2001

Adaptive Proportional-Delay Differentiated Services: Characterization and Performance Evaluation

Matthew K. H. Leung

John C. S. Lui

David K.Y. Yau
Purdue University, yau@cs.purdue.edu

Report Number:
01-001

ADAPTIVE PROPORTIONAL DELAY DIFFERENTIATED SERVICES: CHARACTERIZATION AND PERFORMANCE EVALUATION

Matthew K. H. Leung
John C.S. Lui
David K.Y. Yau

CSD TR #01-001
February 2001

Adaptive Proportional Delay Differentiated Services: Characterization and Performance Evaluation

Matthew K. H. Leung, John C. S. Lui and David K. Y. Yau

***Abstract*— We examine a proportional-delay model for Internet differentiated services. Under this model, an ISP can control the waiting time "spacings" between different classes of traffic. Specifically, the ISP tries to ensure that the average waiting time of class i traffic relative to that of class $i-1$ traffic is kept at a constant specified ratio. If the waiting time ratio of class $i-1$ to class i is greater than one, the ISP can legitimately charge users of class i traffic a higher tariff rate (compared to the rate for class $i-1$ traffic), since class i users consistently enjoy better performance than class $i-1$ users. To realize such proportional-delay differentiated services, we use the time-dependent priority scheduling algorithm. We formally characterize the feasible regions in which given delay ratios can be achieved. Moreover, a set of control parameters for obtaining the desired delay ratios can be determined by an efficient iterative algorithm. We also use an adaptive control algorithm to maintain the correctness of these parameters in response to changing system load. Experiments are carried out to illustrate the short-term, medium-term and long-term relative waiting time performances for different service classes under Poisson, Pareto, MMPP and mixed traffic workloads. We also carry out experiments to evaluate the achieved end-to-end accumulative waiting times for different classes of traffic which traverse multiple hops under our service model.**

1 Introduction

Nowadays, the Internet is being used for many different user activities such as emails, software distribution, video and audio entertainment, e-commerce, and real-time games. Network applications that support these activities have diverse service requirements. For example, email requires reliable data delivery but can tolerate a relatively wide range of packet delays, while video and audio applications can tolerate a certain level of packet loss, but have stringent end-to-end delay requirements. Although some of these applications are designed to be adaptive to available network resources, they nevertheless expect different levels of service from the network in order to have good performance. Therefore, there is a growing need to provide an alternative Internet service model to the conventional one-size-fits-all best-effort service model.

One approach to solving this problem is the *Integrated Services (IntServ)* model proposed by the IETF. IntServ is inherently reservation based – to achieve predictable performance, an application is expected to reserve resources, such as network bandwidth and buffers, using a protocol like RSVP [12]. This raises two important deployment issues. First, *all* the routers along an end-to-end network path must be RSVP-capable in order to realize IntServ benefits. Second, a router has to manage per-flow state and perform per-flow processing. This makes it difficult for IntServ to scale well to tens of thousands of network flows. Although there are proposals for alleviating these difficulties [5, 19], designing a scalable IntServ model is still an open and challenging problem.

Recently, another service model known as *Differentiated Services (DS)* is proposed by the IETF and has received a lot of attention [15]. Under the DS model, traffic flows are aggregated and identified as classes. Since the number of DS traffic classes is expected to be far fewer than the number of flows in IntServ, DS is much less susceptible to the scalability problem. Rather than providing end-to-end performance guarantees for individual flows like IntServ, the DS service objective is to differentiate among classes of traffic using *per-hop* packet forwarding behaviors. In general, there are two approaches for delivering the DS service model: *absolute* differentiated services and *relative* differentiated services.

Under the absolute DS approach, the goal is to achieve performance measures similar to those in the IntServ model, but without keeping per-flow state within routers. Two most prominent schemes for absolute DS are premium services [16] and assured services [1]. In [13, 17], the authors present some elegant mathematical models for analyzing the performance of the absolute DS service model. In [20], the authors illustrate that in order to provide service assurance with coarse spatial granularity and high network utilization, some form of route pinning is required.

In [2], the authors propose a differentiated services model which provides proportional performance spacings, rather than absolute spacings. The goal is to give better performance to class i traffic, relative to class $i-1$ traffic, with a "fixed" quality spacing. If the goal is *consistently* achieved, then class i users will see a better performance than class $i-1$ users. In return, the ISP can legitimately charge class i traffic a higher tariff rate than class $i-1$ traffic. In [2], the authors propose two algorithms, called BPR and WTP, respectively, for implementing proportional *delay* differentiation. For WTP scheduling in particular, they show that in order to achieve a delay ratio of r between two traffic

Matthew K. H. Leung and John C. S. Lui are with the Department of Computer Science & Engineering, The Chinese University of Hong Kong; David K. Y. Yau is with the Department of Computer Science, Purdue University, West Lafayette, IN 47907.
J. C. S. Lui was supported in part by the CUHK Direct Grant and the RGC Earmarked Grant; D. K. Y. Yau was supported in part by the National Science Foundation under grant numbers EIA-9806741 and CCR-9875742 (CAREER), CERIAS, and the Indiana 21st Century Research and Technology Fund.

classes when the system is nearly 100% utilized, the corresponding control parameters should also be set to have ratio r. As we will illustrate, however, the WTP control parameters should in fact depend on the *distribution* of traffic loads. We also formally illustrate the conditions under which given delay ratios are feasible. Specifically, our paper addresses the following questions:

- Given desired waiting time ratios for N traffic classes, under what conditions (e.g., traffic load distribution for the N classes) can feasible WTP control parameters be found so as to achieve the waiting time ratios?
- Given that the waiting time ratios are feasible, how can one efficiently obtain WTP control parameter values that will achieve the waiting time ratios?
- Given the obtained control parameters, can we maintain the waiting time spacings at different time scales?
- Given that real traffic workloads are time-varying in nature, how can we adapt to the time-varying traffic and still be able to maintain the waiting time ratios?

The balance of the paper is organized as follows. In Section 2, we review proportional differentiated services as it is described in [2]. In Section 3, we characterize and analyze the performance of a time-dependent-priority algorithm (which is the same as the WTP algorithm[2]) in achieving proportional delay differentiations. We discuss the *conditions* under which feasible control parameters for the TDP algorithm exist and how they can affect a given set of quality spacings. We also present an efficient iterative method for finding the values of these control parameters when they exist. In Section 4, we present several dynamic adjustment algorithms to handle time-varying traffic. In Section 5, we present experimental results that illustrate the performance of our methods. In particular, we compare waiting time spacings achieved between different classes of traffic using the control parameters in [2] versus using those obtained by our proposed iterative method. We present waiting time spacing results under different time scales, different traffic arrival patterns and time varying traffic patterns. Experiments are carried out to illustrate the achieved end-to-end accumulative waiting times for different classes of traffic under proportional-delay differentiated services. Section 6 concludes.

2 Background

In this section, we review the proportional differentiation service model proposed in [2]. The model has two objectives. First, it should provide *consistent* service differentiation between classes; i.e., a class with higher advertised quality should consistently outperform a class with lower advertised quality. Second, it should allow the quality spacings between classes to be adjusted based on pricing and other criteria. For example, it should be possible to configure the average packet delay of a higher quality service class to be, say, 80% of the delay of a lower quality class. Further, the authors stipulate that these two goals should be met even for "short" timescales.

In [2], the authors propose two scheduling algorithms for approximating the proportional DS model for *delay* differentiation under *heavy-load* conditions. The first one, called the backlog-proportional rate (BPR) scheduler, is based on GPS, but with the modification that the class service rates are dynamically adjusted so that they are in proportion to the corresponding ratios of measured class loads. Specifically, let $r_i(t)$ be the service rate that is assigned to the queue i at time t. If queue i is empty at time t, $r_i(t) = 0$. For two backlogged queues, i and j, the service rate allocation in BPR satisfies the proportionality constraint:

$$\frac{r_i(t)}{r_j(t)} = \frac{b_i}{b_j}\frac{q_i(t)}{q_j(t)}$$

where $q_i(t)$ is the backlog of queue i at time t, and b_i is a set of control variables where $0 \leq b_1 \leq b_2 \leq \ldots \leq b_N$. When the system utilization tends to one, the ratios of the b_i's tend to the target long-term waiting time ratios. The service rate of each class during a busy period can be calculated from the work-conservation constraint:

$$\sum_{i=1}^{N} r_i(t) = R$$

where R is the link capacity. The BPR algorithm exhibits *sawtooth-type* delay variations over short timescales[2].

The second algorithm, called the waiting-time priority (WTP) scheduler, is based on Kleinrock's Time-Dependent-Priorities (TDP) algorithm [8]. Specifically, the priority of a packet from flow i at time t is proportional to the waiting time of that packet at time t, where the proportionality constant, denoted by s_i (using the notation in [2]), is a service parameter for flow i. In other words, the priority of a packet in the queue i at time t is

$$p_i(t) = w_i(t)s_i$$

where $w_i(t)$ is the waiting time of the packet at time t. Using simulations, the authors [2] show that, under heavy system load, the relative average delay experienced by two flows, say i and j, in a WTP server has value close to s_i/s_j, for monitoring timescales as short as a few tens of packet transmission times. Hence, WTP *approximates* the proportional delay differentiation model under *heavy-load conditions* (e.g., when the system utilization is close to 1). The reason it is an approximation is that, on the one hand, it is known that consistent quality spacing cannot be achieved over timescales that are arbitrarily short. On the other hand, no conditions are given in [2] to assess feasibility given a certain value of the monitoring timescale. Hence, the notion of "short" timescales remains imprecise.

Given the lack of a complete characterization of the proportional DS model, we set out to further evaluate its theoretical and physical properties. Our objectives are (1) to further contribute to the understanding of feasibility conditions for achieving proportional-delay differentiated services, and (2) when it is feasible, to derive the values of the control parameters that can achieve given target delay spacings. We mainly focus on the WTP algorithm, which is shown to be highly effective in [2]. We use an analytical approach to characterize the feasibility for achieving proportional delays. We found that, independent of the timescale parameter, system utilization impacts feasibility. Further, we show that when system utilization varies (i.e., traffic intensities are time-varying in practice), the per-flow WTP control parameters should be dynamically adjusted to maintain the target delay differentiations when feasible.

3 Characterization & Performance Analysis

In this section, we summarize some results of time dependent priority (TDP) scheduling. We first characterize a necessary and sufficient condition for a given delay spacing to be feasible under TDP for two traffic classes. We then extend the characterization of TDP to N classes. We also present an iterative method for obtaining the values of the feasible control parameters.

