A Quantitative Study of Differentiated Services for the Internet *

Sambit Sahu, Don Towsley, Jim Kurose
sahu,towsley,kurose@cs.umass.edu
Department of Computer Science
University of Massachusetts
Amherst, MA 01003

Abstract

The Differentiated Services architecture provides router mechanisms for aggregate traffic, and edge mechanisms for individual flows, that together can be used to build services with varying delay and loss behaviors. In this paper, we compare the loss and delay behaviors that can be provided using the services based on combinations of two router mechanisms, *threshold dropping* and *priority scheduling* and two packet marking mechanisms, *edge-discarding* and *edge-marking*. In the first part of our work, we compare the delay and loss behaviors of the two router mechanisms coupled with *edge-discarding* for a wide range of traffic arrivals. We observe that *priority scheduling* provides lower expected delays to preferred traffic than *threshold dropping*. In addition, we find that a considerable additional link bandwidth is needed with threshold dropping to provide same delay behavior as priority scheduling. We further observe little difference in the loss incurred by preferred traffic under both router mechanisms, except when sources are extremely bursty, in which case *threshold dropping* performs better. In the second part of our work, we examine the throughput of a TCP connection that uses a service built upon *threshold dropping* and *edge-marking*. Our analysis shows that a significant improvement in throughput can be achieved. However, we find that in order to fully achieve the benefit of such a packet marking, the TCP window must take the edge-marking mechanism into consideration.

1 Introduction

The diverse and changing nature of service requirements among emerging Internet applications calls for a network architecture that is both flexible and able to distinguish among the needs of the applications. Previous efforts in this direction [26] confirm that a successful deployment of such an architecture necessitates simple mechanisms inside the network. Differentiated Services [1], [4], [20] (diffserv) is such an architecture: the needs of various applications are supported using a simple classification scheme. At a high level, packets are *classified* prior to entering the network via a *packet marking mechanism*, and the services that a router inside the network provides to a packet depend solely on the packet's *class*. Several packet marking and router mechanisms have been proposed [4], [5], [10], [12], [14], [16], [24], [25] in the context of this new service paradigm. However there have been few studies providing an understanding of their comparative merits. The intent of our work is to provide a quantitative comparison of these various mechanisms and to derive the loss and delay behavior of the services that these mechanisms support under a wide range of traffic arrivals.

*This study was supported in part by the National Science Foundation under Grant No. NCR-9527163, NCR-9523807, and CDA-9502639. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

There are two important issues concerning the mechanisms in a diffserv architecture that we examine. The first issue is *how an internal router should treat packets of different classes*. In order to examine this issue, we consider two mechanisms that are representative of two principal proposals [4], [20] that have been widely considered as router mechanisms. One proposal, representative of the mechanism in [20], has a router maintaining separate FIFO queues, one for each class, and serving these queues according to a strict priority rule. The buffer management for each of these queues is based on drop tail mechanism. We refer to this router mechanism as *priority scheduling*. The second proposal representative of the one in [4], distinguishes among the different classes in the way the buffer is managed. This router mechanism, referred to as *threshold dropping*, associates a threshold with each class that is used to determine whether a packet is accepted and the enqueued packets are served according to FIFO scheduling mechanism. Note that while threshold dropping supports multiple service classes by assigning different thresholds, priority scheduling supports multiple service classes by assigning priorities in scheduling the separate buffers. The reason for examining *threshold dropping* and *priority scheduling* mechanisms is that, while the latter allows complete isolation among packet classes, the former one provides very minimal isolation. We expect most often other mechanisms to provide isolation among packet classes that is between the ranges that provided by the above two router mechanisms. Thus, examining *threshold dropping* and *priority scheduling* would provide valuable insights into router mechanisms in the context of diffserv architecture.

The second issue is *whether an edge router should forward packets that fall outside of the "profile"* it has negotiated with the sender. Prior to transmission, there is a negotiation between the edge router and the sender regarding a "service class", that specifies a packet classification, and a "service profile", that specifies an upper bound on the amount of traffic that the sender negotiates to send in the specified service class. The edge router marks all the packets that are within the profile as *in-profile* and the excess as *out-profile*. If the edge-router forwards both *in-profile* and *out-profile* packets into the network, the in-profile and out-profile packets of a single session can potentially observe different loss and delay characteristics due to discriminations provided by the internal router mechanisms. The issue of whether out-profile packets should be forwarded into the network or not has been debated in several papers [4], [6], [11], [16]. Some have argued [4],[16] that allowing out-profile packets into the network improves buffer and link bandwidth utilization when routers use threshold dropping. However, *the consequence of this approach on the performance of specific network protocols, such as TCP, is not understood.*

The first part of our study compares loss and delay behaviors under the two competing router mechanisms: *threshold dropping* and *priority scheduling* when only in-profile packets are forwarded by an edge-router into the network. In order to gain insight into the fundamental differences in the loss and delay behaviors of these two router mechanisms, we start with Poisson model for packet arrivals. We find that it is possible to provide considerably lower delays to preferred packets with priority scheduling than with threshold dropping. In addition, we find that *an additional* $30\% - 70\%$ *link capacity is needed* with threshold dropping in order to provide the same expected delay to preferred packets as provided by priority scheduling. With respect to loss, we find that both router mechanisms provide similar preferred packet loss rates, with threshold dropping providing marginally lower loss.

Next we examine the validity of the above findings under more realistic packet arrivals that we capture via bursty On-Off sources. Using analytical approach, we observe similar results with the following exception: when the sources are extremely bursty and a very small amount of loss is allowed, threshold dropping utilizes the buffer and link bandwidth more efficiently.

The second part of our study examines the effect that accepting both in-profile and out-profile packets of a session into the network has on a TCP connection with threshold dropping. We assume that in-profile packets are preferred over out-profile packets at internal routers. Based on a simple Markov model, we make two interesting observations for the case when the loss probabilities of the in-profile and out-profile packets are significantly different: 1) there is a significant improvement in the throughput of a TCP connection even when a fraction of traffic is sent as in-profile packets; 2) in order to fully utilize the benefits of such a marking scheme, the choice of the maximum TCP window size should take several parameters into account, such as the loss probabilities of in-profile and out-profile packets, and the sender service profile. Otherwise, the benefits of allowing out-profile packets into the network may not be achieved.

Several studies [2], [3], [13], [17], [19], [27] examine delay and/or loss behavior of services in the diffserv architecture using a variety of traffic models. May et al. [17] quantify the expected delay and loss of packets that arrive via a Poisson process, for assured [4] and premium services [20]. Naser et al. [19] quantify the expected delay of packets for constant bit rate and On-Off source traffic using a two-bit architecture which combines assured and premium services. Ibanez et al. [13] consider ways in which assured service can be used to improve TCP throughput, and demonstrate the difficulties in quantifying the impact of assured service on TCP. Yeom et al. [27] perform a similar study, and propose a rather elaborate mechanism to support TCP. Finally, Shenker et al. [2], [3] show that the usefulness of a priority-based service depends heavily on the adaptive capabilities of applications.

