GPS SCHEDULING WITH GRACEFUL RATE ADAPTATION [*]

Anastasios Stamoulis
Department of Electrical Engineering
University of Virginia
Charlottesville, VA 22903

Jörg Liebeherr
Polytechnic University
Department of Electrical Engineering
Brooklyn, NY 11201

Abstract

This paper addresses a shortcoming of the widely used Generalized Processor Sharing (GPS) scheduling algorithm which can have significant impact on the provided service, however, which has been given little attention. Since, with GPS, the service rate received by a session is proportional to the number of backlogged sessions in the system, the service rate of a session may change abruptly if some other session becomes active. This may result in abrupt increases of delay of consecutive packets. In this paper, we present and analyze a new scheduler, called Slow-Start GPS (S^2GPS), which alleviates the problem. S^2GPS is a modification of GPS where a session does not receive its guaranteed service rate immediately after it becomes active. Instead, the service rate of a session is gradually increased.

1 Introduction

The *Generalized Processor Sharing* (GPS) scheduling method is known to support isolation and sharing in a QoS network [2, 4, 5]. In recent years many researchers have studied GPS scheduling in the context of packet switching [1, 2, 3, 6, 7, 9, 10, 11]. GPS can provide rate guarantees to the sessions it services. However, with GPS, a session that has been active for a long period of time can experience dramatic decreases in its service rate when some other previously idle session becomes active. The decrease of the service rates can be quite large, resulting in a possibly significant increase of the delay of consecutive packets of an active session.

In this study we show how to alleviate the problem of abrupt decrease of service rates with GPS. We propose a modification to GPS, called Slow-Start GPS (S^2GPS), that prevents abrupt rate changes and delay increases by gracefully degrading the service rate of active sessions. This is accomplished by the following modification to the original GPS scheduling method. Whenever a session becomes active and starts sending packets, this session is not assigned the full bandwidth at once, but gradually. The name *"slow-start"* was elected to indicate that the service rate of a newly active session is slowly increased when the session starts transmitting. As a result, the service rates of previously active sessions decrease smoothly.

The remainder of this paper is structured as follows. In Section 2 we discuss GPS and its packetized version, PGPS. In Section 3 we study a class of scheduling disciplines that alleviate the problem of abrupt degradation of service and we present the novel S^2GPS scheduler. In Section 4 we analyze the worst-case delays with S^2GPS. In Section 5 we define the packetized version of S^2GPS and show how it can be implemented using the concept of virtual time. Since space limitations do not allow us to include examples in this paper, we refer to [8] for a set of simulation experiments.

2 GPS/PGPS Scheduler

2.1 GPS Server

A Generalized Processor Sharing (GPS) scheduler is a work-conserving[1] scheduler that serves the incoming traffic at a fixed rate r. Each session i that is served by the scheduler is characterized by a weight ϕ_i. Let $S_i(\tau, t)$ be the amount of session i traffic that is served in the interval $(\tau, t]$. Define a session as *backlogged* whenever the queue Q_i of this session is not zero. Then, a GPS scheduler is defined as one for which

$$S_i(\tau,t)/S_j(\tau,t) \geq \phi_i/\phi_j \tag{1}$$

for any pair of sessions i and j. If $B(t)$ is the set of backlogged sessions at time instant t, then every session i in $B(t)$ is served at the instantaneous service rate of:

$$r_i(t) = r\phi_i / \textstyle\sum_{j \in B(t)} \phi_j \tag{2}$$

Therefore, a session i is guaranteed a minimum service rate of g_i for any time interval that it is backlogged:

$$g_i = r\phi_i / \textstyle\sum_{j=1}^{N} \phi_j \tag{3}$$

where N is the maximum number of sessions that are being served by the GPS scheduler.

GPS is an idealized scheduler in that it assumes that traffic is infinitely divisible; hence, it can serve all backlogged sessions simultaneously. However, in reality, only one session can receive service at a time, and a packet has to be fully transmitted before another packet starts being served. Thus, in actual networks the operations of GPS must be approximated. The most popular approximation of GPS is Packet-By-Packet Generalized Processor Sharing (PGPS) [4] which is defined as follows. Let $d_{i,gps}^{(p)}$ be the departure time of a packet p from session i under GPS. Then, PGPS is the service discipline that transmits packets in increasing order of $d_{gps}^{(p)}$'s.

In [4], it is proved that:

$$d_{i,PGPS}^{(k)} - d_{i,GPS}^{(k)} \leq L_{max}/r \qquad \forall i,k \tag{4}$$

where $d_{i,PGPS}^{(k)}$, $d_{i,GPS}^{(k)}$ are the departure times of the k-th packet of session i under GPS. In other words, a PGPS system cannot fall behind a GPS system by more than one maximum packet size. In [4], it was proved that for leaky bucket constrained sessions, GPS guarantees deterministic worst-case delays.