In general, TDP is a *non-preemptive* packet scheduling algorithm which provides a set of control variables $b_i, 1 \leq i \leq N$ where $0 \leq b_1 \leq b_2 \leq \cdots \leq b_N$. The control variable b_i dictates the dynamic instantaneous priority of class i packets. Specifically, if the k-th packet of the class i arrives at the queue at time τ_k, then its priority at time t (for $t \geq \tau_k$), denoted by $q_i^k(t)$, is

$$q_i^k(t) \quad = \quad (t - \tau_k)b_i \; . \tag{1}$$

Figure 1 illustrates a two-class TDP. Assume that the first packet of class 1 arrives at time 0 and the first packet of class 2 arrives at time t_1, and both packets remain in the system until time t_3. During the time interval $(t_1, t_2]$, the class 1 packet will have a higher priority than the class 2 packet. But since the control parameter b_2 is larger than b_1, after time $t > t_2$, the class 2 packet will have a higher priority. Let $N_i(t)$ denote the number of class i packets

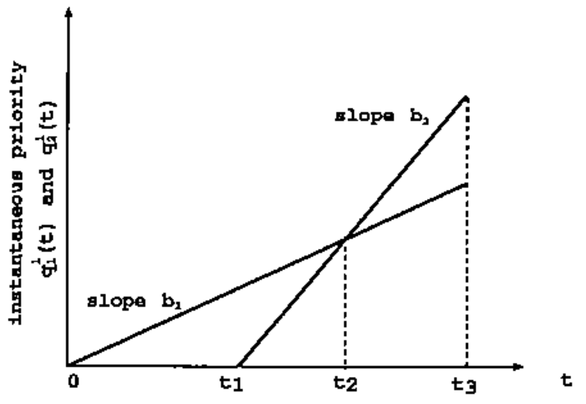


Figure 1: A two-class TDP where $b_1 < b_2$.

waiting in the queue at time t. If the server is ready to transmit a packet at time t, it only needs to consider the priority of the packets at the head of the queue *for each class*. It is because for the same class, the earlier arrival packets always have a higher priority than the later arrival packets. Let us define $q_i(t)$ as the priority of the packet at the head of the class i queue. When the server is ready to transmit a packet, it chooses a packet from class i^* where

$$i^*(t) \quad = \quad arg \max_{i=1..N, N_i(t)>0} \{q_i(t)\} \tag{2}$$

Ties for the highest priority are broken by serving the packet that has been waiting the longest in the system. If there is no packet in the system, then the server will be idle and it will be activated by a newly arriving packet. Note that in the TDP scheduler, a class i packet increases in priority at a faster rate (b_i) than packets of any class $j < i$.

We can obtain the expected long term waiting time of each class of traffic under the TDP scheduling algorithm. Assuming that the arrival process of class i traffic is Poisson with an average rate of λ_i, and the services times of class i packets have a general distribution with the first and second moments given by $\overline{x_i}$ and $\overline{x_i^2}$, respectively, the system utilization, denoted by ρ, of a TDP server is equal to $\sum_{i=1}^{N} \rho_i$ where $\rho_i = \lambda_i \overline{x_i}$. In [8], the author derives a closed-form expression for the average long-term waiting time for class i packets. The closed-form expression is given as

$$W_i = \frac{[W_0/(1-\rho)] - \sum_{k=1}^{i-1} \rho_k W_k [1 - (b_k/b_i)]}{1 - \sum_{k=i+1}^{N} \rho_k [1 - (b_i/b_k)]} \quad i = 1, .., N \tag{3}$$

where $W_0 = \frac{1}{2} \sum_{i=1}^{N} \lambda_i \overline{x_i^2}$ is the expected residual service time. It is interesting to note that the above expression was derived by assuming that packet service times are exponentially distributed. In [14], the authors illustrate that the closed-form expression in Equation (3) is also valid for any general service time distribution.

One attractive feature about the TDP scheduler is that if one wants to maintain certain proportional differentiation of waiting times between different classes of traffic, one can simply *adjust* the control parameters b_i's so as to achieve the desired waiting time spacings. Let $r_{i,j}^t$ be the *target* long term average waiting time ratio between class i and class j traffic, and $r_{i,j}^a$ be the *achieved* long term average waiting time ratio between class i and class j traffic (i.e., $r_{i,j}^a = \frac{W_i}{W_j}$). The goal of proportional-delay differentiated services is to make the achieved long term waiting time ratio equal to the target long term waiting time ratio; i.e., to achieve $r_{i,j}^a = r_{i,j}^t$. In [2], the authors study the proportional average delay behavior of the TDP scheduler and show that when the system utilization tends to 100%, $r_{i,j}^a = r_{i,j}^t$ can be achieved by setting the control parameters equal to the inverse of the target ratio. However, the authors only consider the asymptotic case of 100% utilization. In practice, a router is not under 100% loading at all times. Thus, we investigate other loadings as well. In this paper, we address the following important questions:

1. Given the waiting time ratio requirements for all classes

$r^t_{i,i+1}$ (where $i = 1, 2, \ldots, N-1$), under what conditions of ρ_i's does a solution for b_i's exist?

2. Given ρ_i, the traffic loads of all classes, how to obtain the b_i values so that the achieved waiting time ratios $r^a_{i,i+1}$ are equal to the target ratios $r^t_{i,i+1}$?

To understand the problem, we start with a simple case of two traffic classes. We then go on to solve the general problem of N traffic classes.

3.1 Two-class Proportional DS

Theorem 1 *For two classes of traffic, let $r^t_{1,2}$ be the target ratio of the average waiting time of class 1 traffic to that of class 2 traffic. Then, $r^a_{1,2} = r^t_{1,2}$ is feasible if and only if the system utilization ρ satisfies:*

$$1 - \frac{1}{r^t_{1,2}} < \rho < 1$$

Proof: First, $\rho < 1$ is required so that the system is stable. Let us first show the *only if part* (i.e., if $r^a_{1,2} = r^t_{1,2}$, then $1 - \frac{1}{r^t_{1,2}} < \rho < 1$). According to Equation (3), packets of the two classes have average waiting times of

$$\begin{aligned} W_1 &= \frac{[W_0/(1-\rho)]}{1-\rho_2[1-(b_1/b_2)]} \\ W_2 &= [W_0/(1-\rho)] - \rho_1 W_1[1-(b_1/b_2)]. \end{aligned}$$

By substituting W_1 into W_2, we have

$$W_2 = [W_0/(1-\rho)]\frac{1-\rho[1-(b_1/b_2)]}{1-\rho_2[1-(b_1/b_2)]}.$$

The achieved ratio of the long term average waiting time between class one and class two is given as

$$r^a_{1,2} = \frac{W_1}{W_2} = \frac{1}{1-\rho[1-(b_1/b_2)]}. \quad (4)$$

If the target ratio is achieved, that is, $r^a_{1,2} = r^t_{1,2}$, then

$$r^t_{1,2} = \frac{1}{1-\rho[1-(b_1/b_2)]}.$$

After rearranging the above equation, we have

$$\rho = \frac{b_2}{b_2 - b_1}\left(1 - \frac{1}{r^t_{1,2}}\right) \quad (5)$$

Since $0 < b_1 < b_2$, this implies that $\frac{b_2}{b_2-b_1} > 1$. Therefore, $\rho > 1 - \frac{1}{r^t_{1,2}}$

Next, we consider the *if part* (i.e., if $1 - \frac{1}{r^t_{1,2}} < \rho < 1$, then $r^a_{1,2} = r^t_{1,2}$). If $\rho > 1 - \frac{1}{r^t_{1,2}}$, then we can let

$$\rho = \frac{b_2}{b_2 - b_1}\left(1 - \frac{1}{r^t_{1,2}}\right) \quad (6)$$

where b_1 and b_2 are some constants such that $0 < b_1 < b_2$. By substituting it into Equation (3), we get $r^a_{1,2} = r^t_{1,2}$. ■

Remarks: The implication of the above theorem is that in order to achieve the target ratio $r^t_{1,2}$, we need to have a *sufficient* amount of traffic and *enough* packets which are backlogged in the system. For example, if the requirement is $r^t_{1,2} = 10$, then the system has to be *at least* 90% utilized so as to achieve the desired waiting time spacing. In other words, if the system utilization is less than 90%, then we cannot achieve $r^a_{1,2} = 10$.

We make two observations from the above theorem. First, the ratio of the average waiting times does not *solely* depend on b_2/b_1, but rather, depends on the system utilization also. Only when the utilization tends to one ($\rho \to 1$) will the control parameters $b_2/b_1 = r^t_{1,2}$ achieve the desired waiting time spacing. Second, if the system utilization is known, we can adjust b_1 and b_2 such that the achieved waiting time ratio will be equal to our target value $r^t_{1,2}$. Let us present how to choose the proper values for b_i's.

Corollary 1 *If $b_1 = 1$, $b_2 = \rho/(\rho - 1 + \frac{1}{r^t_{1,2}})$, and $1 - \frac{1}{r^t_{1,2}} < \rho < 1$, then $r^a_{1,2} = r^t_{1,2}$.*

Proof: By substituting $b_1 = 1$, $b_2 = \rho/(\rho - 1 + \frac{1}{r^t_{1,2}})$ into Equation (3), we can achieve $r^a_{1,2} = r^t_{1,2}$. ■

Corollary 2 *If $1 - \frac{1}{r^t_{1,2}} < \rho < 1$, then any b_1 and b_2 such that $b_1/b_2 = (\rho - 1 + \frac{1}{r^t_{1,2}})/\rho$ can achieve $r^a_{1,2} = r^t_{1,2}$.*

Proof: First, we need some results for the TDP system. In [8], the author states that for two TDP systems $\mathcal{A}$ and $\mathcal{B}$ wherein the control parameters for system $\mathcal{A}$ are $\{b_i\}$ and the control parameters for system $\mathcal{B}$ are $\{b_i'\}$. If we maintain the following relationship

$$\frac{b_i}{b_{i+1}} = \frac{b_i'}{b_{i+1}'} \qquad \text{for } i = 1, 2, \ldots, N,$$

then W_i in $\mathcal{A}$ will be equal to W_i' in $\mathcal{B}$. In other words, the average waiting time of a TDP system depends not on the *exact* value of the control parameters b_i's but rather, depends on the *ratios* of b_i's.

If $1 - \frac{1}{r^t_{1,2}} < \rho < 1$, from Corollary 1, $b_1 = 1$, $b_2 = \rho/(\rho - 1 + \frac{1}{r^t_{1,2}})$ can achieve $r^a_{1,2} = r^t_{1,2}$. As TDP system depends on the ratios of b_i's only, any b_1' and b_2' such that

$$\begin{aligned} \frac{b_1'}{b_2'} &= \frac{1}{\rho/(\rho - 1 + \frac{1}{r^t_{1,2}})} \\ &= \frac{\rho - 1 + \frac{1}{r^t_{1,2}}}{\rho} \end{aligned}$$

can achieve $r^a_{1,2} = r^t_{1,2}$. ■

In conclusion, to satisfy a specified system performance requirement $r^t_{1,2}$, we need to measure the system utilization and set the parameters b_1 and b_2 accordingly. The traffic loading measurement and dynamic adjustment of the control parameters $\{b_i\}$'s will be discussed in detail in Section 4. Then the achieved long term average waiting time ratio of class one traffic to class two traffic will be equal to the target value of $r^t_{1,2}$, provided that ρ is within the feasibility region $(1 - 1/r^t_{1,2}, 1]$.