While each of these pieces of work contributes to the understanding of the benefits and limitations of the various services, yet the question that, we believe, is of interest to the networking community remains unanswered: given a set of session requirements, what router and packet marking mechanisms should be provided in a diffserv architecture to best satisfy the session requirements? We address this question through a quantitative comparison of the services that are built from the two router mechanisms and two packet marking mechanisms. We find that the answer depends on both the service requirements and the traffic model.

The rest of the paper is organized in the following way. In Section 2, we provide the details of diffserv architecture and derive a framework for comparison of two router mechanisms. In Section 3, we compare the router mechanisms when sources are characterized by Poisson arrivals using Markov analysis. In Section 4, we consider traffic that is characterized by Markov modulated On-Off sources and provide comparative analysis via fluid model when the sources require strict quality of service. In Section 5, we study the effect of forwarding both in-profile and out-profile packets of a session on the throughput of TCP connection using the Markov model. Section 6 concludes the paper.

2 Diffserv Architecture and Our Model

The major components of a differentiated services architecture are: (1) edge mechanisms that include metering, marking, shaping and policing individual flows at an edge router and (2) core-router mechanisms, i.e., router mechanisms inside the network for buffering and forwarding aggregate traffic. An edge router provides the access point for the end-host to the core network. Prior to packet transmission, a sender specifies a *service class*, that defines a packet classification, and a *service profile* that indicates the amount of traffic that the sender is allowed to send in the specified class. The edge router monitors the sending rate to determine if the sender exceeds the service profile. It classifies packets falling within the service profile as *in-profile* and *out-profile*

Metric	Threshold Dropping
p_h^{TD}	loss probability for preferred packets
p_l^{TD}	loss probability for non-preferred packets
d_h^{TD}	expected delay for preferred packets
d_l^{TD}	expected delay for non-preferred packets
B_l^{TD}	buffer threshold for non-preferred packets
B	total buffer space

Metric	Priority Scheduling
p_h^{PS}	loss probability for preferred packets
p_l^{PS}	loss probability for non-preferred packets
d_h^{PS}	expected delay for preferred packets
d_l^{PS}	expected delay for non-preferred packets
B_l^{PS}	buffer allocated to non-preferred packets
B_h^{PS}	buffer allocated to preferred packets

Table 1: Definition for Notations

otherwise, and inserts a tag in the packet header indicating both the service class and whether the packet is in-profile or out-profile. In addition to marking packets, the edge router can decide whether to shape, drop or forward the out-profile packets into the network. As we shall see later, this decision has a significant impact on the network resource utilization, network protocol and service behavior of each class. The router inside the network differentiates the packets based only on their marking by the edge router. As a result, the mechanisms inside the network are simple, requiring no per-flow state information.

In order to compare different router mechanisms that have been previously considered [4], [16], [20], we introduce two router mechanisms, *threshold dropping* (TD) and *priority scheduling* (PS). Most of the router mechanisms that have been considered can be classified as either TD or PS or some variation of these two, based on how they buffer and schedule packets. For the rest of the paper, we assume that router mechanisms distinguish between two classes of packets, *preferred* and *non-preferred*, with packets in the *preferred* class receiving "preference" over the packets in the *non-preferred* class. The threshold dropping (TD) router mechanism enqueues and serves packets in the following way: Let B denote the total buffer size, and B_l^{TD} denote a buffer threshold for non-preferred packets. Let $B(t)$ denote the buffer occupancy at time t. All accepted packets are queued into a single buffer and served according to a *FIFO scheduling mechanism*. A preferred packet arriving at time t is accepted as long as there is space in the buffer, i.e., $B(t) < B$. A non-preferred packet is accepted only if $B(t) \leq B_l^{TD}$. Note that TD allows buffer sharing between preferred and non-preferred packets as long as $B(t) \leq B_l^{TD}$. We shall refer to this router mechanism as TD with the implicit assumption that enqueued packets are served according to FIFO scheduling. Under PS router mechanism, the buffer space is partitioned, i.e., one buffer of size B_h^{PS} is allocated to the preferred packets and the second, of size B_l^{PS}, to the non-preferred packets. A packet is admitted as long as there is space in the corresponding buffer. We assume that the buffers are selected for service according to *strict priority scheduling*, i.e., a non-preferred packet receives service only when the preferred queue is empty. We shall refer this combination of droptail buffer management with priority scheduling for serving the packets as PS router mechanism.

We consider two packet marking mechanisms, *edge-discard* and *edge-marking*, that an edge router uses to mark the incoming packets of individual flows. The *edge-discard* (ED) mechanism marks in-profile packets and forwards them into the network while discarding out-profile packets. The *edge-marking* (EM) mechanism forwards both in-profile and out-profile packets into the network. We define *a service as a combination of a packet marking and a router mechanism*. In this work we will consider services built by combining ED or EM with TD or PS.

3 Threshold Dropping vs. Priority Scheduling with Edge-discarding: Poisson Arrival Model

In this section, we compare the loss and delay characteristics that ED coupled with either ED or PS provide at an internal router. We assume that packets of each class arrive according to a Poisson process and packet transmission times are exponentially distributed. With an appropriate Markov model formulation, we derive the expressions for the loss probability and expected delays of preferred and non-preferred packets under both services. More *realistic traffic models* are studied in the next section. The assumption that arrivals are described by Poisson processes allows us to examine several issues using simple analysis. We shall use the notations described in Table 1.

3.1 Markov Model for Threshold Dropping with Edge-discarding

We assume that preferred and non-preferred packets arrive according to Poisson processes with rates λ_h and λ_l respectively. The packet service times are assumed to be exponential random variables with parameter μ. A non-preferred packet is accepted only if the buffer occupancy is less than B_l^{TD} and a preferred packet is accepted as long as there is a space in the buffer. Let $N(t)$ denote the number of packets in the system at time $t \geq 0$, including the one currently being served. It can be easily shown that $\{N(t) : t \geq 0\}$ is a birth-death process. Note that a non-preferred packet is dropped when $N(t) \geq B_l^{TD} + 1$, hence the arrival rate to the buffer when $N(t) \geq B_l^{TD} + 1$ is λ_h and is $\lambda_h + \lambda_l$ when $N(t) < B_l^{TD} + 1$.