[*]This work is supported in part by the National Science Foundation under Grant No. NCR-9309224 and NCR-9624106.

[1]A scheduler is work-conserving if it is not idle if there is incoming traffic to be transmitted. Otherwise, it is non-work-conserving.

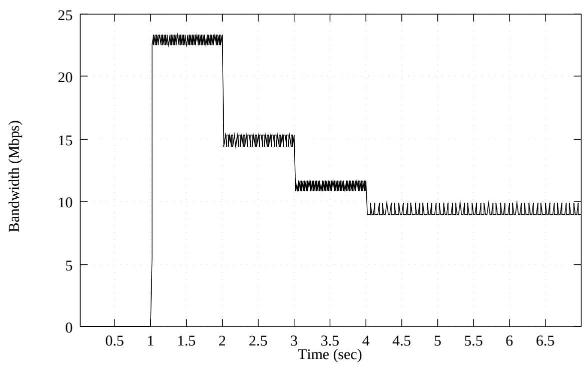


Figure 1: Available bandwidth for Session 1 under GPS.

2.2 GPS and abrupt decrease of service rates

When new sessions start transmitting packets, the service rates of the previously active sessions decrease abruptly. This abrupt decrease, which in some cases can be dramatic, can result in abrupt increases of delay and jitter. The decrease of service rates is a direct result of the fact that, under GPS, the service rate that a session receives is dependent on the number of backlogged sessions (as (2) indicates).

Let us present an example that illustrates how the service rate of a session under GPS can decrease rapidly. Suppose that we have a switch that operates at 45 Mbps. Further suppose that the switch serves five sessions. All sessions have the same weights, i.e., $\phi_0 = \phi_1 = \phi_2 = \phi_3 = \phi_4$. The guaranteed rate for every session is 9 Mbps. Let us assume that session 0 becomes active at time $t = 0$ sec, session 1 at $t = 1$ sec, . . ., session 4 at time 4sec.

In Figure 1 we plot the bandwidth that is available to session 1 in such a scenario.[2] As the figure indicates, when session 1 starts transmitting at time $t = 1$ sec, it immediately obtains its fair share of the bandwidth, which is 22.5 Mbps. When sessions 2, 3, and 4 start transmitting, the available bandwidth for session 1 is decreased rapidly to 15 Mbps, 11.25 Mbps and 9 Mbps, respectively. The figure clearly shows that GPS abruptly changes the service rates of a session whenever a new session becomes active.[3]

3 Slow-Start GPS Schedulers (S^2GPS)

In GPS, changes in the service rates occur when the set of the backlogged sessions changes. This happens when either "new" sessions become active or some sessions cease to be backlogged. When "new" sessions become backlogged, they demand their share of the bandwidth. This results in the decrease of the service rates of the "old" sessions and a potential increase of delays. On the other hand, when some sessions are no longer backlogged, the service rates of all other sessions have to be increased abruptly. In this paper we are only interested in the first case. Handling of the second case can be done by making the scheduler non-work-conserving.

Our approach is based on the following basic idea: when a new session k with ϕ_k becomes active at time instant t_k (We use t_k to denote the time instant when session k becomes active after an idle period), then the weight of this session will be gradually increased from an initial value of $\phi = 0$ to its final value, ϕ_k. Since the new session will not receive at once its fair, we anticipate that the service rates of all other sessions will not drop dramatically.

In a slow-start GPS scheduler, for each new session k, we use $T_k > 0$ to specify the length of the slow-start period, that is, the amount of time that has to pass before session k is assigned its fair share of the bandwidth. If session k becomes active at t_k, its service rate is increased in the interval $[t_k, t_k + T_k]$, and at time $t_k + T_k$, the session has obtained its fair share of the bandwidth. Let us denote the instantaneous service rates as $\hat{r}_k(t)$. A slow-start GPS scheduler is a work-conserving scheduler that maintains two sets of sessions, $B(t)$ and $B_{new}(t)$. $B(t)$ is the set of active sessions at time t, and $B_{new}(t) = \{k|\hat{r}_k(t) < r_k(t)\}$ is the set of all newly active sessions that have not yet acquired their fair share of the bandwidth at time t.

The slow-start GPS scheduler is characterized by the following properties:
1. The service rate of a newly active session in the slow-start phase is an increasing function in time. Thus, $0 \leq \hat{r}_k(t) \leq \hat{r}_k(t + \Delta t) \leq r_i(t + T_k)$ for $t_k \leq t < t + \Delta t \leq t_k + T_k$.
2. If session k is backlogged throughout the interval $[t_k, t_k + T_k]$, then after time $t_k + T_k$, session k is served at a rate at least as large as the service rate under GPS. Note that $\hat{r}_k(t)$ can be greater than $r_k(t)$ for $t \geq t_k + T_k$. This will happen when some session (other than k) is in the slow-start phase. Thus, $\hat{r}_k(t) \geq r_k(t)$ for $t \geq t_k + T_k$.
3. For any two connections i and j in $B(t) - B_{new}(t)$, we have that $\hat{r}_i(t)/\hat{r}_j(t) = \phi_i/\phi_j, \forall t$.