3.2 N-class Proportional DS

For the general case of N classes, the problem becomes very complicated because to find the values of the control parameters $\{b_i\}$'s, we need to solve Equation (3), which is a system of N non-linear equations. Nevertheless, we can calculate W_i by using the ***conservation law principle***, provided that the configuration of the system (ρ_i and W_0) is known. The conservation law [8] states that if a scheduling discipline is independent of the service time of jobs, then the weighted average of the waiting times of all classes is ***invariant***, and it is equal to the average waiting time of an M/G/1 system. Mathematically, the relationship is

$$\sum_{i=1}^{N} \frac{\rho_i}{\rho} W_i \quad = \quad \frac{W_0}{1-\rho}. \tag{7}$$

Let us define $s_i = r^t_{i,i+1} r^t_{i+1,i+2} \cdots r^t_{N-1,N}$. If we can achieve $r^a_{i,j} = r^t_{i,j}$, then $s_i = W_i/W_N$, so we can express all W_i's in terms of W_N and s_i. That is $W_i = s_i W_N$ for all $i = 1, 2, \ldots, N-1$. Substituting this expression of W_i in Equation (7), we have

$$\frac{W_0}{1-\rho} = \frac{1}{\rho} \sum_{i=1}^{N} \rho_i s_i W_N.$$

We can express W_i in terms of s_i, ρ_i, and W_0, which are:

$$W_i \quad = \quad s_i \frac{\rho W_0}{1-\rho} \left(\sum_{i=1}^{N} \rho_i s_i \right)^{-1} \quad \text{for } i = 1, 2 \ldots, N. \tag{8}$$

From the above equations, we observe that if $r^a_{i,j} = r^t_{i,j}$ is achieved, the only unknown in Equation (3) is the vector $\boldsymbol{b} = [b_1, b_2, \cdots, b_N]$. Now, putting all b_i's in Equation (3) on the left hand side, we have

$$b_i \left(W_i \sum_{k=i+1}^{N} \frac{\rho_k}{b_k} \right) - \frac{1}{b_i} \left(\sum_{k=1}^{i-1} \rho_k W_k b_k \right) = \frac{W_0}{1-\rho} - \sum_{k=1}^{i-1} \rho_k W_k - \left(1 - \sum_{k=i+1}^{N} \rho_k \right) W_i. \tag{9}$$

Letting

$$A(i) \quad = \quad W_i \sum_{k=i+1}^{N} \frac{\rho_k}{b_k}; \quad B(i) = \sum_{k=1}^{i-1} \rho_k W_k b_k;$$
$$R(i) \quad = \quad \frac{W_0}{1-\rho} - \sum_{k=1}^{i-1} \rho_k W_k - \left(1 - \sum_{k=i+1}^{N} \rho_k \right) W_i,$$

we have

$$A(i)\, b_i - \frac{B(i)}{b_i} \quad = \quad R(i) \qquad i = 1, 2, \ldots, N. \tag{10}$$

Now, we have a system of non-linear equations for solving the b_i's. Since all the b_i's have to be positive, there should be a condition for ρ_i and s_i such that $\{b_i\}$'s are positive. The result is expressed in the following theorem.

Theorem 2 *A necessary condition to have positive solutions of the b_i's is $R(1) > 0$ and $R(N) < 0$.*

Proof: Since $b_i > 0$ for $i = 1, 2, \ldots, N$, we have $A(i) > 0$ and $B(i) > 0$. However, $R(i)$ can be positive or negative. Let us consider three cases.
Case 1: For $i = 1$, we have $B(1) = 0$, which implies that

$$b_i = \frac{R(1)}{A(1)}.$$

Since $b_1 > 0$, this in turn implies that $R(1) > 0$.
Case 2: For $1 < i < N$, we use the result from Equation (10) and get

$$A(i)\, b_i^2 - R(i)\, b_i - B(i) = 0.$$

Since we want $\{b_i\}$'s to be positive, we have

$$b_i = \frac{R(i) + \sqrt{R(i)^2 + 4A(i)B(i)}}{2A(i)}.$$

Because $R(i)^2 + 4A(i)B(i) > R(i)^2$, therefore $|R(i)| < \sqrt{R(i)^2 + 4A(i)B(i)}$. Hence we have

$$b_i > \frac{R(i) + |R(i)|}{2A(i)} \geq 0.$$

In summary, for $1 < i < N$, b_i is always greater than zero even when $R(i)$ is negative.
Case 3: For $i = N$, we have $A(N) = 0$, which implies that

$$b_N = -\frac{B(N)}{R(N)}$$

Since $b_N > 0$ and $B(N) > 0$, we conclude $R(N) < 0$. ■

Remarks: The implication of the above theorem is that a necessary condition for a feasible region (e.g., a region wherein a positive solution of b_i's exist) is $R(1) > 0$ and $R(N) < 0$. If the system configuration (ρ_i, s_i) falls outside of this region, it is possible that there exist no positive values of the b_i's for which the TDP scheduler can obtain the target waiting time ratios.
The first condition $R(1) > 0$ implies

$$\frac{W_0/(1-\rho)}{W_1} \quad > \quad 1 - \sum_{i=2}^{N} \rho_i \tag{11}$$

where $W_0/(1-\rho)$ is the average waiting time of the aggregate traffic. If we want a *large* waiting time differentiation, W_1 has to be larger than $W_i, i = 2, \ldots, N$. Since $W_1 \geq W_2 \geq \cdots \geq W_N$, this implies the fraction on the left hand side of Equation (11) has to be small. Thus, $\sum_{i=2}^{N} \rho_i$ should be close to one to make the inequality hold. The *physical meaning* is that to have a large waiting time differentiation, there should be a *sufficient* amount of higher

class packets to keep the system busy so that the lower class packets are delayed adequately.
The second condition is $R(N) < 0$, which implies

$$\frac{W_0}{1-\rho} \quad < \quad \sum_{i=1}^{N-1} \rho_i W_i + W_N. \tag{12}$$

By the conservation law, $W_0/(1-\rho) = \sum_{i=1}^{N} \frac{\rho_i}{\rho} W_i$. If we put it back into Equation (12), we have

$$\begin{aligned} \sum_{i=1}^{N} \frac{\rho_i}{\rho} W_i \quad &< \quad \sum_{i=1}^{N-1} \rho_i W_i + W_N \\ 0 \quad &< \quad \sum_{i=1}^{N-1} \rho_i \left(W_N - W_i(1-\rho)\right). \end{aligned} \tag{13}$$

Since $W_1 \geq W_2 \geq \cdots \geq W_N$, to make the right hand side of Equation (13) positive, one way is for ρ to be large (i.e., tend to 1). If ρ tends to 1, the value of the left hand side in Equation (12) will be large. To make the inequality hold, the value of the right hand side in Equation (12) should be larger. Since $W_1 \geq W_2 \geq \cdots \geq W_N$ and the major part of $\sum_{i=1}^{N-1} \rho_i W_i + W_N$ is the weighted average of the mean waiting time of the first $N-1$ classes, to attain a large value, ρ_i should be large, especially for the lower traffic classes. The *physical meaning* is that in order to have a large waiting time differentiation, the server has to delay packets of the lower classes so as to have large waiting times $W_i, i = 1, \ldots, N-1$. If their traffic loading is high, many of them will be backlogged and their waiting time will increase. Last but not least, another important implication of the above necessary conditions is that even though the system utilization ρ remains unchanged, it is still possible that certain distributions of ρ_i's will not lead to a positive solution of b_i's. In such cases, the system cannot achieve the target waiting time ratios.

We now present an efficient algorithm for computing the values of b_i's, provided that the necessary condition is satisfied. In general, we have to find a solution for the set of non-linear equations in Equation (9). To achieve this, the following iterative algorithm is proposed. The iterative algorithm is based on the Gauss-Seidel iteration method [9], which has a well-known condition for convergence.

First, let ϕ_i be the *functional evaluation operator* for b_p where $b_i = \phi_i(b_1, b_2, \ldots, b_N)$ for $i = 1, 2, \ldots, N$ where

$$\phi_i(b_1, b_2, \ldots, b_N) \quad = \quad \begin{cases} \frac{R(i)+B(i)/b_i}{A(i)} & \text{for } i \neq N, \\ \sqrt{\frac{b_i B(i)}{-R(N)}} & \text{for } i = N, \end{cases} \tag{14}$$

and for $i = 1, \ldots, N$, we have:

$$f_i(b) \quad = \quad A(i)\, b_i - B(i)/b_i - R(i) \tag{15}$$

The iterative algorithm is:

Procedure: Iterative Algorithm

```
Input: $\lambda_i$, $\overline{x_i}$, $\overline{x_i^2}$ for $i = 1, .., N$.
/*average arrival rate of class $i$ traffic,
      $1^{st}$, $2^{nd}$ moments of service times */
Output: $b = [b_1, b_2, \ldots, b_N]$.

1.  begin
2.    $k = 0$; $b_i^{(0)} = \frac{1}{W_i}$ for $i = 1, \ldots, N$; /* initialize */
3.    /* test for convergence */
4.    while (($\sum_{i=1}^{N} |f_i(b^{(k)}| > \epsilon$) and
                 $k <$ MAX_ITERATION_COUNT )
5.      begin /* update the value of $b_i^{(k)}$ */
6.        for ($i = 1; i <= N; i = i + 1$)
7.          $b_i^{(k+1)} = \phi_i(b_1^{(k+1)}, b_2^{(k+1)}, .., b_{i-1}^{(k+1)}, b_i^{(k)}, .., b_N^{(k)})$;
8.        $k = k + 1$;
9.      end
10. end
```

In line 2, we initialize the starting point of the iterative algorithm. The functional evaluation operator $f_i(\cdot)$ in line 4 is the set of Equations (15). That is, for the k^{th} iteration control parameters $b_i^{(k)}$, we test whether these control parameters can satisfy Equation (15) or not. If the absolute aggregated error is less than a pre-defined error threshold, then we obtain the correct control parameters. For completeness, we show the convergence condition of our algorithm in [11].

4 Dynamic Adjustment & Related Issues

From Corollary 1 and Theorem 2, it is clear that the values of the control parameters to support a given quality spacing are a function of traffic loadings. These traffic loadings are not constant in a realistic system. To maintain a given quality spacing, it is thus necessary to monitor changes in system conditions and to adapt the control parameters accordingly. (Other researchers have also looked at adapting network control parameters based on past measurements of network utilization, in different contexts than ours. For example, adapting admission control criteria for predictive service using past measurements of network traffic is discussed in [7]; adapting the ECN marking probabilities of packets – for router congestion control – based on the total traffic arrival rate at a router is proposed in [18].) In this section, we present efficient dynamic measurement algorithms for tracking the loads of different traffic classes.