Let π_i denote the steady state probability that there are i packets in the system, i.e., $\pi_i = \lim_{t\to\infty} Pr\{N(t) = i\}$. Solving the above birth-death process, we obtain loss probabilities and expected delays for both traffic classes. Let $\rho_h = \lambda_h/\mu$ and $\rho_l = \lambda_l/\mu$. The loss probabilities are given by:

$$p_h^{TD} = \frac{1}{N_f} (\rho_h + \rho_l)^{B_l^{TD}+1} \rho_h^{B-B_l^{TD}} \quad (1)$$

$$p_l^{TD} = \frac{1}{N_f} (\rho_h + \rho_l)^{B_l^{TD}+1} \sum_{i=0}^{B-B_l^{TD}} \rho_h^i \quad (2)$$

where $N_f = \sum_{i=0}^{B_l^{TD}+1} (\rho_h + \rho_l)^i + (\rho_h + \rho_l)^{B_l^{TD}+1} \sum_{i=1}^{B-B_l^{TD}} \rho_h^i$. Note that $P_0 = 1/N_f$. Similarly, the expected delay suffered by each traffic type can be derived to be:

$$d_h^{TD} = \frac{P_0}{\mu} \left(1 + \sum_{i=1}^{B_l^{TD}} (i+1) (\rho_h + \rho_l)^i + (\rho_h + \rho_l)^{B_l^{TD}} \sum_{i=1}^{B-B_l^{TD}} (B_l^{TD} + i + 1)\rho_h^i \right) \quad (3)$$

$$d_l^{TD} = \frac{P_0}{\mu} \frac{\left(1 + \sum_{i=1}^{B_l^{TD}} (i+1) (\rho_h + \rho_l)^i\right)}{\sum_{i=1}^{B_l^{TD}} (\rho_h + \rho_l)^i} \quad (4)$$

Similar analysis for a different packet marking scheme is provided in [17].

3.2 Markov Model for Priority Scheduling with Edge-discarding

We now derive the corresponding expressions for loss and expected delays when PS is used with TD. We assume the same packet arrival model that we considered with TD. For simplicity, we further assume that buffers are served according to pre-emptive priority scheduling. We do not expect our results to differ much from the case when non pre-emptive priority scheduling is used. Let $N(t) = \{N_h(t), N_l(t)\}$ denote the state at time t, where $N_h(t), N_l(t)$ denote the number of preferred and non-preferred packets in the system respectively at time $t \geq 0$. The process $\{N(t) : t \geq 0\}$ is a 2-dimensional Markov chain. Let Q denote the infinitesimal matrix generator for the above described Markov chain. Let $\pi(i, j)$ be the steady state probability that there are i preferred packets and j non-preferred packets present. Solving $\pi Q = 0$ yields the steady state distribution of the packets in the system. Using this, we find the loss probabilities and expected delays for both classes of traffic. A preferred packet is lost when, upon arrival, it finds B_h^{PS} outstanding preferred packets in the queue. Hence the preferred packet loss probability can be derived as:

$$p_h^{PS} = \sum_{i=0}^{B_l^{PS}+1} \pi(B_h^{PS} + 1, i) \tag{5}$$

Similarly, the loss probability for non-preferred packets can be derived as:

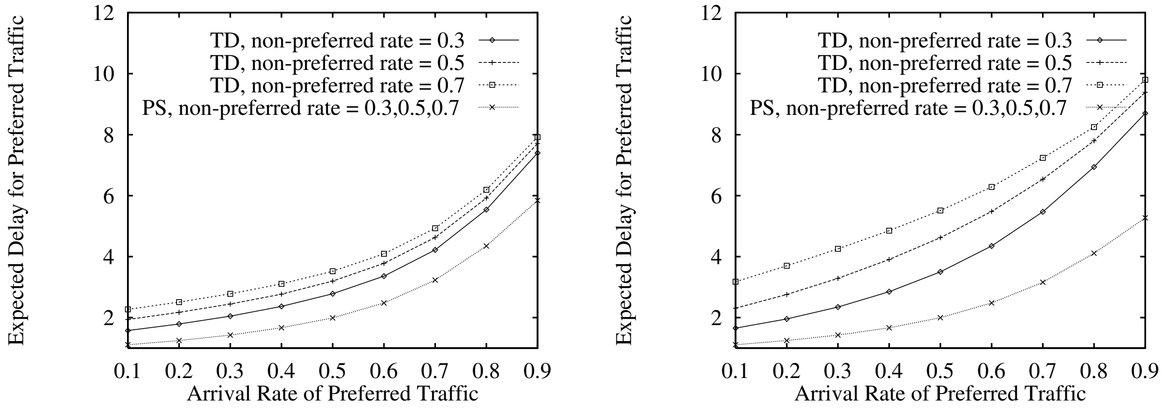
$$p_l^{PS} = \sum_{i=1}^{B_h^{PS}+1} \pi(i, B_l^{PS} + 1) + \left(1 + \frac{\lambda_h}{\lambda_l}\right) \pi(0, B_l^{PS} + 1) \tag{6}$$

The expected delays for each class of traffic can be determined by applying Little's law for each class separately as $d_h^{PS} = \frac{E[N_h]}{\lambda_h(1-p_h^{PS})}$ and $d_l^{PS} = \frac{E[N_l]}{\lambda_l(1-p_l^{PS})}$, where $E[N_h]$, $E[N_l]$ denote the expected number of preferred and non-preferred packets in the system.

3.3 Comparison of Threshold Dropping and Priority Scheduling

Loss Comparison: We want to compare the preferred packet loss probability as a function of the arrival rate λ_h under TD and PS. As there are several parameters that have to be chosen carefully for a fair comparison, we approach the problem in the following way: First we fix B_l^{TD} which determines the non-preferred packet loss probability p_l^{TD} as a function of arrival rates λ_l and λ_h. We then determine the buffer partition B_l^{PS}, B_h^{PS} such that p_l^{PS} is as close to p_l^{TD} as possible. If multiple partitions satisfy this constraint, we choose the one that minimizes p_h^{PS}. If there is no such partition, we determine the additional amount of buffer that is required to satisfy this condition. Last, we determine p_h^{TD} and p_h^{PS} as a function of arrival rate λ_h of preferred packets for different values of λ_l and B_l^{TD}.

Expected Delay Comparison: Next we compare the expected delay incurred by the preferred packets as a function of arrival rates λ_h and λ_l. Similar to the loss comparison, we first fix B_l^{TD}. For a given value of



(a) $B_l^{TD} = 2$

(b) $B_l^{TD} = 5$

Figure 1: Delay Comparison

λ_h and λ_l, we determine p_l^{TD},p_h^{TD}, and d_h^{TD}. Next we determine the buffer partition B_h^{PS}, B_l^{PS} such that $p_h^{PS} \leq p_h^{TD}$ and $p_l^{PS} \leq p_l^{TD}$. If multiple partitions satisfy this constraint, we choose the one that minimizes d_h^{PS}. We determine and compare d_h^{TD} and d_h^{PS} as a function of the arrival rate of preferred packets for different values of the non-preferred packet arrival λ_l.