In this paper we investigate a slow-start GPS scheduler where the increase of service rates is carried out linearly with respect to time. Also, we assume that the length of the slow start period is identical for all sessions, that is, $T_k = T, \forall k$. Then the service rate of a session $k \in B_{new}(t)$ that becomes active at time t_k and is continuously backlogged in the interval $[t_k, t_k + T]$ is given by:

$$\hat{r}_k(t) = \frac{t - t_k}{T} r(t) \qquad : \ t_k \leq t \leq T + t_k \tag{5}$$

At time $t = T + t_k$, k is removed from $B_{new}(t)$ because it will have been assigned its fair share of the bandwidth. We refer to this scheduler as **Slow-Start GPS (S^{2}GPS)**.

In Figure 2 we illustrate the difference between GPS and S^{2}GPS. This figure depicts the service rate of session k as a function of time. The figure shows three events: At time t_k, session k becomes active, at time t_j, session j becomes active and at time t_x session k becomes idle. Under GPS, the service rate function $r_k(t)$ for a session k consists of linear horizontal segments. Under S^{2}GPS the service rate function does not change abruptly at the points in time where a session becomes active or a session leaves the system.

If $B(t)$ and $B_{new}(t)$ are constant in $[t_k, T + t_k]$, then a session $k \in B_{new}(t)$ is served at the instantaneous rate of:

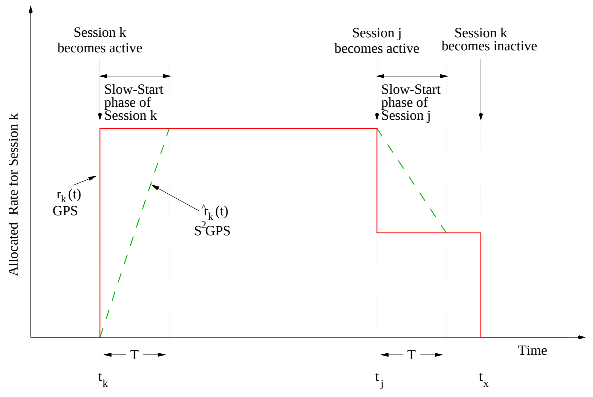
$$\hat{r}_k(t) = \frac{t - t_k}{T} \frac{\phi_k}{\sum\limits_{i \in B(t)} \phi_i} r \qquad : \ t_k \leq t \leq T + t_k \tag{6}$$

and a session $j \in B(t) - B_{new}(t)$ is served at the instantaneous rate of:

$$\hat{r}_j(t) = \frac{\phi_j \ (r - \sum\limits_{k \in B_{new}(t)} \hat{r}_k(t))}{\sum\limits_{i \in B(t) - B_{new}(t)} \phi_i} \qquad : t_k \leq t \leq T + t_k \tag{7}$$

[2] The figure depicts the results for PGPS, a packetized version of GPS (see Section 6). Note that the fluctuations of the service rate are caused by the fact that PGPS is an approximation of GPS. As a result, the total available bandwidth oscillates.

[3] In Figure 3 we show how our slow-start mechanism for GPS smoothes the changes of the service rate.



Session k becomes active
Session j becomes active
Session k becomes inactive
Slow-Start phase of Session k
Slow-Start phase of Session j
Allocated Rate for Session k
$r_k(t)$ GPS
$r_k(t)$ S^{2}GPS
T
T
Time
t_k
t_j
t_x

Figure 2: Service rate of Session k as a function of time in GPS and S^{2}GPS.

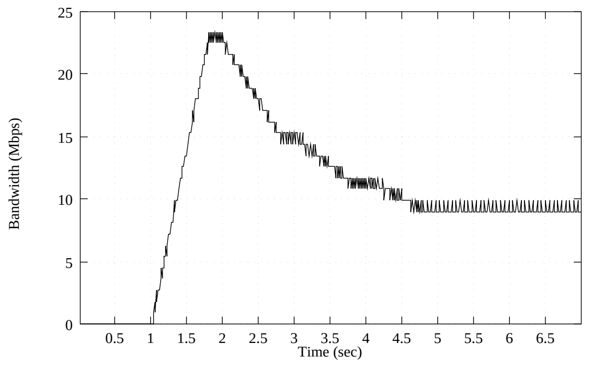


Figure 3: Available bandwidth for Session 1 under S^{2}GPS with $T = 0.8$ sec.