Assuming that the traffic conditions evolve slowly in practice, we can predict present traffic arrival rates by a history of past arrival rates. Specifically, we monitor the number of packet arrivals over definite time windows. To estimate the present arrival rate of a flow given these past samples, we experimented with two possible strategies:

- **Jumping window.** We use a jumping window of size W (in seconds). At the end of each window, the number of packet arrivals for a flow during the window divided by W gives a new arrival rate estimate for the flow. Estimates for all flows are fed to the iterative algorithm to give new control parameters, b_i's, for use in the current window. The window size W provides controlled tradeoff between system stability and responsiveness. This is because a larger W incorporates more history into the estimation, giving estimates that are more robust against transient conditions. A smaller W, on the other hand, allows quicker adaptations based on more instantaneous measures of system behavior. The jumping window algorithm is specified in Figure 2.

- **Exponential averaging.** Instead of using a jumping window of size W, this algorithm is based on the exponential averaging technique proposed by Jacobson and Karels [6] for estimating TCP round trip times, which includes second-order statistics for added robustness. In the algorithm specification in Figure 3, `delta`, `sigma` and `zeta` are input parameters. Of these, `delta` and `sigma` (between 0 and 1) control, independently of the measurement window, how much the new estimate for the arrival rate and its variance, respectively, is weighted by previous history. As the parameters become smaller, more history is admitted. This contributes to system stability at the expense of responsiveness.

```
Initially:
  arrival_rate_i = 0.0;    /* estimate for flow i traffic */

At the end of each measurement window:
  counter_i = # of class i pkt arrivals within the window;
  W = window size (in seconds);

  arrival_rate_i := counter_i / W;
```

Figure 2: jumping window algorithm

In each strategy, we set initial control parameters such that their ratios are inverses of the corresponding target spacing ratios; i.e., $b_i/b_j = 1/r_{i,j}, \forall i,j$. Adaptations are then made periodically, with period W. Besides giving a more eagerly reacting system, a smaller W clearly adds to the operation cost of a router, in that the iterative algorithm has to run with higher frequency. We also consider a very efficient baseline approach in which after the initial control parameters are determined, they remain unchanged thereafter (i.e., W is ∞ and there is no adaptation). We call it the *static control* approach.

In Section 5, we evaluate the performance of our adaptation strategies, when compared with the use of static control parameters. For dynamic operating environments, such as described by an MMPP with significantly varying flow arrival rates between states, we show that adaptation can be generally beneficial and allow target quality spacings to be maintained. Further issues regarding the stability, robustness, and responsiveness of the jumping window and exponential averaging algorithms are also discussed.

```
Note that delta, sigma and zeta are input parameters. We set
  delta = 1/8.
  sigma = 1/4.
  zeta = 2.0;

Initially:
  arrival_rate_i = 0.0;
  diff = 0.0;
  dev  = 0.0;

At the end of each measurement window:

  counter_i = # of class i pkt arrivals within the window;
  W = window size (in seconds);
  /* to calculate */
  diff = counter_i / W - arrival_rate_i;
  n_arrival_rate = (1.0 - delta) * arrival_rate_i +
                         delta * counter_i / W;
  n_dev = (1.0 - sigma) * dev + sigma * |diff|;
  /* set new estimate of the deviation of arrival rate */
  dev = n_dev;

  /* compute new estimate of arrival rate. This will */
  /* be used as input to find the control vector b.  */

  arrival_rate_i = max(n_arrival_rate + zeta * dev, 0);
```

Figure 3: exponential averaging algorithm

5 Experimental Results

In this section, we present the results of our experiments (further experiments are also be found in [11]). We classify the experiments into four types, A, B, C, and D, for which the goals are:

Type A (comparisons with [2]): Illustrate the effectiveness of our iterative algorithm, as compared to the results in [2], in finding the values of the WTP control parameters. We compare both long term and short term achieved waiting time ratios between different classes of traffic.

Type B (non-Poisson traffic): Illustrate the *insensitivity* of our iterative algorithm for non-Poisson traffic. To do so, we study achievable waiting time spacings when the input traffic is Pareto, MMPP, or mixed (i.e., combination of Poisson, MMPP and Pareto traffic).

Type C (dynamic adjustment algorithms): Illustrate the effectiveness of our dynamic adjustment algorithms. We study the quality of the achieved waiting time spacings under varying traffic intensities and see how well our dynamic algorithms can adjust the WTP control parameters under changing operating conditions. We study both long and short term waiting time ratios achieved under heterogeneous input traffic models (i.e., Pareto, MMPP and Poisson).

Type D (end-to-end accumulative waiting time): Illustrate the achieved end-to-end accumulative waiting times for different traffic classes, when the traffic traverses multiple network nodes. We investigate the achieved end-to-end accumulative waiting times under different traffic scenarios, such as different router loads and cross traffic patterns.

5.1 Type A: Comparisons with [2]

In this subsection, we report results from several experiments. In the first experiment, we compare performance results using the control parameters taken from [2] versus control parameters obtained using our iterative algorithm. We investigate long-term and short-term average waiting time spacings under various system utilizations.

Experiment A.1 (Comparisons with [2]): We consider three classes of traffic. The arrival process of class i ($i = 1, 2, 3$) is Poisson with a rate of λ_i. The packet length distribution is the same for all classes where 40% of the packets are 40 bytes, 50% are 550 bytes, and 10% are 1500 bytes. The output link capacity is 441 bytes/unit time, where the time unit can be normalized to achieve an arbitrary link speed. In each run of the experiment, we generate at least 50,000 packet arrivals for each class. Then, we average the waiting times for each class and compare the achieved waiting time ratios with the target ratios. In part one of the experiment, we set $\lambda_1 = 0.35, \lambda_2 = 0.3, \lambda_3 = 0.3$ (since the service time requirement is normalized to one, the system utilization is $\rho = 0.95$) and consider the target waiting time spacing of $r^t_{i,i+1} = 4.0$. Table 1 illustrates the achievable spacings, using the control parameters in [2] and our proposed method. We observe that the

Method used	control parameters	$r^a_{1,2}$	$r^a_{2,3}$
[2]	$b_1 = 1, b_2 = 4, b_3 = 16$	3.366	3.030
our approach	$b_1 = 1, b_2 = 5.11$ $b_3 = 35.937$	3.990	3.890

Table 1: Comparison between achievable waiting time spacing (where $r^t_{i,i+1} = 4.0$ and $\lambda_1 = 0.35, \lambda_2 = \lambda_3 = 0.3$).

Method used	control parameters	$r^a_{1,2}$	$r^a_{2,3}$
[2]	$b = [1, 2, 4]$	1.39	1.36
our approach	can't pass feasibility test	—	—

Table 2: Determination of non-achievable waiting time spacing (where $r^t_{i,i+1} = 2.0$ and $\lambda_1 = \lambda_2 = \lambda_3 = 0.2$).

proposed control parameters in [2] cannot achieve the target spacings even at high system loading. However, our proposed algorithm can find the appropriate values of the control parameters such that the target waiting time ratios can be achieved. In the second part of the experiment, we set the arrival rates as $\lambda_1 = 0.2, \lambda_2 = 0.2, \lambda_3 = 0.2$ (or $\rho = 0.6$) and $r^t_{i,i+1} = 2.0$. Table 2 illustrates the achievable spacings. As shown, our algorithm can determine that it is ***not possible*** to achieve the target waiting time spacings ($r^t_{i,i+1} = 2$) for the given load distribution. Indeed, using the proposed control parameter values in [2], we can only achieve spacing values around 1.3. These experiments illustrate that our iterative algorithm can determine whether it is feasible to achieve given waiting time spacings, and if so, the correct values of the control parameters.

Experiment A.2 (Long-term waiting time spacing): In the first part of the experiment, we want to test whether we can achieve the target waiting time ratios under different system utilizations. We consider three classes of traffic. All arrival processes are Poisson. For a low system utilization case ($\rho = 0.2$), the arrival rates are $\lambda_1 = 0.05, \lambda_2 = 0.1, \lambda_3 = 0.05$. For a medium utilization case ($\rho = 0.6$), the arrival rates are $\lambda_1 = 0.2, \lambda_2 = 0.2, \lambda_3 = 0.2$. For a high utilization case ($\rho = 0.9$), the arrival rates are $\lambda_1 = 0.3, \lambda_2 = 0.3, \lambda_3 = 0.3$. The packet length distribution is similar to the one in Experiment A.1. The experimental results are summarized in Tables 3 to 4. As we can observe, our iterative algorithm is very efficient (less than 20 iterations of the algorithm are needed) in finding the correct control parameter values.

ρ	$\rho = 0.2$	$\rho = 0.6$	$\rho = 0.9$
$r^t_{1,2}$	1.1	1.1	1.1
$r^t_{2,3}$	1.1	1.1	1.1
$[b_1, b_2, b_3]$	$[1, 2.03, 4.17]$	$[1, 1.18, 1.4]$	$1, 1.11, 1.24]$
# of loops	5	11	12
$r^a_{1,2}$	1.10	1.09	1.10
$r^a_{2,3}$	1.12	1.09	1.10

Table 3: Long-term average waiting time spacings with $r^t_{i,i+1} = 1.1$ under different system utilizations.

ρ	$\rho = 0.2$	$\rho = 0.6$	$\rho = 0.9$
$r^t_{1,2}$	2.0	2.0	2.0
$r^t_{2,3}$	2.0	2.0	2.0
$[b_1, b_2, b_3]$	outside feasible region	outside feasible region	$[1, 2.32, 5.554]$
# of loops	—	—	20
$r^a_{1,2}$	—	—	2.01
$r^a_{2,3}$	—	—	1.99

Table 4: Long-term average waiting time spacings with $r^t_{i,i+1} = 2.0$ under different system utilizations.

In the second part of the experiment, we vary the number of traffic classes, and see how robust our algorithm is in finding the appropriate control parameter values. In experiment A, we consider four classes, whose traffic arrival rates are $\boldsymbol{\lambda} = 0.2, 0.2, 0.25, 0.25$, respectively. In experiment B, we consider five classes, whose traffic arrival rates are $\boldsymbol{\lambda} = 0.1, 0.1, 0.1, 0.3, 0.3$, respectively. In experiment C, we consider six classes, whose traffic arrival rates are $\boldsymbol{\lambda} = 0.1, 0.1, 0.1, 0.1, 0.25, 0.25$, respectively. In experiment D, we consider seven classes, whose traffic arrival rates are $\boldsymbol{\lambda} = 0.1, 0.05, 0.05, 0.05, 0.05, 0.3, 0.3$, respectively. The results are shown in Table 5. In all the cases, our iterative algorithm is highly efficient (requiring between 12 and 39 iterations) in obtaining the control parameter values for given target waiting time spacings.