Let us first consider the effect that the choice of B_l^{TD} has on performance. Intuitively, too small a value of B_l^{TD} should result in a very high loss rate for non-preferred packets. On the other hand, too large a value of B_l^{TD} can affect the loss and delay of preferred packets significantly. Hence in order to examine the impact of B_l^{TD} on the loss and expected delay, we consider several values for λ_l and B_l^{TD}. Specifically, we use $\lambda_l = 0.3, 0.5, 0.7$, and $B = 15$, and $B_l^{TD} = 2, 5$. We choose these values for B_l^{TD} following the recommendations in [5] that B_l^{TD} should not be too close to B.

Figure 1(a) compares the expected delay incurred by a preferred packet as a function of the preferred packet arrival rate, λ_h, when $B_l^{TD} = 2$. We observe that under PS, a preferred packet incurs lower expected delay than under TD. As the non-preferred packet arrival rate increases, the expected delay for a preferred packet increases under TD. On the other hand, under PS, d_h^{PS} *is not affected by changes in the non-preferred packet arrival rate*. Now we examine the sensitivity of these observations when B_l^{TD} is increased. Figure 1(b) illustrates similar results for $B_l^{TD} = 5$. We observe an increase in the expected delay incurred by preferred packets with threshold dropping over the case when $B_l^{TD} = 2$, while the expected delay is unaffected under priority scheduling. We expect these results not to defer much when non-preemptive priority scheduling is used instead of pre-emptive priority scheduling. The key observations from these two set of numerical comparisons are:

- Due to buffer isolation and priority scheduling, preferred packets are not affected by the behavior of non-priority packets with the PS router mechanism.
- There is no mechanism to provide lower expected delays to preferred packets with TD without substantially increasing the loss rate of non-preferred packets.

Figure 2 compares the loss probabilities of preferred packets under TD and PS as a function of arrival rates λ_h and λ_l. Here we have chosen $B_l^{TD} = 5$. We observe that for low values of λ_l, preferred packets

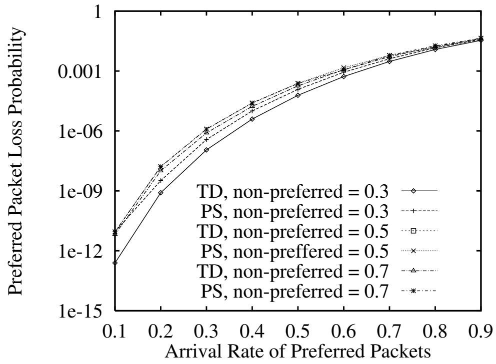


Figure 2: Loss Comparison with Moderate Provisioning for Non-Preferred Packets

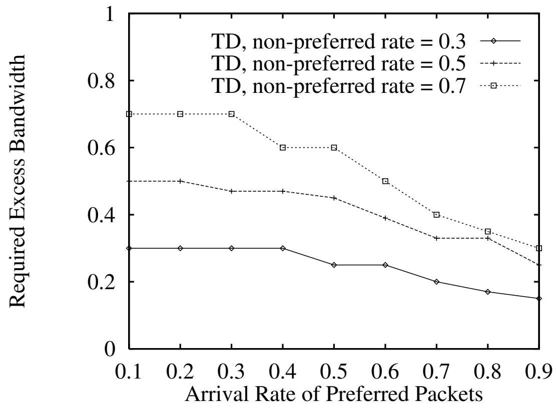


Figure 3: Excess Bandwidth Required with Threshold Dropping

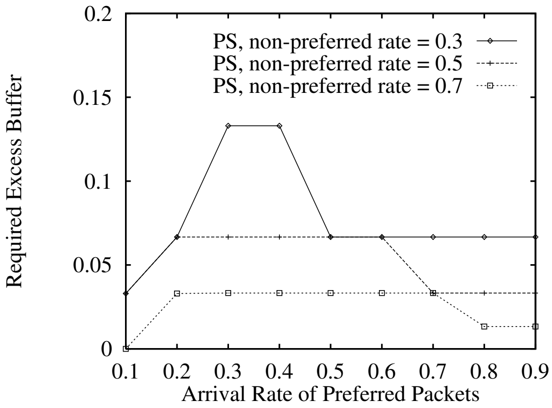


Figure 4: Excess Buffer Required with Priority Scheduling

incur lower loss with TD. However, the loss probabilities are similar for higher values of λ_l. This is due to the benefits of buffer sharing under TD, which are more pronounced when λ_l is relatively low. Such sharing is not possible under PS due to buffer partitioning. However, this multiplexing gain is reduced as the arrival rate of non-preferred packets increases. We observe even smaller difference between the loss probabilities that preferred packets incur under TD and PS when $B_l^{TD} = 2$.

3.4 Excess capacity Analysis and Observations

We observed in Section 3.3 that preferred packets incur considerably lower expected delays under priority scheduling. Also we observed very little difference between the preferred packet loss under these two mechanisms except when λ_l is smaller. In this case, TD provides lower loss to preferred packets. We now determine the additional resources that are needed to achieve 1) identical expected delays for preferred packets under threshold dropping, and 2) identical preferred packet loss under priority scheduling.

Let Δc denote the additional link bandwidth that is required under TD to achieve an expected preferred packet delay identical to that under PS. Figure 3 shows Δc as a function of the preferred packet arrival rate,

λ_h. We observe that $30\% - 70\%$ additional bandwidth is required with TD to achieve identical performance. Figure 4 illustrates the additional amount of buffer that is required in order for PS to achieve the same loss for the preferred traffic as that achieved with TD for the case illustrated in Figure 2. The results indicate that less than 15% additional buffer is sufficient to achieve the same loss rate for preferred packets

Summarizing the results in this section, we observe that preferred packets incur lower delays under PS and, more important, this result is insensitive to non-preferred packet arrival rate λ_l. We observe that a considerable amount of additional bandwidth is needed under TD to provide the same expected delays to preferred packets as with PS. On the other hand, it is possible to achieve a lower preferred packet loss rate under TD, although it is only marginally better than under PS. This is largely due to the benefits of buffer sharing under TD. Moreover, Figure 4 indicates that only a small amount of additional buffer is required to achieve identical preferred packet loss under PS.

These two observations lead us to conclude that PS is a better router mechanism for providing loss and delay guarantees, when packets arrive according to a Poisson process. It also suggests that, in order to support multiple levels of services with different delay requirements, PS should be an important component in a diffserv architecture. Our analysis so far has assumed Poisson traffic. In order to understand the effect of bursty arrivals on our findings, we next compare these two mechanisms for bursty arrivals.