Equations (6) and (7) illustrate the fact that when a session is in the slow-start phase, the bandwidth, which corresponds to the difference between the fair share and the actual service rate, is distributed among the “old” sessions.

In Figure 3 we present the available bandwidth of a session for a ^{2}GPS scheduler with $T = 0.8$ sec. The input parameters are identical to those used for Figure 1.

One can make the following observations about S^{2}GPS. First, equation (1) still holds for any two sessions that belong to $B(t) - B_{new}(t)$. Second, in the long term, when all sessions are active and their service rates have assumed their steady-state values, a S^{2}GPS system behaves exactly in the same way as a GPS system. Third, if the duration T of the slow-start phase is small relatively to the time that a session is active, then S^{2}GPS will not necessarily cause a dramatic increase to the average delay of the session. However, depending on the value of T, we expect the worst-case delay in a slow-start GPS scheduler to be larger than that in GPS.

4 Analysis of S^{2}GPS

The linear change of service rate in S^{2}GPS increases the worst-case delay of new sessions that arrive in the system. In the following we derive bounds for the worst-case delay.

In S^{2}GPS, when a previously idle session becomes active, it is assigned its fair share of the bandwidth only gradually. As a result, we expect the worst-case delay in a S^{2}GPS system to be larger than the worst-case delay in the corresponding GPS system. For sessions that have been assigned their fair share of the scheduler bandwidth, the worst-case delay in S^{2}GPS system is expected to be the same as in a GPS system, as the session will receive service at a minimum rate of g_i. Therefore, the difference between the worst-case delays in S^{2}GPS and GPS system for a session i is obtained by evaluating the difference of service that the session receives until it is assigned its fair share of the bandwidth. In the following we will derive the worst-case delay bound for sessions that are constrained by leaky buckets.

In our analysis we take advantage of the so-called isolation property of GPS (which is also retained in S^{2}GPS). For the calculation of the worst-case delay, we consider a S^{2}GPS system where a maximum of N sessions can be admitted. Without loss of generality, we will calculate the worst-case delay for the k-th session that starts transmitting at time 0. As in GPS, in S^{2}GPS, session k experiences its worst-case delay when: (1) all the other sessions in the system are continuously backlogged, **and** (2) session k is greedy, i.e., $A_k(0,t) = \sigma_k + \rho_k t$. The first condition keeps the maximum service rate attained by session k to the minimum rate g_k. The second condition is required because the worst-case scenario will occur when a session transmits at its maximum allowable rate.

Let $Q_k(t)$ be the queue size (or backlog) of session k at time instant t. Then we have:

$$Q_k(t) = A_k(0,t) - S_k(0,t) \tag{8}$$

Let $\delta_k(t)$ be the delay of each arrival at time t. Then we have:

$$Q_k(t) = S_k(t, t + \delta_k(t)) \tag{9}$$

For ease of notation, we will use $A_k(t)$ for $A_k(0,t)$ and $S_k(t)$ for $S_k(0,t)$. Figure 4 depicts $A_k(t)$ and $S_k(t)$. As Figure 4 suggests, $\delta_k(t)$ is the horizontal distance between $A_k(t)$ and $S_k(t)$; $Q_k(t)$ is the vertical distance between $A_k(t)$ and $S_k(t)$.

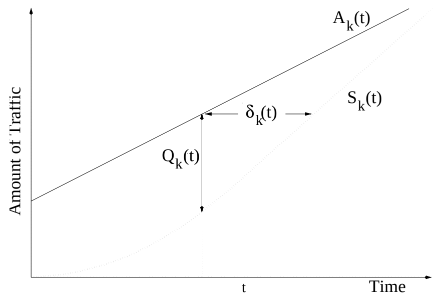


Figure 4: Relation of $\delta_k(t)$, $Q_k(t)$, $A_k(t)$, and $S_k(t)$.

Under S^{2}GPS, the service rate of session k is given by:

$$r_k(t) = \begin{cases} \frac{t}{T} g_k & : \quad t \le T \\ g_k & : \quad t > T \end{cases} \tag{10}$$

The amount of traffic $S_k(t_1, t_2)$ that is served in the interval $[t_1, t_2]$ and the instantaneous service rate are associated through the equation:

$$S_k(t_1, t_2) = \int_{t_1}^{t_2} r_k(t) dt \tag{11}$$

From (9) and (11) we have:

$$Q_k(t) = \int_{t}^{t+\delta_k(t)} r_k(\tau)\, d\tau \tag{12}$$

From (8) and (12) we obtain:

$$S_k(0,t) + \int_t^{t+\delta_k(t)} r_k(\tau)d\tau = A_k(0,t)$$

As $S_k(0,t) = \int_0^t r_k(\tau)d\tau$:

$$\sigma_k + \rho_k t = \int_0^{t+\delta_k(t)} r_k(\tau)\, d\tau \qquad (13)$$

Equation (13) will be used to evaluate the delay $\delta(t)$ as a function of time t. Hence, we are able to calculate the maximum delay $\delta_{\max}$, which is given in the following theorem. (Refer to [8] for a complete proof of Theorem 1.)

