}	7^{+}	9^{+}	11^{+}	12^{+}	$2\frac{2}{3}^{-}$	$4\frac{1}{3}^{-}$	6^{-}	$8\frac{2}{3}^{-}$	$10\frac{2}{3}^{-}$
(non rate prop.)	Departure Slots	2	5	6	8	$\underline{10}$	11	1	3	4	7	9
NPEDF	Deadlines	4	5	6	8	9	11	2	3	4	8	10
	Departure Slots	4(3)	5	6	8(7)	9	11	1	2	3(4)	7(8)	10

Table 1. Scheduling of packets using 4 scheduling policies*.

proportional assignments, PGPS no longer fully protects some connections from other misbehaved connections. This defeats the original purpose of scheduling for providing guarantees. Using the previous assignment, $\phi_1 = \frac{2}{5}^{-}$ and $\phi_2 = \frac{3}{5}^{+}$, consider the following arrival scenario: C2 misbehaves by sending a packet in every slot, while C1 adheres to its traffic specification. Under this pattern, C1 only receives an average throughput of slightly less than two-fifths per slot when in fact C1 needs an average throughput of two-thirds per slot.

The last scheduling policy we consider is the Non-Preemptive Earliest Deadline First (NPEDF). Using this discipline, each packet is assigned a deadline which is equal to the arrival time of the packet plus a specified maximum delay. Packets are served in increasing order of deadlines. This discipline has been recently shown [9] to provide the largest possible schedulability region for a set of connections, each of which specifies a burstiness constraint and a maximum delay. Hence, if a scheduling discipline such as SCED can schedule a set of connections, so can NPEDF. From Table 1 we can see that under the NPEDF policy C1 and C2 are indeed schedulable.

However, unlike SCED, the NPEDF policy does not provide protection from connections which violate their traffic constraints. A misbehaved connection which asks for a small maximum delay can easily steal bandwidth from other connections which ask for a larger maximum delay by sending more traffic than it initially declares. This will result in delay violations for the connections with a larger maximum delay.

*Underlined number indicates a departure slot in which the required maximum delay of C1 or C2 is violated. (·) indicates an alternative departure slot. The server is assumed to be 'cut-through' (a packet can depart in the same slot in which it arrives).

6 Conclusion

We have proposed a new scheduling policy called SCED which is based on a general framework for specifying guarantees in the network using service curves. A simple sufficient condition under which a set of service curves can be simultaneously guaranteed is proved. We also showed by an example that SCED provides a larger schedulability region than Virtual-Clock and PGPS under rate proportional assignment. SCED provides protection from misbehaved connections and easy computation of performance bounds under arbitrary traffic constraints.

For future work, we will study the extension of this scheduling policy to tandem networks. The concept of service curves can conceivably be extended to stochastic service curves along the same lines that deterministic burstiness constraints have been extended to stochastic constraints [12] [17] [1].

References

[1] C. S. Chang. Stability, queue length, and delay of deterministic and stochastic queueing networks. *IEEE Trans. on Automatic Control*, vol.39, no. 5, May 1994, pp. 913-931.

[2] R. L. Cruz. A calculus for network delay, Part I: Network Elements in Isolation, *IEEE Trans. on Information Theory,* vol. 37, no. 1, Jan. 1991, pp. 114-131.

[3] R. L. Cruz. A calculus for network delay, Part II: Network Analysis, *IEEE Trans. on Information Theory,* vol. 37, no. 1, Jan. 1991, pp. 132-141.

[4] R. L. Cruz. Service burstiness and dynamic burstiness measures: a framework. *Journal of High Speed Networks*, vol. 1, no. 2, 1992, pp. 105-127.

[5] R. L. Cruz. Quality of service guarantees in virtual circuit switched networks. Invited paper to appear in *IEEE Journal on Selected Areas in Communications*, 'Advances in the Fundamentals of Networking'.

[6] A. Demers, S. Keshav, S. Shenker. Analysis and simulation of a fair queueing algorithm. *Proc. ACM SIGCOMM '89*, 1989, pp. 1-12.

[7] D. Ferrari and D. Verma. A scheme for real-time channel establishment in wide-area networks. *IEEE Journal on Selected Areas in Communications,* vol. 8, no. 4, Apr. 1990, pp. 368-379.

[8] N. Figueira and J. Pasquale. Leave-in-time: a service discipline for control of real-time communications in a packet switching data network. To appear in *Proc. ACM SIGCOMM '95.*

[9] L. Georgiadis, R. Guerin, and A. Parekh. Optimal multiplexing on a single link: delay and buffer requirements. *Proc. IEEE INFOCOM '94,* vol. 2, 1994, pp. 524-532.

[10] S. J. Golestani. Congestion-free communication in high-speed packet networks, *IEEE Trans. on Communications,* vol. 39, no. 12, Dec. 1991, pp. 1802-1812.

[11] C. Kalmanek, H. Kanakia, and S. Keshav. Rate controlled servers for very high-speed networks. *Proc. IEEE GLOBECOM '90,* vol. 1, 1990, pp. 12-20.

[12] J. Kurose. On computing per-session performance bounds in high-speed multi-hop computer networks. *Proc. ACM Sigmetrics and Performance '92*, 1992.

[13] 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. on Networking*, vol. 1, no. 3, June 1993, pp. 344-57.

[14] E. Reich. On the integrodifferential equation of Takács. I. *Annals of Mathematical Statististics,* vol. 29, pp. 563-570, 1958.

[15] J. W. Roberts and J. T. Virtamo. The superposition of periodic cell arrival streams in an ATM multiplexer. *IEEE Trans. on Communications,* vol. 39, no. 2, Feb. 1991, pp. 298-303.

[16] D. Verma, H. Zhang, D. Ferrari. Delay jitter control for real-time communication in a packet switching network. *Proc. IEEE TRICOMM '91,* 1991, pp. 35-43.

[17] O. Yaron and M. Sidi. Performance and stability of communication networks via robust exponential bounds. *IEEE/ACM Trans. on Networking,* vol.1, no.3, June 1993, pp. 372-85.

[18] H. Zhang and D. Ferrari. Rate-controlled service disciplines. *Journal of High Speed Networks,* vol. 3, no. 4, 1994, pp. 389-412.

[19] L. Zhang. VirtualClock: a new traffic control algorithm for packet-switched networks. *ACM Trans. on Computer Systems,* vol. 9, no. 2, May 1991, pp. 101-24.