Lastly, we evaluate system performance under different class load distributions. We consider three classes of traffic with target spacings of $r^t_{i,i+1} = 1.1$. In all the cases considered, the system utilization is $\rho = 0.9$. The results

Exp.	# of loops	control parameters (b)	achieved spacing ($r^a_{i,i+1}$)
A	38	[1.0, 1.113, 1.239, 1.380]	$r^a_{1,2} = 1.11$, $r^a_{2,3} = 1.09$ $r^a_{3,4} = 1.11$.
B	12	[1.0, 1.115, 1.242, 1.383, 1.539]	$r^a_{1,2} = 1.10$, $r^a_{2,3} = 1.10$ $r^a_{3,4} = 1.10$, $r^a_{4,5} = 1.10$
C	39	[1.0, 1.116, 1.244, 1.385, 1.543, 1.719]	$r^a_{1,2} = 1.10$, $r^a_{2,3} = 1.09$ $r^a_{3,4} = 1.11$, $r^a_{4,5} = 1.10$ $r^a_{5,6} = 1.10$
D	13	[1.0, 1.117, 1.247, 1.391, 1.551, 1.728, 1.926]	$r^a_{1,2} = 1.10$, $r^a_{2,3} = 1.10$ $r^a_{3,4} = 1.10$, $r^a_{4,5} = 1.10$ $r^a_{5,6} = 1.10$, $r^a_{6,7} = 1.10$

Table 5: Long-term waiting time spacings as we vary the number of traffic classes ($r^t_{i,i+1} = 1.1$).

are shown in Table 6. Our algorithm achieves waiting time spacings remarkably close to 1.1 across the different traffic distributions. From the experiments in this subsection, we conclude that our proposed algorithm can robustly determine accurate control parameter values under a variety of operating conditions (i.e., different system utilizations, different numbers of traffic classes, and different traffic load distributions).

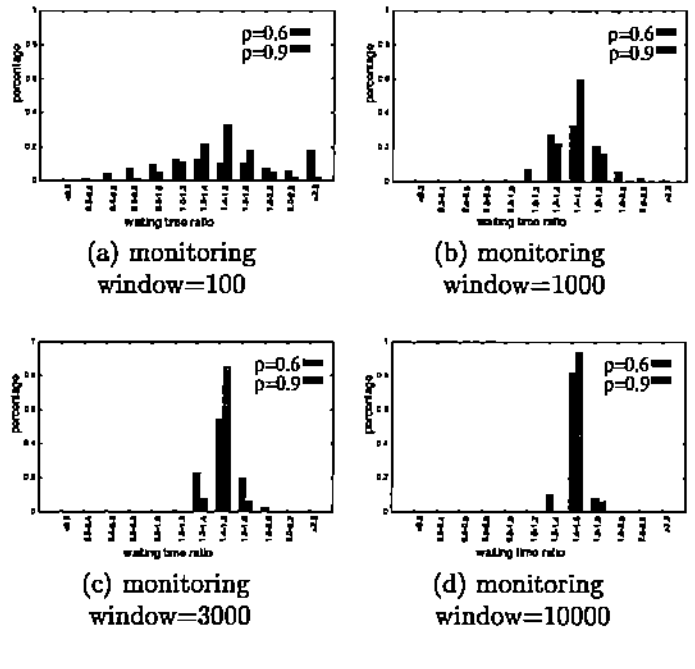
load distribution (%)	# of loops	b	$r^a_{1,2}$	$r^a_{2,3}$
33.3-33.3-33.3	11	[1, 1.113, 1.238]	1.10	1.10
30-20-50	9	[1, 1.113, 1.238]	1.10	1.10
20-30-50	6	[1, 1.113, 1.238]	1.10	1.10
10-45-45	4	[1, 1.113, 1.238]	1.10	1.11
45-10-45	51	[1, 1.113, 1.239]	1.10	1.10
45-45-10	36	[1, 1.113, 1.238]	1.10	1.10

Table 6: Waiting time spacings for three classes of traffic under different traffic loading distributions (target $r^t_{i,i+1} = 1.1$ and $\rho = 0.9$).

Experiment A.3 (Short-term waiting time spacing): Besides long-term analysis, we also study the short-term behavior of our packet scheduling algorithm. In these experiments, we want to find out the ratios of the average waiting times between successive classes within a *fixed time interval* (or what we call the *monitoring window*). We measure the average waiting times of all the packets that get served within a monitoring window. The length of the monitoring window is varied to be 100, 1000, 3000 and 10,000 p-units, where a p-unit is the average packet transmission (or service) time. Figures 4 is the histogram for the achieved short-term waiting time ratios under different system utilizations, target waiting time spacings and monitoring window sizes. The x-axis in a histogram shows the whole range of possible waiting time ratios (the end partitions are less than 0.2 and greater than 2.2, respectively). As an example, Figure 4(d) is the histogram for the waiting time spacings under $\rho = 0.6$ and 0.9, target ratios $r^t_{i,i+1} = 1.1$ and monitoring window size of 10,000 p-units. From Figures 4, we make two observations:

1. The short-term waiting time ratios approach asymptotically the target waiting time ratios as we increase the monitoring window size, i.e., the larger the monitoring window size, the higher the chance of achieving the target waiting time ratios over a short time scale. As the length of the monitoring window size increases to infinity (which corresponds to the steady state waiting time ratios), the target waiting time ratios $r^t_{i,i+1}$ can be exactly achieved.
2. the higher the system utilization, the short-term waiting time ratios approach the target waiting time ratios $r^t_{i,i+1}$ at a faster rate than with lower system utilization.

Note that if the system is highly utilized, then the achievable waiting time ratios can be kept close to our target spacings even in *short* timescales. As can be observed from these figures, most of the waiting time ratios fall within our target spacings. Moreover, the variance of the ratios is small. However, if the system utilization is low, the variance of the waiting time spacings is large. For really short time scales (e.g., monitoring window size of 100 p-units), only a small percentage of the data points lies within our target region. The reason is that when the system utilization is low, the scheduler needs a longer time to serve sufficient packets so that the per-class waiting times can reach the equilibrium values. This makes the target waiting time ratios difficult to achieve over a short monitoring window. When the system utilization is high, sufficient packets can arrive, even within a small monitoring window, so that the waiting times of traffic within the monitoring window will be close to the equilibrium values.



(a) monitoring window=100

(b) monitoring window=1000

(c) monitoring window=3000

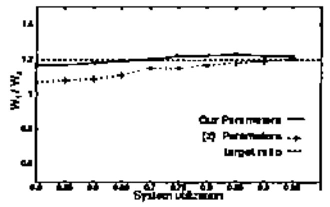
(d) monitoring window=10000

Figure 4: Achieved short-term waiting time ratios, target ratio $r^t_{i,i+1} = 1.5$, system utilization $\rho = 0.6$ or $\rho = 0.9$

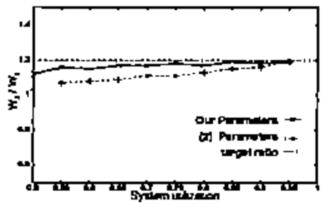
5.2 Type B: Non-Poisson Traffic

Realistic traffic may not be Poisson, and a robust packet scheduler should not depend on modeling assumptions. In [3], the authors study a scheduling approach in which the delay ratios between service classes are dynamically monitored to affect the scheduling decision. However, the approach can cause large delay ratio deviations in short time scales. Although the WTP scheduler does not depend on modeling assumptions, its control parameters are computed based on M/G/1 queuing analysis. Therefore, in this subsection, we consider non-Poisson input traffic. We evaluate the effectiveness of our iterative algorithm in finding WTP control parameter values to meet target waiting time ratios. For all the experiments in this subsection, we consider three classes of traffic. The load is evenly distributed among the three classes. The packet length distribution is the same for all the classes, where 40% of the packets are 40 bytes, 50% are 550 bytes, and 10% are 1500 bytes. The output link capacity is 441 bytes/unit time, where the time unit can be normalized to achieve an arbitrary link speed.

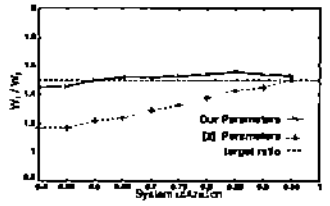
Experiment B.1 (Pareto traffic): We consider three Pareto traffic arrival processes. The packet inter-arrival times for each class are Pareto distributed with the shape parameter equal to 1.9. We vary the system utilization from 0.5 to 0.95 and compare the achieved long term waiting time ratios between our iterative algorithm and the method in [2]. We consider two target waiting time spacings, $r^t_{i,i+1} = 1.2$ and $r^t_{i,i+1} = 1.5$. Figure 5 presents the comparisons. From these figures, we observe that using our iterative algorithm: (1) the long term achieved waiting time ratios are significantly *closer* to the target waiting time ratios, as compared to the approach in [2], and (2) we can successfully obtain the control parameter values even when the traffic arrivals are non-Poisson.



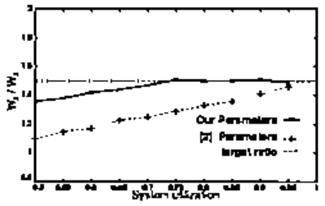
(a) W_1/W_2, $r^t_{i,i+1} = 1.2$



(b) W_2/W_3, $r^t_{i,i+1} = 1.2$



(c) W_1/W_2, $r^t_{i,i+1} = 1.5$



(d) W_2/W_3, $r^t_{i,i+1} = 1.5$

Figure 5: Pareto input traffic: achieved waiting time ratio vs. system utilization.

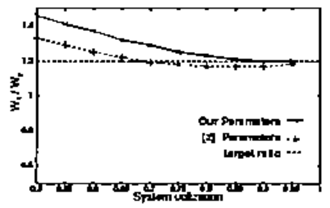
Experiment B.2 (Mixed traffic): In this experiment, we consider three vastly different input traffic sources. Class 1 traffic is generated based on a Poisson process with rate λ, class 2 traffic is generated based on a Pareto distribution with shape parameter $\alpha = 1.9$, and class 3 is generated based on a two-state MMPP[4]. Specifically, the two-state MMPP has four parameters $\lambda_1, \lambda_2, \mu_1$ and μ_2, where λ_1 and λ_2 are the conditional traffic arrival rates, and μ_1 and μ_2 are the conditional transitional rates given that the Markov chain is in states 1 and 2, respectively. These four parameters are computed based on five values: the mean arrival rate of the overall process m_1, the variance of the arrival rate m_2, the third moment of the arrival rate m_3, the time interval Δt, and the lag-1 autocorrelation coefficient c. Note that Δt and c can control the length of the resident time in a state of the Markov Chain. For simplicity, we set $m_3 = 0$ and $c = 1/e$ in all our experiments. The parameters are related as follows:

$$\tau = -\frac{\Delta t}{\ln c_1}; \quad \delta = \frac{m_3}{\sqrt{m_2^3}}; \quad \eta = 1 + \frac{\delta}{2}[\delta - \sqrt{4+\delta^2}] \tag{16}$$

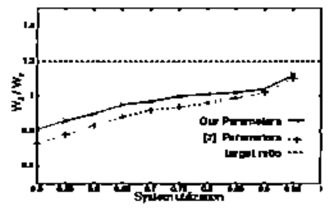
$$\lambda_1 = m_1 + \sqrt{\frac{m_2}{\eta}}; \quad \lambda_2 = m_1 - \sqrt{m_2\eta}; \tag{17}$$

$$\mu_1 = \frac{1}{\tau(1+\eta)}; \quad \mu_2 = \frac{\eta}{\tau(1+\eta)} \tag{18}$$

The experiment setting in this case is similar to the setting of Experiment B.1. Here, we set $r^t_{i,i+1} = 1.2$, $m_2 = 0.01, m_3 = 0$ and $\tau = 500$. Each class of traffic contributes the same amount of system load. The results are summarized in Figures 6. From these figures, observe that both our iterative algorithm and the approach in [2] are *not effective* in achieving the target waiting time ratios. However, as we will show in the following section, we can obtain much better results using a dynamic adjustment algorithm to find the control parameter values. We will give a more detailed explanation of this phenomenon in the section.