Theorem 1 *The worst-case delay δ_k^{max} of the (σ_k, ρ_k)-constrained session k is:*

$$\delta_k^{max} = \begin{cases} \frac{T}{2} + \frac{\sigma_k}{g_k} & : \quad T < \frac{2\sigma_k}{g_k} \\ \sqrt{\frac{2T\sigma_k}{g_k}} & : \quad \frac{2\sigma_k}{g_k} \le T < \frac{2\sigma_k g_k}{\rho_k^2} \\ \frac{\sigma_k}{\rho_k} + \frac{\rho_k T}{2g_k} & : \quad \frac{2\sigma_k g_k}{\rho_k^2} \le T \end{cases} \qquad (14)$$

provided that $g_k \ge \rho_k$.

5 A Packet-by-packet version of S^{2}GPS

S^{2}GPS assumes a fluid model where traffic is infinitely divisible. In reality, however, a scheduler can serve one packet at a time. In this section we define the packet approximation of S^{2}GPS, called Slow-Start Packetized Generalized Processor Sharing or S^{2}PGPS. Also, we show how S^{2}GPS can be implemented using the concept of virtual time.

5.1 S^{2}PGPS

In this Subsection, we define the packet approximation of S^{2}GPS, called S^{2}PGPS. S^{2}PGPS is the scheduling discipline that transmits packets in increasing order of their finishing times under the S^{2}GPS system. S^{2}GPS attempts to approximate the fluid model as closely as possible. Note that S^{2}PGPS is derived from S^{2}GPS in the same way that PGPS is derived from GPS [4]. The question that arises is whether S^{2}PGPS is a good approximation of a S^{2}GPS system. We will prove that this is indeed the case. Specifically, we can show that a S^{2}PGPS system cannot fall behind from the corresponding S^{2}GPS system by more than one maximum packet size. We will take advantage of the following results that are available for GPS/PGPS [4]:

1. Let p and q be packets in a GPS system at time τ, and suppose that packet p completes service before packet q if there are no arrivals after time τ. Then, the packet p will also complete service before the packet q for any pattern of arrivals after time τ.

2. Let $\hat{F}_p$, F_p be the times that packet p departs from the PGPS and GPS systems, respectively. Then for all packets p, $\hat{F}_p - F_p \le \frac{L_{\max}}{r}$.

3. Let $\hat{S}_i(0,\tau)$ be the amount of service that session i receives under S^{2}GPS. Then, for all times τ and for each session i $S_i(0,\tau) - \hat{S}_i(0,\tau) \le L_{\max}$.

It is not hard to show that the above properties also apply in S^{2}GPS. This is because the proofs of these properties for PGPS, which are given in [4], are not sensitive to time-dependent service rates. Thus, a S^{2}PGPS system cannot fall behind from the corresponding S^{2}GPS system by more than one packet size. These properties facilitate the translation of delay bounds under a S^{2}GPS system to the corresponding S^{2}PGPS system.

5.2 Virtual Time Implementation of S^{2}PGPS

In this section we present an implementation of the S^{2}PGPS based on virtual times. The concept of virtual time as a means of implementing PGPS was proposed in [2, 4]; the virtual time $V(t)$ in [2, 4] is used as a measure of progress in the system. When packets arrive, the scheduler assigns virtual time deadlines to them and serves packets in increasing order of these deadlines. The virtual time is set to zero at the beginning of a busy period and increases at the marginal rate of $1/\sum_{i\in B(t)} \phi_i$. Thus, during any system busy period $[t_1, t_2]$, $V(t)$ evolves as follows:

$$\begin{aligned} V(t_1) &= 0 && (15) \\ \partial V(t)/\partial t &= 1/\sum_{i\in B(t)} \phi_i \qquad : t_1 \le t \le t_2 && (16) \end{aligned}$$

which yields:

$$V(t) = \int_{t_1}^{t_2} 1/\sum_{i\in B(t)} \phi_i \; d\tau \qquad (17)$$

Let us define as an *event* in the system the arrival or the departure of a packet. Let e_j be the time that the j-th event in the system occurs. By observing that $B(t)$ is constant in any time interval during which no events occur, we obtain:

$$V(e_j+\tau) = V(e_j) + \tau/\sum_{i\in B(e_j)} \phi_i \qquad 0 \le \tau \le e_{j+1} - e_j \qquad (18)$$

For the p-th packet of the k-th session, the virtual start time $S_i^{(p)}$ and virtual finish time $F_i^{(p)}$ are defined as:

$$S_i^{(p)} = \max\{F_i^{(p-1)}, V(t_i^{(p)})\} \qquad (19)$$

$$F_i^{(p)} = S_i^{(p)} + L_i^{(p)}/\phi_i \qquad (20)$$

where $t_i^{(p)}$ is the arrival time of packet p and $L_i^{(p)}$ is the length of packet p. The scheduler serves packets in increasing order of their virtual finishing times.