4 Effect of Bursty Sources: Fluid Models

In this section, we examine whether our results change if bursty arrivals are considered. We model bursty arrivals by a set of Markov modulated "On-Off" sources [18]. Each "On-Off" source is characterized by a generator matrix $G = \begin{pmatrix} -\alpha & \alpha \\ \beta & -\beta \end{pmatrix}$, where $1/\alpha$ and $1/\beta$ are the mean "Off" and mean "On" periods. The source toggles between the On and Off states, where the duration of the On (Off) period is exponentially distributed with parameter β (α). When a source is in the On state, it transmits fluid at a constant rate r_c and is silent when in the Off state. We define the burstiness, λ, of a source to be the ratio of its mean On and Off periods, i.e., $\lambda = \alpha/\beta$. Source i is then considered to be burstier than source j if $\lambda_i < \lambda_j$.

We model a constant server with capacity c, i.e., it can serve c amount of fluid per unit time. We assume that the preferred traffic is characterized by a homogeneous mix of N On-Off sources. The number of sources that are in the On state at any given time t is described by a Markov chain with generator matrix M, where M can determined from the generator matrix G. The $(i,j)^{th}$ element of the matrix M is given by:

$$M_{ij} = \begin{cases} i\beta & : \quad j = i-1 \\ -(i\beta + (N-i)\alpha) & : \quad j = i \\ (N-i)\alpha & : \quad j = i+1 \end{cases}$$

The non-preferred traffic is modeled as a constant bit rate (CBR) source that transmits fluid at rate r. We further assume that the preferred traffic is inelastic, i.e., the preferred traffic requires strict delay and loss guarantees.

We consider the following scenario for comparing the delay and loss behaviors of TD and PS. In the case of PS, we assume that the peak rate admission control is used to decide whether a preferred source can be admitted or not. This is a specific implementation of premium service [20] that provides no loss and zero delay to preferred traffic. On the other hand, we slightly relax this strict loss and delay requirements with TD, i.e., we allow a loss in the range of 0.001% and a delay of 0.01. The reason for allowing this relaxation is that buffer sharing under TD between preferred and non-preferred traffic makes it impossible to provide zero loss and zero delay to preferred traffic without significantly underutilizing the buffer and link bandwidth. Our intent is to examine (1) whether peak-rate admission control underutilizes the resources with PS when traffic is bursty and (2) whether a slight relaxation in loss and delay requirements makes TD an attractive alternative.

4.1 Fluid Model for Threshold Dropping with Edge-discarding

We first derive the expressions for the loss probability and expected delay for the preferred traffic and loss probability for the non-preferred traffic. We use a similar analysis as in [8]; the main difference between our model and the one in [8] is that, we have uncorrelated preferred and non-preferred traffic. Let Σ_t denote the number of On-Off sources that are in the "On" state at time t . Let Q_t denote the queue length at time t. We denote the set of all possible aggregate On-Off source states by $\mathcal{S} = \{0, 1, 2, \ldots, N\}$. Define $\pi_i(x)$ as the state distribution of the system in equilibrium, when $\Sigma_t = i$ and $Q_t \leq x$. Thus $\pi_i(x)$ is denoted by:

$$\pi_i(x) = \lim_{t \to \infty} Pr[\Sigma_t = i, Q_t \leq x], \qquad \forall i \in \mathcal{S}, 0 \leq x \leq B \tag{7}$$

Let $\pi(x)$ denote the vector representing the stationary distribution of the system for queue length less than x, i.e., $\pi(x) = [\pi_0(x), \pi_1(x), \ldots, \pi_N(x)]$. Using the approach as in [18], [8] for modeling fluid sources, the two sets of differential equations can be derived for representing the dynamics of the system. One set of equations describes the behavior when $\Sigma_t < B_l^{TD}$ and the other when $B_l^{TD} < \Sigma_t < B$. These are given by:

$$\frac{d}{dx}\pi(x)D^{(0)} = \pi(x)M, \quad 0 < x < B_l^{TD} \tag{8}$$

$$\frac{d}{dx}\pi(x)D^{(1)} = \pi(x)M, \quad B_l^{TD} < x < B \tag{9}$$

where $D^{(0)} = diag\,\{r - c, r + r_c - c, \ldots, r + Nr_c - c\}$, and $D^{(1)} = diag\{-c, r_c - c, 2r_c - c, \ldots, Nr_c - c\}$ are referred as drift matrices. $D_{ii}^{(0)}$ represents the rate of change of buffer when i sources are in the "On" state for the case $0 < x < B_l^{TD}$. The solutions of these equations are given by [18]:

$$\pi(x) = \begin{cases} \pi^{(0)}(x) = \sum_{i=0}^{N} a_i^{(0)} \phi_i^{(0)} exp(z_i^{(0)} x), & 0 \leq x \leq B_l^{TD} \\ \pi^{(1)}(x) = \sum_{i=0}^{N} a_i^{(1)} \phi_i^{(1)} exp(z_i^{(1)} x), & B_l^{TD} < x \leq B \end{cases}$$

where, $\{z_i^{(j)}, \phi_i^{(j)}\}$ are the left eigenvalues and the eigenvectors of $M \cdot {D^{(j)}}^{-1}$, $j = 0, 1$. Using the techniques in [18] for deriving the closed form solutions for the eigenvalues and eigenvectors, we determine $\{z_i^{(j)}, \phi_i^{(j)}\}$. In order to determine the $2(N + 1)$ coefficients a_i^j, $i = 0, 1, \ldots, N, j = 1, 2$, we use the following boundary conditions:

$$\pi_i^{(0)}(0) = 0, \quad i \in \mathcal{S}_U^{(0)} \tag{10}$$

$$\pi_i^{(0)}(B_l^{TD}) = \pi_i^{(1)}(B_l^{TD}), \quad i \in \mathcal{S}_D^{(0)} \cup \mathcal{S}_U^{(1)} \tag{11}$$

$$\pi_i^{(1)}(B) = p_i, \quad i \in \mathcal{S}_D^{(1)} \tag{12}$$

where $p_i = \binom{N}{i} \frac{\alpha^i \beta^{N-i}}{(\alpha+\beta)^N}, \quad 0 \le i \le N$ and the set $\mathcal{S}_U^{(j)}$ constitutes the set of all states in $\mathcal{S}$ for which the diagonal elements in the drift matrix $D^{(j)}$ are positive. Similarly $\mathcal{S}_D^{(j)}$ denotes the set of states for which the diagonal elements of the drift matrix are negative. We solve these numerically to determine a_i^j for $0 \le i \le N, j = 1, 2$. Next using the steady state distribution $\pi_i^j(x), j = 1, 2$, we determine $p_h^{TD}, p_l^{TD}, d_h^{TD}$ which are given by:

$$p_h^{TD} = \frac{\sum_{i=0}^{N} \{r + ir_c - c\}\{p_i - \pi_i^{(1)}(B)\}}{\sum_{i=0}^{N} ir_c p_i} \tag{13}$$

$$p_l^{TD} = \sum_{i=0}^{N} \{p_i - \pi_i^1(B_l^{TD})\} + \frac{\sum_{i \in \mathcal{S}_U^0 \cap \mathcal{S}_D^1} \{\pi_i^{(1)}(B_l^{TD}) - \pi_i^{(0)}(B_l^{TD})\}\{i + r - c\}}{r} \tag{14}$$

$$d_h^{TD} = \frac{1}{T^{(1)}} \left(\int_{t=0}^{\frac{B_l^{TD}}{c}} \sum_{i=0}^{N} i\pi_i^0(ct)dt + \int_{t=B_l^{TD}}^{B} \sum_{i=0}^{N} i\pi_i^{(1)}(ct)dt + \sum_{i=0}^{N} iP(\Sigma = i, Q = B_l^{TD}) + c\left[1 - \sum_{i=0}^{N} \pi_i^{(1)}(B)\right] \right) \tag{15}$$

where $T^{(1)} = \sum_{i=0}^{N} i\pi_i^{(1)}(B) + c\left[1 - \sum_{i=0}^{N} \pi_i^{(1)}(B)\right]$.