(a) W_1/W_2, $r^t_{i,i+1} = 1.5$



(b) W_2/W_3, $r^t_{i,i+1} = 1.5$

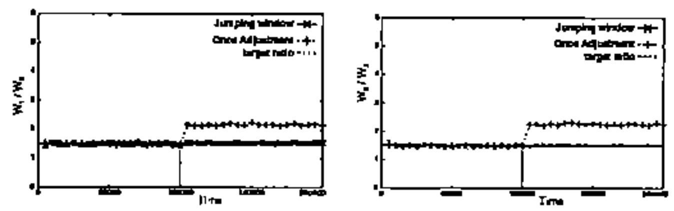
Figure 6: Achieved waiting time ratio against system utilization for heterogeneous traffic classes.

5.3 Type C: Dynamic Adjustment Algorithms

In this subsection, we consider the use of a dynamic adjustment algorithm to monitor changing traffic loads and dynamically adjust the control parameter values in response to the changes. Our goal is to evaluate the effectiveness of such an approach.

We first give an outline of the experiments carried out. In the first experiment, we aim to demonstrate that some form of dynamic adjustment is needed when system utilization can vary significantly. In a second set of experiments, we study the performance of *different* dynamic adjustment algorithms under Poisson arrivals. (Further results showing how the adaptation window size can impact performance are found in [11].) We also carry out similar experiments for Pareto and MMPP traffic arrivals. Lastly, we conduct experiments to study the short-term behavior of dynamic adjustment. For the experiments described below, we consider three classes of traffic. The load is evenly distributed among the three classes. The packet length distribution is the same for all classes, where 40% of the packets are 40 bytes, 50% are 550 bytes, and 10% are 1500 bytes. The output link capacity is 441 bytes/unit time.

Experiment C.1 (Necessity of dynamic adjustment): In this experiment, we illustrate that it is *necessary* to use some form of dynamic adjustment to cope with system load changes. There are three independent classes of arrivals. Each of them is Poisson distributed. Initially, the arrival rates of the three classes are 0.2, 0.2 and 0.2, respectively. At time 100,000 p-unit, the arrival rates of the three classes increase to 0.3, 0.3 and 0.3, respectively. This implies that the system utilization changes from 0.6 to 0.9 at time unit 100,000. The target ratio $r^t_{i,i+1}$ is 1.5 and the adaptation window (i.e., how often we update load estimates and adjust control parameter values) is chosen to be 100 p-units. We compare the performance of the jumping window algorithm with that using static control, as defined in Section 4. The results are shown in Figure 7. In the figure, the x-axis is the time line while the y-axis is the ratio of the average waiting times. A data point is an average packet waiting time measured over the last 500 time units. The graph shows that dynamic adjustment using the jumping window algorithm can maintain the target waiting time ratios (e.g., $r^t_{i,i+1} = 1.5$) even when the system utilization changes significantly. By monitoring per-class traffic arrival rates, any increase in system utilization can be detected in time. The control parameters are then adjusted to new values consistent with the new utilization. In contrast, static control cannot maintain the target waiting time ratios when the utilization changes.

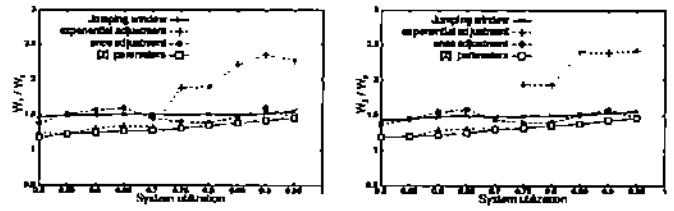


W_1/W_2 W_2/W_3

Figure 7: Achieved waiting time ratio when traffic loading changes at time 100000

Experiment C.2 (Comparisons of dynamic adjustment algorithms under Poisson traffic): In this experiment, we want to compare the effectiveness of the dynamic adjustment algorithms proposed in Section 4. We use three independent classes of Poisson arrivals, and run the adjustment algorithms (i.e., jumping window, exponential averaging, and static control) under different system utilizations. The adaptation window is chosen to be 100 p-units. We also compare the results with the method in [2], in which the parameters are statically chosen as proportional to the target spacings (e.g., $b_1 = 1$, $b_i = b_{i-1} \times r^t_{i-1,i}$). Figure 8 illustrates the result where $r^t_{i,i+1} = 1.5$, while Table 7 illustrates the the result where $r^t_{1,2} = 1.5$ and $r^t_{2,3} = 2.0$. We have the following observations:

1. *Dynamic adjustment is effective.* By using jumping window, the waiting time ratio can be kept close to our target ratios under various utilizations, while the method in [2] cannot achieve the targets most of the time.
2. Exponential averaging fails to keep the waiting time ratios close to 1.5, according to the choice of the input parameters in Fig. 3. The reason is that a correct choice of the `delta` and `sigma` parameters of exponential averaging (Fig. 3) should also depend on the arrival distributions of the different classes of traffic. When the arrival rate fluctuates more, the weight of previous history in parameter estimation should also be increased. Therefore, exponential averaging requires another control algorithm to tune the weightings dynamically, giving rise to another parameter tuning problem. It makes the adjustment problem more difficult and convoluted. On the other hand, jumping window appears more straightforward to tune and is able to maintain the waiting time ratios very well in this experiment (this observation is also supported in our later experiments). Therefore, we believe that for our purposes, the simpler jumping window algorithm can be as effective as the more sophisticated exponential averaging approach.



(a) W_1/W_2 vs. system utilization
(b) W_2/W_3 vs. system utilization

Figure 8: Waiting time ratio vs. system utilization for different dynamic adjustment algorithms, three Poisson traffic sources, $r^t_{i,i+1} = 1.5$, and adaptation window of size 100 p-units

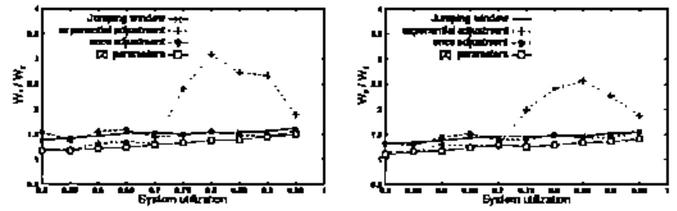
Experiment C.3 (Comparisons of dynamic adjustment algorithms under Pareto traffic): The experiment setting is the same as Experiment C.2, except that

method used	$r^t_{i,i+1}$	$\rho = 0.70$	$\rho = 0.80$	$\rho = 0.90$
jumping window	$r^t_{1,2} = 1.5$	1.50	1.50	1.52
jumping window	$r^t_{2,3} = 2.0$	1.93	1.99	2.05
[2]	$r^t_{1,2} = 1.5$	1.28	1.34	1.42
[2]	$r^t_{2,3} = 2.0$	1.50	1.63	1.78

Table 7: Achieved long term waiting time ratios, for Poisson arrivals under different system utilizations, $r^t_{1,2} = 1.5$ and the $r^t_{2,3} = 2.0$

the arrival processes for the three input traffic classes are Pareto-distributed. The packet inter-arrival times for each class are generated by a Pareto distribution with the shape parameter equal to 1.9. This experiment serves to evaluate algorithm performance under long-range dependent traffic. We experiment with the jumping window, exponential averaging and static control algorithms under system utilizations ranging from 0.5 to 0.95. We also compare the results with the method in [2]. The target waiting time ratios are $r^t_{i,i+1} = 1.5$ and the adaptation window is chosen to be 100 p-units. The results are shown in Figure 9.

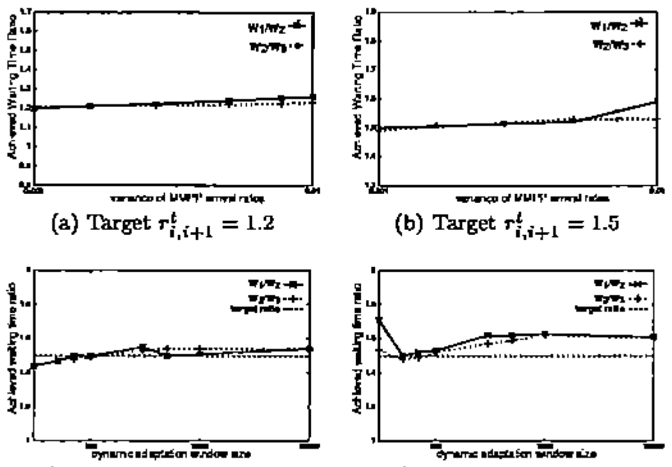
From these figures, we conclude that: (1) Traffic monitoring and parameter adjustment are needed, (2) jumping window dynamic adjustment is effective in maintaining the target waiting time spacings even for long-range dependent traffic, and (3) exponential averaging with the choice of input parameters shown in Fig. 3 is considerably less effective than the simpler jumping window algorithm.



(a) W_1/W_2 vs. system utilization (b) W_2/W_3 vs. system utilization

Figure 9: Waiting time ratio vs. system utilization for different dynamic adjustment, three Pareto traffic sources, $r^t_{i,i+1} = 1.5$, adaptation window is 100 p-units

Experiment C.4 (robustness of jumping window algorithm under different target waiting time ratios, variances of arrival rates, and adaptation window sizes for MMPP traffic): Since the jumping window algorithm has shown the best performance, we use it to study the achieved waiting time ratios for different target ratios $r^t_{i,i+1}$ and different variances of MMPP arrival rates. We have three classes of MMPP traffic, whose parameters are $\tau = 100, m_2 = 0.05$ and $m_3 = 0$ (please refer to Equation(16) and Equation(17) for setting MMPP traffic parameters). The average system utilization is kept at 0.6. Figure 10 illustrates the results. It shows that the jumping window algorithm can maintain the target waiting time ratios under a variety of operating conditions. However, by comparing Figures 10(a) (variance 0.001) and 10(b) (variance 0.05), notice that with a higher variance of input traffic, incorporating too much history with a large adaptation window size can result in suboptimal performance. With small variance of input (Figure 10(a)) traffic, algorithm performance is largely insensitive to the adaptation window size.