An implementation of S^{2}PGPS with virtual time is not straightforward and must address the following two problems:

1. From (18) we have that the virtual time $V(t)$ is calculated as a function of the ϕ's of the sessions. Note, however, that the weights of the sessions in the slow-start phase have to be modified to reflect the increasing service rates that these sessions receive, and the decreasing service rates of the other, i.e., the previously active sessions.

2. For a packet p that is transmitted during the slow-start phase of a S^2GPS of a session k, the service rates of the session at the beginning and at the end of the transmission will be different. Correspondingly, the weight ϕ_k of a session k will take different values during the transmission of a packet. As (20) suggests, the deadline of a packet depends on the ϕ_k of the session. If the ϕ_k of session k is not constant over the transmission of a packet of the session, then it is not obvious how a deadline can be assigned to this packet.

We proceed to present solutions to these two problems. In Subsection 5.3 we show how the weights of sessions in the slow-start phase can be calculated. In Subsection 5.4 we show how the virtual finishing times are calculated in S^{2}GPS.

5.3 Definition of $\phi_k(t)$

Our goal is to define the ϕ_k of every session k in $B_{new}(t)$ as a function of time, $\phi_k(t)$, such that:

$$r_k(t) = \frac{\phi_k(t)}{\sum\limits_{j \in B(t) - B_{new}(t)} \phi_j + \sum\limits_{j \in B_{new}(t)} \phi_j(t)} r \tag{21}$$

Using (6), (7), and (21), we can calculate $\phi_k(t)$ for every session in the slow-start phase. We show how this can be done in the case when (i) only one session k is in the slow-start phase, and (ii) the set $B(t)$ of the backlogged sessions is constant in the time interval $[t_k, t_k + T]$. In Subsection 5.5 we discuss how this restriction can be relaxed.

To derive $\phi_k(t)$, recall that the service rate of session k in time interval $[t_k, t_k + T]$ is given by:

$$r_k(t) = \frac{t - t_k}{T} \frac{\phi_k}{\sum\limits_{i \in B(t)} \phi_i} r \tag{22}$$

As only session k is in the slow-start phase, (21) becomes:

$$r_k(t) = \frac{\phi_k(t)}{\sum\limits_{i \in B(t),\, i \neq k} \phi_i + \phi_k(t)} r \tag{23}$$

and (22) becomes:

$$r_k(t) = \frac{t - t_k}{T} \frac{\phi_k}{\sum\limits_{i \in B(t),\, i \neq k} \phi_i + \phi_k} r \tag{24}$$

From (23) and (24) we have:

$$\frac{\phi_k(t)}{\sum\limits_{i \in B(t),\, i \neq k} \phi_i + \phi_k(t)} = \frac{t - t_k}{T} \frac{\phi_k}{\sum\limits_{i \in B(t),\, i \neq k} \phi_i + \phi_k}$$

which implies:

$$\phi_k(t) = \frac{\frac{t-t_k}{T} \frac{\phi_k}{\sum\limits_{i \in B(t)} \phi_i} \sum\limits_{i \in B(t),\, i \neq k} \phi_i}{1 - \frac{t-t_k}{T} \frac{\phi_k}{\sum\limits_{i \in B(t)} \phi_i}} \tag{25}$$

Equation (25) clearly shows that $\phi_k(t)$ is an increasing function of time t. At time $t = t_k + T$, we have that $\phi_k(t_k + T) = \phi_k$ which implies that session k will be assigned its fair share of the bandwidth.

5.4 Virtual Finishing Time in S^{2}PGPS

Let p be a packet of session k that is transmitted during the slow-start phase of session k. In this subsection we will show how the virtual finishing time of this packet can be computed upon the arrival of the packet. Note that the virtual finishing time is used as the deadline with which the packet p will be tagged upon its arrival.

As (25) suggests, the weight ϕ_k of a session k in the slow-start phase changes during the transmission of a packet p of this session. We calculate an average value of the weight ϕ_k of the session over the transmission of the packet and we call it the "effective" value $\phi_{eff}^{(p)}$. Using $\phi_{eff}^{(p)}$, it is possible to calculate the virtual finishing time as $S_k^{(p)} + \frac{L_k^{(p)}}{\phi_{eff}^{(p)}}$. In other words, we are able to use (20) provided that we have calculated $\phi_{eff}^{(p)}$. We will show how $\phi_{eff}^{(p)}$ can be calculated for a session k in the slow-start phase. We assume that only session k is in the slow-start phase and that $B(t)$ is constant in $[t_k, t_k + T]$.