4.2 Fluid Analysis for Priority Scheduling with Edge-discarding

We next consider the fluid model for ED coupled with the PS router mechanism. We consider a service model that provides zero loss and zero delay guarantees to preferred packets with PS. In order to provide these guarantees, a peak-rate based admission control is required for the admission of sources transmitting preferred traffic [1].

Figure 5 illustrates the model of PS mechanism with bursty preferred traffic. Here $R_{arr}(t)$ denotes the aggregate preferred traffic at time $t > 0$. A key result in [8] states that each traffic class can be analyzed separately as follows. Consider the model shown in Figure 6 where the preferred traffic arrives into the first buffer. The aggregate of the departure process, $R_{dep}(t)$, from the first buffer and the non-preferred traffic arrival process, is the arrival process for the second buffer. The loss probability and expected delay of a preferred packet can be derived by considering an arrival process $R_{arr}(t)$ into a buffer of size B_h^{PS} and a server of capacity c in isolation. Similarly, the loss for non-preferred traffic can be derived by considering an arrival process $R_{dep}(t) + r(t)$ into a buffer of size B_l^{PS} and a server of capacity c separately.

In our case, $R_{arr}(t) < c$ because of the peak rate admission control for the preferred traffic; hence $R_{dep}(t) = R_{arr}(t)$. As a result, the loss probabilities of non-preferred traffic can be computed by analyzing a fluid model with an arrival rate $r(t) + R_{arr}(t)$ that is fed into a buffer of size B_l^{PS} and a server with

[1] This is a specific implementation of the one defined as expedited forwarding in [14]

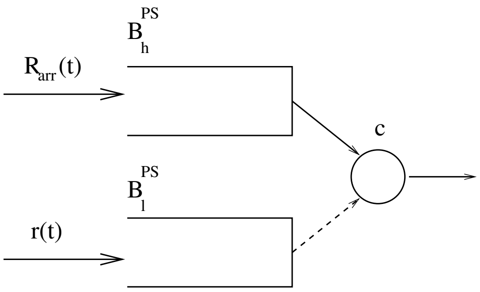


Figure 5: Fluid Model for Priority Scheduling

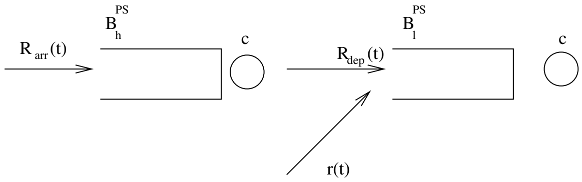


Figure 6: Equivalent Fluid Model for Priority Scheduling

capacity c, where $r(t)$ is the arrival rate of non-preferred traffic in the original model. Thus the analysis of PS mechanism in Figure 5 reduces to an analysis of a fluid model with a single class traffic [18]. Note that $B_l^{PS} = B$ as no buffer is required for preferred traffic with peak rate admission control.

4.3 Comparison of Results

Using the analytical models in Sections 4.1 and 4.2, we compare the services based on TD and PS for bursty arrivals. We shall observe:

- ED coupled with PS is able to provide better delay behavior to preferred traffic.
- TD and PS provide almost identical loss behavior to preferred traffic with one exception. When a source is *extremely bursty*, ED coupled with TD is a better service.

To simplify the choices in parameters, we use the following normalization: We chose the mean ON period as one time unit, during which one unit of information is generated. The mean Off period is $1/\lambda$, where $\lambda = \beta/\alpha$. Recall that λ represents the burstiness of a source, i.e., a smaller value of λ means the source is more bursty. For the numerical computations, we vary λ from 0.1 to 4.0. This varies the time spent in the On period from 10% to 75%.

Loss Comparison: The maximum number of preferred sources that can be supported with PS is N_p, where $N_p \le c \le N_p + 1$. Under PS, the admitted preferred traffic experiences no loss because of the peak-rate based admission control. It is not feasible to provide such a service with TD due to the interference between the preferred and non-preferred traffic. Instead, we allow preferred traffic to incur loss probability, $\tilde{p}_h$, under TD and examine the benefits of the resulting service. Specifically, we determine the number of preferred sources that can be supported such that non-preferred traffic incurs the same loss under both the mechanisms and $p_h^{TD} < \tilde{p}_h$.

Delay Comparison: The preferred traffic generated by these N_p sources experience zero delay with PS as there is no buffering for this class of traffic irrespective of non-preferred traffic. Next we determine the number of preferred sources that can be supported with TD for a given arrival rate of non-preferred traffic such that $d_h^{TD} < \tilde{d}_h$, and $p_h^{TD} < \tilde{p}_h$, where $\tilde{d}_h$ and $\tilde{p}_h$ denote the relaxations in expected delay and loss probability that we allow with TD for preferred traffic.