(a) Target $r^t_{i,i+1} = 1.2$ (b) Target $r^t_{i,i+1} = 1.5$

(c) $r^t_{i,i+1} = 1.2$, variance=0.001, vs. adaptation window sizes (d) $r^t_{i,i+1} = 1.5$, variance=0.05, vs. adaptation window sizes

Figure 10: Achieved waiting time for MMPP traffic under $\rho = 0.6$ and various conditions

Experiment C.5 (Achieved short-term waiting time ratios using jumping window): The objective of these experiments is to study the short-term behavior of our packet scheduling algorithm using the jumping window dynamic adjustment algorithm. We measure the ratios of the average waiting times between successive classes in consecutive time intervals. (The average is calculated over the waiting times of all packets that get served within a specified adaptation window.) The length of the adaptation window is varied to be 50, 100, 500, and 1000 p-units. The adaptation window size we use in the experiment is 100 p-units, the system utilization is 0.6, and the target waiting time ratios are $r^t_{i,i+1} = 1.5$. As seen from Figures 11 and 12, our iterative algorithm in Section 3, when used with the jumping window dynamic adjustment algorithm, can achieve the target waiting time ratios *better* than the approach in [2]. (Corresponding results for MMPP traffic can be found in [11].) That is, as we increase the monitoring window size, our short-term waiting time ratios will approach the target value of 1.5. On the other hand, the algorithm in [2] appears to approach a value different from the target. Another observation is that we can more accurately achieve the target waiting time ratios as we increase the monitoring window size. In fact, the achieved average waiting time ratios should approach asymptotically the target ratios as the monitoring window size approaches infinity.

Besides the histograms, we present the sample mean and

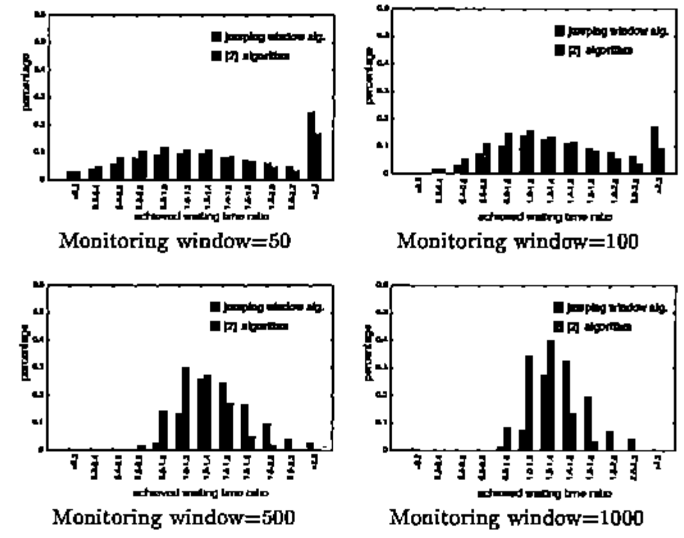


Figure 11: Histograms of short-term waiting time ratio under Poisson arrivals, $\rho = 0.6$ and target waiting time ratio $r^t_{i,i+1} = 1.5$

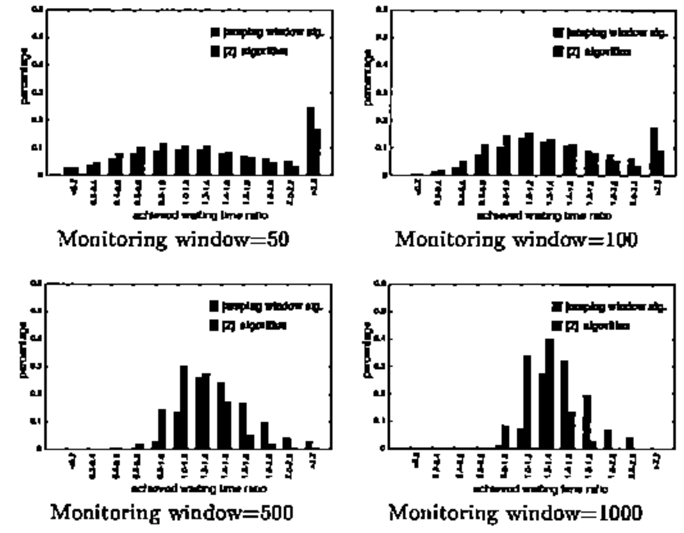


Figure 12: Histograms of short-term waiting time ratio under Pareto arrivals, $\rho = 0.6$ and target waiting time $r^t_{i,i+1} = 1.5$

the modified variance of the waiting times under different monitoring windows. We provide results from experiments under both medium and high system utilizations ($\rho = 0.6$ and 0.9, respectively) . The sample mean and the modified variance are defined as follows. We take the average waiting time ratio obtained in each time monitoring window as a sample. By averaging the samples over the number of monitoring windows measured, the sample mean is obtained. Formally, we have:

$$\text{Sample mean} \;=\; \sum X_i/N \qquad (19)$$

where X_i is the sample average waiting time ratio at the i^{th} monitoring window and N is the total number of monitoring windows. Let r be the target waiting time ratio (in this case $r = 1.5$). The modified variance is defined as:

$$\text{modified variance} \;=\; \sum(X_i - r)^2/N \qquad (20)$$

Note that the modified variance can measure the *deviation* of the sample mean from the target waiting time ratio. As can be seen from Tables 8 and 9, our proposed jumping window dynamic adjustment algorithm is more *accurate* in achieving the target waiting time ratios, for Poisson arrivals. Corresponding results for Pareto and MMPP arrivals support the conclusion, and can be found in [11].

Monitoring window size (p-unit)	100	500	1000
Sample mean for jumping window	2	1.57	1.50
Modified variance for jumping window	368	0.117	0.067
Sample mean for [2]'s method	1.63	1.25	1.23
Modified variance for [2]'s method	282	0.124	0.102

Table 8: Poisson arrivals with utilization=0.6, waiting time target ratio=1.5

Monitoring window size (p-unit)	100	500	1000
Sample mean for jumping window	1.5	1.5	1.51
Modified variance for jumping window	0.112	0.029	0.017
Sample mean for [2]'s method	1.4	1.41	1.41
Modified variance for [2]'s method	0.103	0.028	0.019

Table 9: Poisson arrivals with utilization=0.9, waiting time target ratio $r^t_{i,i+1} = 1.5$

5.4 Type D: End-to-end Accumulative Waiting Time

In this subsection, we illustrate the end-to-end accumulative waiting time performance for a sequence of nodes employing the adaptive WTP scheduling algorithm. We carry out three sets of experiments. The first one consists of three heterogeneous nodes and three classes of end-to-end traffic. The second one consists of three homogeneous nodes, two classes of end-to-end traffic, and a class of cross traffic at each link. The third one consists of three heterogeneous nodes, two classes of end-to-end traffic, and a heterogeneous class of cross traffic going through each link.

Experiment D.1 (Heterogeneous nodes): The simulation setup is as follows. There are three nodes connected in series (Figure 13), and three classes of traffic going through nodes 1, 2, and 3, before arriving at a common destination. Each class is a Poisson source with the same arrival rate $\lambda/3$. The packet length distribution is the same for all the classes, where 40% of the packets are 40 bytes, 50% are 550 bytes, and 10% are 1500 bytes. The link capacities of 1-2, 2-3 and 3-destination are 441 bytes/unit time, 551 bytes/unit time and 735 bytes/unit time, respectively[1].

[1]Since our algorithm controls only the queuing delay at each node, we do not model the link propagation delay in our experiments.

(The time unit can be normalized to achieve an arbitrary link speed.) Hence, the capacities of 2-3 and 3-destination are 25% and 67%, respectively, higher than that of 1-2. We run different simulations with λ ranging from 0.80 to 0.95. At each node, the jumping window adjustment algorithm is used, and the adjustment window size is 100 p-units. In a simulation, each traffic class generates at least 50,000 packets. We measure the achieved long term waiting time ratios between consecutive classes at each node, and their achieved long term end-to-end accumulative waiting time ratios. The target waiting time ratios $r^t_{i,i+1}$ are 1.5. The results are summarized in Table 10.

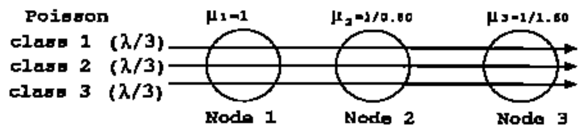


Figure 13: The multiple-node traffic in Experiment D.1

Achieved waiting time ratio $r^a_{i,i+1}$	total traffic arrival rate λ			
	$\lambda =$ 0.80	$\lambda =$ 0.85	$\lambda =$ 0.90	$\lambda =$ 0.95
Node 1 (W_1/W_2)	1.51	1.51	1.51	1.56
Node 2 (W_1/W_2)	1.65	1.66	1.60	1.63
Node 3 (W_1/W_2)	1.67	1.72	1.67	1.72
Node 1 (W_2/W_3)	1.49	1.50	1.53	1.54
Node 2 (W_2/W_3)	1.55	1.54	1.57	1.54
Node 3 (W_2/W_3)	1.50	1.51	1.57	1.59
end-to-end accumulative waiting time (W_1/W_2)	1.55	1.56	1.54	1.57
end-to-end accumulative waiting time (W_2/W_3)	1.50	1.51	1.54	1.54

Table 10: Multiple node waiting-time ratios with different link capacities under Poisson arrivals and target waiting-time ratios $r^t_{i,j} = 1.5$.

Experiment D.2 (Homogeneous nodes with cross traffic): The simulation setup is as follows. There are

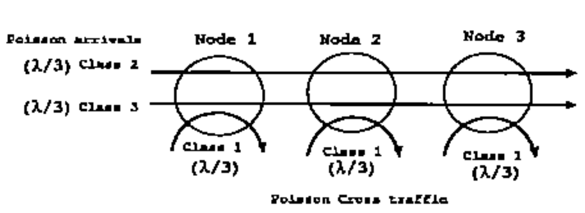


Figure 14: Multiple-node delay differentiations in the presence of cross traffic (Experiment D.2).

three nodes connected in series (Figure 14). Two classes of traffic (class 2 and class 3), each Poisson with arrival rate $\lambda/3$, go through nodes 1, 2, and 3, in that order, before arriving at a common destination. In addition, a class 1 Poisson source of cross traffic, also of rate $\lambda/3$, traverses each link. The packet length distribution is the same as in Experiment D.1, and the capacity of each link is 441 bytes/unit time. We run different simulations with λ ranging from 0.5 to 0.95. We measure the achieved long-term waiting time ratio between class 2 and class 3 traffic at each node, and their achieved long-term end-to-end accumulative waiting time ratio. The target waiting-time ratios $r^t_{i,i+1}$ are 1.5. The results are summarized in Table 11.

Achieved waiting time ratio $r^a_{i,i+1}$	total traffic arrival rate λ				
	$\lambda =$ 0.50	$\lambda =$ 0.60	$\lambda =$ 0.70	$\lambda =$ 0.80	$\lambda =$ 0.90
Node 1 (W_2/W_3)	1.43	1.49	1.50	1.50	1.52
Node 2 (W_2/W_3)	1.42	1.42	1.42	1.42	1.45
Node 3 (W_2/W_3)	1.41	1.40	1.42	1.42	1.46
end-to-end accumulative waiting time (W_2/W_3)	1.42	1.44	1.44	1.45	1.49

Table 11: Multiple-node waiting-time ratios with cross traffic under Poisson arrivals; target waiting-time ratios are $r^t_{i,i+1} = 1.5$.