Let w_p denote the elapsed time between the end of the transmission of packet p and the arrival of the first packet of session k. Clearly, the transmission of packet p ends at time $t_k + w_p$. We can calculate w_p in terms of w_{p-1}. As the transmission of packet p will start at $t_k + w_{p-1}$ and end at $t_k + w_p$, we have:

$$\begin{aligned} L_k^{(p)} &= \int_{t_k + w_{p-1}}^{t_k + w_p} r_k(t)\, dt \\ &= \int_{t_k + w_{p-1}}^{t_k + w_p} \frac{t - t_k}{T} \frac{\phi_k}{\sum_{i=1}^{N} \phi_i} r\, dt \\ &= \int_{w_{p-1}}^{w_p} \frac{\tau}{T} \frac{\phi_k}{\sum_{i=1}^{N} \phi_i} r\, dt \end{aligned}$$

which yields:

$$w_p = \sqrt{\frac{2L_k^{(p)} \sum_{i=1}^{N} \phi_i}{r\phi_k} + (w_{p-1})^2} \tag{26}$$

However, packet p is served at a rate $\frac{r\phi_{eff}^{(p)}}{\sum\limits_{i=1,\, i \neq k}^{N} \phi_i + \phi_{eff}^{(p)}}$. As the transmission of the packet takes time $w_p - w_{p-1}$, we have:

$$\frac{r\phi_{eff}^{(p)}}{\sum\limits_{i=1\, i \neq k}^{N} \phi_i + \phi_{eff}^{(p)}} (w_p - w_{p-1}) = L_k^{(p)}$$

which yields:

$$\phi_{eff}^{(p)} = \frac{L_k^{(p)} \sum\limits_{i=1,\, i \neq k}^{N} \phi_i}{r(w_p - w_{p-1}) - L_k^{(p)}} \tag{27}$$

Using (26) and (27), we can now devise a procedure for S^{2}GPS that assigns a deadline to an incoming packet of a new session. The deadlines of packets from sessions that are not in the slow-start phase can be directly calculated using (20). In the following, we present pseudo-code for the algorithm that assigns deadlines to packets from sessions that are in the slow-start phase:

```
procedure Assign_Deadline_To(Packet $p$, session $k$)
1. if $p = 1$
2.     set $w_0 = 0$
3.     add session $k$ to $B_{new}$
4. if $w_{p-1} < T$
5.     calculate $w_p$ using (26)
6.     calculate $\phi_{eff}^{(p)}$ using (27)
7.     assign a deadline to packet $p$ using (20)
8. else /* the slow-start phase of session $k$ is over */
9.     remove $k$ from $B_{new}(t)$
```

When the first packet of session k arrives, session k enters a slow-start phase. Thus it is inserted to $B_{new}(t)$ and

w_0 is initialized to 0. Then w_1 is calculated from (26) and $\phi^{(1)}_{eff}$ is calculated from (27). A deadline to the first packet of the session packet is assigned using (20). When the p-th packet of session k arrives, the scheduler has to check if the session is still in the slow-start phase. This check can be carried out in the following way. If the $(p-1)$-th packet departs before the end of the slow-start phase, i.e., $w_{p-1} < T$, then packet p will also be transmitted during the slow-start phase of session k. Thus, w_p is calculated using (26) and $\phi^{(p)}_{eff}$ is calculated using (27). However, if $w_{p-1} > T$, then the slow-start phase of session k is over. session k is removed from $B_{new}(t)$ and its ϕ is set to ϕ_k for all the other packets from session k.

5.5 Relaxing the Assumption

So far we have shown how a S^{2}PGPS system can be implemented by assigning deadlines to packets of sessions in the slow-start phase provided that the following conditions hold:

Condition C_1: Only one session is in the slow-start phase.
Condition C_2: B(t) is constant if a session is in the slow-start phase.

If this assumption holds, then S^{2}PGPS provides the following guarantees:
Guarantee G_1: The new session will experience a linear increase of its service rate.
Guarantee G_2: The worst-case delay bounds as given in Equation (14) will hold.
Guarantee G_3: Previously active sessions experience smooth decreases of their service rates.

We now relax this assumption and examine the impact on guarantees G_1, G_2, and G_3, if conditions C_1 or C_2 do not hold.