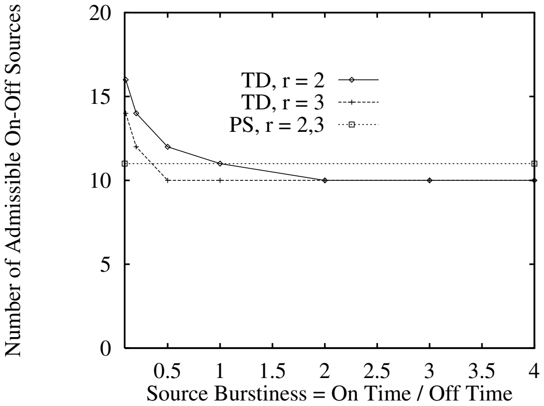


Figure 7: Loss Comparison as a function of Burstiness

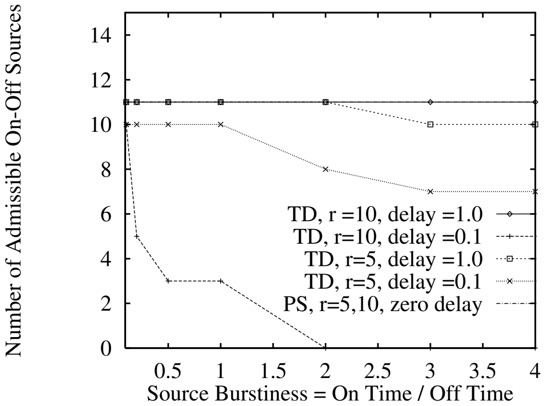


Figure 8: Delay Comparison as a function of Burstiness

Figure 7 compares the number of preferred sources that can be supported with TD and PS as a function of source burstiness, λ, when a very *small amount of loss* for preferred traffic is allowed with TD. We have chosen a server capacity, $c = 11.1$, and total buffer size of 15 data units. The number of preferred sources that can be supported with PS is 11 and is not affected by the source burstiness. The number of preferred sources that can be supported with TD is heavily influenced by the burstiness of preferred sources. First, we consider a non-preferred arrival rate $r = 2.0$. We observe that, by allowing a preferred traffic loss probability of 10^{-6}, 50% more preferred sources can be supported with TD than with PS when the sources are extremely bursty, i.e., $\lambda = 0.1$. But as the source burstiness decreases (i.e., λ increases), this difference decreases. For $\lambda > 1.0$, PS can support a greater number of sources than TD does when $r = 2.0$. To examine if these results differ when the non-preferred traffic is changed, we consider an arrival rate $r = 3.0$. We observe that PS can support a greater number of preferred sources for $\lambda > 0.4$. This indicates that PS can support a greater number of preferred sources except in the presence of extremely bursty sources. We observe similar results when we allow a loss probability of 10^{-4} for preferred traffic with TD.

Figure 8 compares the number of preferred sources that can be supported with TD and PS as a function of source burstiness when the preferred traffic is allowed to incur a small delay under TD, i.e., $\tilde{d}_h = 0.1, 1.0$. Note that PS can support 11 preferred sources providing zero delay irrespective of the non-preferred arrival rate. However, we observe that the number of sources that can be supported with TD depends on: (1) the relaxation in expected delay, $\tilde{d}_h$ that is allowed for the preferred traffic and, (2) the non-preferred arrival rate. When the non-preferred arrival rate, r, is 5.0, and $\tilde{d}_h = 1.0$, TD can support almost same number of sources as PS. But as the non-preferred arrival rate increases to 10, the number of sources that can be supported with TD decreases, although the reduction is not very dramatic. But if the delay relaxation, $\tilde{d}_h$, is reduced to 0.1, there is a sharp decrease in the number of sources that TD can support. For example, when $r = 10.0$ and $\tilde{d}_h = 0.1$, TD cannot support even a single source for $\lambda > 2.0$. *This result illustrates the superiority of PS for preferred sources with stringent delay requirements.*

Next we evaluate the excess bandwidth Δc that is necessary for TD to be able to support an identical number of sources as PS while providing almost zero delay $(d_h^{TD} < 10^{-3})$ to the preferred traffic. Figure 9 shows the excess bandwidth that is required with TD as a function of λ. We examined both $r = 5.0$ and $r = 10.0$ for the

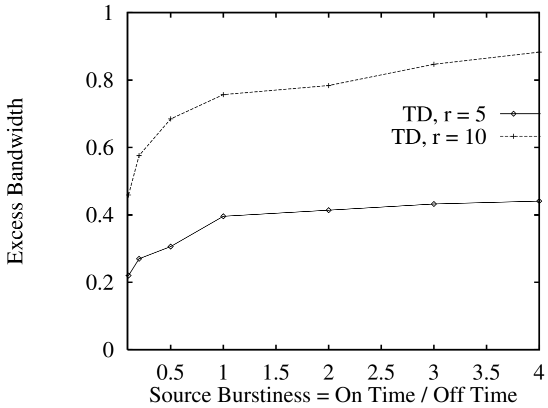


Figure 9: Excess Bandwidth for TD

illustration. We find that Δc is as high as 85% for $r = 10$.

Summarizing the results for the bursty sources needing strict loss and delay behaviors, we observe that PS is a better router mechanism for providing lower delays to preferred traffic when coupled with ED mechanism. We observe that as much as additional 85% bandwidth is necessary with TD to be able to provide similar delays as PS. Comparing the loss behavior of services built with TD to that of services of PS, we observe that when the sources are extremely bursty, it is possible to support a larger number of sources with TD if a small amount of preferred traffic loss is tolerated with TD.

5 Edge-marking with Threshold Dropping: TCP Source

We now turn our attention to examining the effect that the network service derived from a combination of edge marking (EM), and threshold dropping (TD) has on the throughput of a TCP connection. Due to edge-marking, packets belonging to the same flow may receive different marking, i.e., conforming packets marked as in-profile and the excess as out-profile. As a router inside a diffserv network provides services to packets based on this marking, packets of a same flow can incur different loss and delay. The behavior of TCP congestion control mechanism is not well understood in such a scenario where packets belonging to the same TCP session incur different loss and delay. We derive a Markov model for determining the throughput of a source using TCP Reno when edge-marking and threshold dropping are used in a diffserv architecture. We then examine the effect of forwarding out-profile packets using this Markov model.

5.1 Throughput of TCP with Edge-marking and Threshold Dropping

We describe a stochastic model for determining the throughput of a TCP Reno connection when packets use the network service derived from a combination of edge marking (EM), and threshold dropping (TD). We assume an infinite source model for TCP, i.e., the sender has enough data to keep the transmission active. We model the TCP connection in the following way. Let p_1 and p_2 denote the loss rates for in-profile and out-profile packets respectively. Let W_{max} denote the maximum TCP window size. We assume that a packet is marked in-profile if it is among the first W_a packets of the current window, and out-profile otherwise. We also assume that once a packet is lost, all subsequent packets in the same window are also lost. We refer to W_a as the assured window size, i.e., if the window size, $W(t)$ at time $t > 0$, is less than W_a, then all the packets are marked in-profile, and if $W(t) > W_a$, then the first W_a packets are marked in-profile, and the next $W(t) - W_a$ packets are marked out-profile.

We are interested in determining the throughput of this TCP connection as a function of W_a, W_{max}, p_1 and p_2. A recent work [22] models the behavior of TCP Reno as a Markov process for a single service, where all packets incur the same loss rate. This model has been shown to be reasonably accurate when compared to the measurements over the Internet [21] and with simulation [15]. We modify the above Markov model to account for the two regimes of packet loss that we have assumed for in-profile and out-profile packets. Each state Q_i in the Markov chain represents the current window size W_i, the number of packets that are lost in the previous round L_i, and whether or not the current state is as the result of a timeout. It can be shown easily that $\{Q_i(t)\}, t > 0$ is a Markov process. The details of the calculation of the transition probability matrix P, the number of packets that are transmitted by the source when it is in state i and the duration of this state for computing the throughput can be found in the Appendix. Solving $\pi P = \pi$, we derive the throughput as a function of p_1, p_2, W_a, W_{max}, round trip time R and the timeout period T.