Experiment D.3 (Heterogeneous nodes with heterogeneous cross traffic): The simulation setup is as

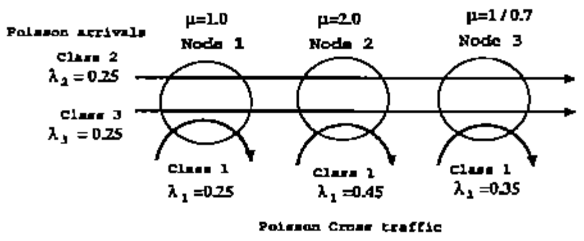


Figure 15: Multiple heterogeneous node delay differentiation in the presence of heterogeneous cross traffic (Experiment D.3).

follows. There are three nodes connected in series (Figure 15). Two classes of end-to-end traffic (classes one and two), each Poisson with rate 0.25, go through nodes 1, 2 and 3, in that order, and arrive at a common destination. In addition, class one Poisson sources of cross traffic traverse links 1-2, 2-3 and 3-destination with rates 0.25, 0.45 and 0.35, respectively. The packet length distribution is the same as in the previous experiment, while the link capacities of 1-2, 2-3, and 3-destination are 441 bytes/unit time, 882 bytes/unit time, and 630 bytes/unit time, respectively. The target waiting time ratios are $r^t_{2,3} = 1.2$ and $r^t_{2,3} = 1.5$. At each node, the jumping window algorithm is used, with the adjustment window size being 100 p-units. In a simulation, each traffic source generates at least 50,000 packet arrivals. We measure the achieved long-term waiting time ratio between class 2 and class 3 at each node, and also the achieved long-term end-to-end accumulative waiting time ratio between the two classes. The results (with statistics reset after an initial 500 p-unit warmup period) are summarized in Table 12.

achieved waiting time ratio $r^a_{2,3}$	target ($r^t_{2,3} = 1.2$)	target ($r^t_{2,3} = 1.5$)
node 1 W_2/W_3	$r^a_{2,3} = 1.19$	$r^a_{2,3} = 1.50$
node 2 W_2/W_3	$r^a_{2,3} = 1.22$	$r^a_{2,3} = 1.43$
node 3 W_2/W_3	$r^a_{2,3} = 1.21$	$r^a_{2,3} = 1.48$
end-to-end accumulative waiting time (W_2/W_3)	1.20	1.49

Table 12: Achieved multiple-node waiting time ratios with heterogeneous cross traffic under Poisson arrivals (Experiment D.3)

Observations:
There are two major observations from the above experiments: (1) the achieved long-term waiting time ratios are close to the target waiting time ratios ($r^t_{i,i+1} = 1.5$), and (2) the end-to-end accumulative waiting time ratios are also very close to the target waiting time ratios (provided that the flows traverse a *same sequence* of WTP nodes, as in our experiments). The observation holds even when the links traversed have possibly cross traffic and heterogeneous service capacities. Note that the waiting time ratios at nodes 2 and 3 are not as accurate as that at node 1, because the traffic pattern is *distorted* after going through node 1. Thus, the traffic arrivals at nodes 2 and 3 may no longer be Poisson, which causes minor deviations from the target waiting time ratios.

5.5 Summary of Experimental Results

We provide a summary of our experimental results:

- Our proposed iterative algorithm can efficiently determine whether it is feasible to achieve given target waiting time spacings.
- When it is feasible, our proposed iterative algorithm can accurately determine control parameter values such that the achieved waiting time ratios are equal or very close to the target waiting time ratios.
- Although the theoretical result is based on the assumption that the input traffic is Poisson, our results show that even when the input traffic is non-Poisson (e.g., Pareto, MMPP or mixed traffic), the proposed iterative algorithm can still efficiently and accurately determine the control parameter values so as to achieve the given target waiting time ratios.
- Our proposed jumping window dynamic adjustment algorithm can effectively adapt to changes in input traffic intensity and update the control parameters accordingly. Therefore, the achieved waiting time ratios remain close to the target waiting time ratios.
- For short-term waiting time ratio performance, we can achieve the target waiting time ratios over time scales greater than about 1000 p-units.
- If two end-to-end flows traverse a same sequence of nodes each providing proportional-delay differentiated services, their achieved end-to-end accumulative waiting time ratio can be very close to the target waiting time ratio.

6 Conclusion

In this paper, we consider a WTP scheduler to achieve proportional-delay differentiated services. The scheduler tries to ensure that the average waiting time of class i traffic relative to that of class $i-1$ traffic is consistently a specifiable ratio. This way, an ISP can legitimately charge users of class i traffic a higher tariff rate (compared to the rate of class $i-1$ traffic) because class i users consistently enjoy better performance than class $i-1$ users.

For two-class WTP, we obtain a necessary and sufficient condition for a given delay spacing to be feasible. For the general $N-$class WTP, we present a set of necessary conditions, and give their physical meanings. Using these conditions, we can easily determine if a given delay-proportional differentiation can be achieved or not. We also present an efficient iterative algorithm for finding values of the WTP control parameters that will realize a set of specified waiting time spacings, provided that these parameters exist. Since the arrival rates of flows are time varying, we present a dynamic measurement and adaptation technique so that the system can track the arrival rates of each flow and adjust values of the control parameters so as to maintain the target waiting time spacings.

Experiments are carried out to illustrate that using our control parameter values, we can obtain waiting time spacings that are closer to the given target waiting time ratios, when compared with the results in [2]. We also show that the achieved waiting time ratios are close to the target waiting time ratios under short, medium and long timescales for different traffic arrival patterns. We show that by using the dynamic adjustment approach and the iterative algorithm, we can provide proportional waiting time differentiated services under different time scales and under different input arrival processes. Lastly, we demonstrate that useful delay ratios can be obtained even when traffic has to traverse multiple nodes in the network. Future work include, for $N > 2$ classes, how router can efficiently solve the systems of non-linear equations every time the load conditions change.

Acknowledgment

We wish to thank the anonymous referees for their helpful and insightful comments.

References

[1] D. Clark, W. Fang. *Explicit Allocation of best Effort Packet Delivery Service.* IEEE/ACM Transactions on Networking, Vol. 6, pp. 362-373. August, 1998.

[2] C. Dovrolis, D. Stiliadis, P. Ramanathan. *Proportional Differentiated Services: Delay Differentiation and Packet Scheduling.* ACM SIGCOMM'99, pp. 109-119, August, 1999.

[3] C. Dovrolis, *Proportional Differentiated Services for the Internet,* Ph.D. Dissertation, Department of Com-

puter Science, University of Wisconsin, Madison, December 2000.

[4] W. Fisher and K.S. Meier-Hellstern. *The Markov-modulated Poisson process (MMPP) cookbook.* Performance Evaluation, 18:149171, 1992.

[5] R. Guerin, S. Kamat, V. Peris, R. Rajan. *Scalable QoS Provision Through Buffer Management.* ACM SIGCOMM, September, 1998.

[6] Van Jacobson and Mike Karels. *Congestion Avoidance and Control.* Available from ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z, 1992. Original version in Proc. ACM SIGCOMM 88 by Van Jacobson only.

[7] S. Jamin, P. B. Danzig, S. Shenker and L. Zhang. *A Measurement-based Admission Control Algorithm for Integrated Services Packet Networks.* Proc. ACM SIGCOMM, Boston, MA, September 1995.

[8] L. Kleinrock. *Queueing Systems: Volume 2.* Wiley-interscience, 1976.

[9] E. V. Krishnamurthy, S. K. Sen. *Numerical Algorithms Computations in Science and Engineering.* Affiliated East-West Press, 1991.

[10] M. K. H. Leung, J. C. S. Lui, D. K. Y. Yau. *Characterization and Performance Evaluation for Proportional Delay Differentiated Services.* IEEE ICNP, Osaka, Japan, November 2000.

[11] M. K. H. Leung, J. C. S. Lui, D. K. Y. Yau. *Adaptive Proportional-delay Differentiated Services: Characterization and Performance Evaluation*, Technical Report, Department of Computer Science and Engineering, Chinese University of Hong Kong (also as Technical Report CS-01-009, Dept of Computer Sciences, Purdue University, West Lafayette, IN), June 2001.

[12] A. Mankin, F. Baker, B. Braden, S. Bradner, M. O'Dell, A. Romanow, A. Weinrib, and L. Zhang. *RSVP Version 1: Applicability Statement, Some Guidelines on Deployment.* September 1997. IETF RFC 2208.

[13] M. May, J. C. Bolot, C. Diot and A. Jean-Marie. *Simple Performance Models of Differentiated Services Schemes for the Internet.* IEEE INFOCOM, March 1999.

[14] A. Netterman, I. Adiri. *A Dynamic Priority Queue with General Concave Priority Functions.* Operation Research, 27(6), pp. 1088-1100, 1979.

[15] K. Nichols, S. Blake, F. Baker, D. L. Black. *Definition of the Differentiated Service Field (DS Field) in the IPv4 and IPv6 Headers.* IETF RFC 2474, Dec 1998.

[16] K. Nichols, V. Jacobson and K. Poduri. *Expedited Forwarding PHB Group.* Internet RFC 2598, June 1999.

[17] S. Sahu, D. Towsley, J. Kurose. *A Quantitative Study of Differentiated Services for the Internet.* Proc. IEEE Global Internet'99, Rio de Janeiro, Brazil, December 1999.

[18] S. Srisankar Kunniyur and R. Srikant, *A Time Scale Decomposition Approach to Adaptive ECN Marking,* IEEE Infocom, Anchorage, Alaska, April 2001

[19] I. Stoica, S. Shenker, H. Zhang. *Core-Stateless Fair-Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Network.* ACM SIGCOMM, September 1998.

[20] I. Stoica, H. Zhang. *LIRA: An Approach for Service Differentiation in the Internet.* Proceedings NOSS-DAV, 1998.

Matthew K. H. Leung obtained B.Sc (first class honors) from the Department of Computer Science & Engineering at the Chinese University of Hong Kong and continued his M.Phil study there. His research interests include queueing theory and networking. He is currently a member of the technical staff at HSBC.

John C. S. Lui (M '94) received his Ph.D. in Computer Science from UCLA. When he was a graduate student, he participated in a parallel database project in the IBM Thomas J. Watson Research Center. After his graduation, he joined a team at the IBM Almaden Research Laboratory/San Jose Laboratory and participated in research and development of a parallel I/O architecture and file system project. He later joined the Department of Computer Science and Engineering of the Chinese University of Hong Kong. His current research interests are in communication networks, distributed multimedia systems, OS design issues, parallel I/O & storage architectures and performance evaluation theory. His personal interests include films and general reading.

David K. Y. Yau (M '97) received the B.Sc. (first class honors) degree from the Chinese University of Hong Kong, and the M.S. and Ph.D. degrees from the University of Texas at Austin, all in computer sciences. From 1989 to 1990, he was with the Systems and Technology group of Citibank, NA. He was the recipient of an IBM graduate fellowship, and is currently an Assistant Professor of Computer Sciences at Purdue University, West Lafayette, IN. He received an NSF CAREER award in 1999, for research on network and operating system architectures and algorithms for quality of service provisioning.