Let us suppose that C_1 does not hold. We will show that G_2 and G_3 still hold. When C_1 does not hold, there is a time period where $n > 1$ sessions $k_1, k_2, \ldots, k_n$ are in the slow-start phase. Without loss of generality we can assume that session k_1 first entered the slow-start phase, then session k_2 and so on, i.e., $t_{k_1} \le t_{k_2} \le \ldots \le t_{k_n}$. In this scenario, the increase of the service rate of session k_1 will be linear only until the time that session k_2 enters the slow-start phase. In other words, in the time interval $[t_{k_1}, t_{k_2}]$ the service rate of session k_1 increases linearly. After time t_{k_2} session k_1 sees its service rate increase but not in a linear fashion. This is due to (27) not taking into account the ϕ 's of sessions $t_{k_2} \ldots t_{k_n}$. However, at time $t_{k_1} + T$, session k_1 is guaranteed to be served at a minimum rate g_k as given by (3). Similarly, the service rates of sessions k_2, ..., k_n also increase gradually but not linearly. Thus, when a new session enters the system, it is *always* served at a gradually increasing rate; the increase is linear only if assumptions C_1 and C_2 hold. As a result, G_3 holds.

Next we prove that G_2 holds even if C_1 does not hold. Recall that the delay bounds are computed by making worst-case assumptions, i.e., all $N-1$ sessions are continuously backlogged and only one session k is in the slow-start phase. Let us denote the service rate of session k as $r^{one}_k(t)$ when only session k is in the slow-start phase and $r^{many}_k(t)$ when several sessions are in the slow-start phase. We have $r^{one}_k(t) \ge r^{many}_k(t)$ since S^{2}GPS is work-conserving and a session in the slow-start phase is always served at a rate smaller than the service rate the session gets when it is not in the slow-start phase. Hence, when C_1 does not hold, a session in the slow-start does not experience the worst-case delays, and G_2 holds.

Thus, relaxing C_1 does not have any detrimental effects on S^{2}PGPS. It turns out that this is also the case with relaxing assumption C_2; we will show that G_2 and G_3 still hold. When C_2 does not hold, then some previously active sessions are removed from the system while some "new" sessions are still in the slow-start phase. If an "old" session ceases to be backlogged, its service rate is distributed to all remaining sessions in the system. The sessions in $B_{new}(t)$ will experience a sudden increase in the service rates. As their service rates will increase, their delays will decrease; hence, G_2 will hold. However, this increase of service rates will not be at the expense of the remaining sessions in $B(t) - B_{new}(t)$, as these sessions will also take a significant part of the bandwidth that was made available. Thus the smooth decrease of service rates is guaranteed in this case, too, and G_3 will hold. Hence, the slow-start nature of S^{2}PGPS is preserved.

6 Conclusions

In this paper we have shown that the GPS scheduling discipline is unable to provide graceful degradation of service since the service rates of active sessions decrease abruptly when new sessions start transmitting. We have proposed and analyzed a modification to GPS, called Slow-Start GPS or S^{2}GPS that remedies this problem. We have shown how S^{2}GPS can be implemented in packet-switched networks; the packetized version of S^{2}GPS, S^{2}PGPS, can be implemented using the concept of virtual time.

References

[1] J. C. R. Bennett and H. Zhang. WF2Q: Worst-case fair weighted fair queueing. In *Proc. IEEE Infocom '96*, March 1996.

[2] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. In *Proc. ACM Sigcomm'89*, pages 1–12, 1989.

[3] S. J. Golestani. A self-clocked fair queueing scheme for broandband applications. In *Proc. IEEE Infocom '94*, pages 636–646, 1994.

[4] 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*, 1(3):344–357, June 1993.

[5] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in intergrated services networks: the multiple node case. *IEEE/ACM Transanctions on Networking*, 2:137–150, April 1994.

[6] D. Saha, S. Mukherjee, and S. H. Tripahi. Carry-over round robin: a Simple cell scheduling mechanism for ATM networks. In *Proc. IEEE Infocom'96*, pages 630–637, 1996.

[7] S. Shreedhar and G. Varghese. Efficient fair queueing using deficit round robin. *IEEE Transanctions on Networking*, 4(3):375–385, June 1996.

[8] A. Stamoulis and J. Liebeherr. S^2GPS: Slow-Start Generalized Porcessor Sharing. Technical Report CS-97-03, University of Virginia, 1997.

[9] D. Stiliadis and A. Varma. Design and analysis of frame-based fair queueing: a new traffic scheduling algorithm for packet-switched networks. In *Proc. ACM Sigmetrics'96*, pages 104–115, May 1996.

[10] O. Yaron and M. Sidi. Generalized processor sharing networks with exponentially bounded burstiness arrivals. *Journal of High Speed Networks*, 3:375–387, 1994.

[11] Z. L. Zhang, D. Towsley, and J. Kurose. Statistical analysis of generalized processor sharing scheduling discipline. In *Proc. ACM Sigcomm'94*, pages 68–77, August 1994.