We have described our model as if the service profile is specified as an assured window size W_a. If the service profile is specified as a maximum assured rate [5], R_a, then the following approximation can be used to map R_a into W_a, $W_a \approx R * R_a$. Thus, for a given assured rate R_a, the first W_a packets in the current window are marked as in-profile, and if the current window size $> W_a$, the rest of the packets in that window are marked out-profile.

5.2 TCP Throughput Results

Using the Markov model in Section 6.1, we examine the effect of edge-marking with threshold dropping on TCP throughput. We compute the throughput seen by a TCP connection as a function of assured window size, W_a, for a fixed maximum window size W_{max}; W_a varies between 0 and W_{max}. When $W_a = 0$, we refer to this as *non-assured TCP* which means that all packets are marked out-profile. When $W_a = W_{max}$, all of the packets of the TCP connection are sent as in-profile packets.

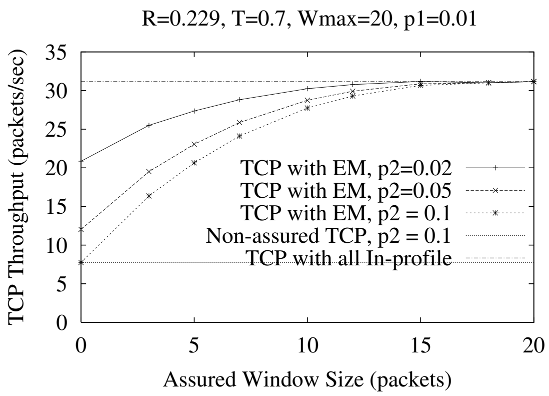


Figure 10: Throughput of TCP vs. Assured Window Size

Figure 10 shows the throughput (in packets/sec) as a function of the assured window size, W_a (in packets).

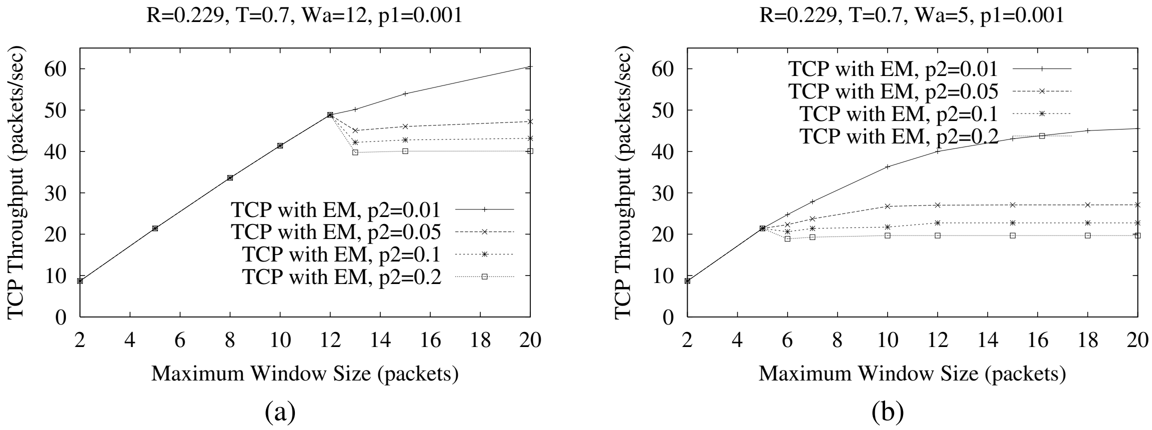


Figure 11: Throughput of TCP vs. Assured Window Size

The loss rates are $p_1 = 0.01$, $p_2 = 0.02, 0.05, 0.1$, W_{max} is 20, $R = 0.229$ and $T = 0.7$. [2] Consider the case when $p_1 = 0.01$ and $p_2 = 0.1$. For the case that $W_a = 0$, i.e., all packets are marked out-profile, the TCP throughput is 8 packets/sec. When $W_a = 20$, i.e., when all packets are marked in-profile, the achievable throughput is 31 packets/sec. We observe that the throughput is an increasing concave function of assured window size, W_a. In addition, we observe a non-linear relationship between throughput and W_a, i.e., most of the benefits are obtained for $W_a = 10$. Thus, the source benefits from using a small assured window size. For example, when $W_a = 5$, it is possible to increase the throughput by 200% over the non-assured TCP throughput. We observe similar trends when we choose different values for the out-profile packet loss probability. Comparing the case when $p_2 = 0.1$ with $p_2 = 0.05$, we observe that when p_2 is closer to p_1, the relative improvement in throughput decreases.

The above results show the benefits of assuring (even) a portion of a TCP flow compared to the use of non-assured TCP where all packets are sent as out-profile. We now study the behavior of TCP throughput when W_a is fixed and W_{max} is varied. Figure 11(a) shows the TCP throughput versus W_{max} for a fixed value of W_a. We choose $W_a = 12$, $p_1 = 0.001$, and the same values for R and T used earlier. We observe that throughput increases as W_{max} increases as long as $W_{max} \leq W_a$, irrespective of the value of p_2. This is because when $W_{max} \in (0, W_a)$, all packets are sent as in-profile packets. However, the throughput behaves very differently as a function of W_{max} when $W_{max} > W_a$, and is very sensitive to the loss probability of out-profile packets, p_2. For $p_2 = 0.01$, we observe an increase in throughput for $W_{max} > W_a$. However, when $p_2 = 0.05, 0.1$, or 0.2 we observe that TCP throughput decreases for $W_{max} > W_a$. Furthermore, this decrease in throughput increases with p_2. We recalculated the TCP throughput with $p_1 = 0.01$. We observe the similar trend with the exception that when $W_{max} > W_a$, we see a comparably smaller reduction in throughput when $p_2 = 0.05, 0.1$. Figure 11(b) illustrates the results for $W_a = 5$, a smaller assured window than the one used in Figure 11. In this case, except for $p_2 = 0.2$, the throughput increases even after $W_{max} > W_a$ for the range of values we have chosen.

The two key results of the above study are:

[2] We have investigated the above computation of throughput with several other measured values of R and T. We find that the qualitative conclusions of the results do not change.

- As shown in Figure 10, the throughput of a TCP source is significantly improved by sending a fraction of its packets, however small, marked in-profile. We also observe that there is a non-linear relationship between W_a and the observed throughput. The incremental gain in TCP throughput due to an increase in W_a decreases rapidly once W_a reaches a certain threshold. This threshold is dependent on p_1, p_2, and W_{max}. A sender might want to evaluate this threshold for W_a and specify a service profile accordingly.

- Figures 11(a) and 11(b) indicate that the maximum TCP window size should take into consideration the service profile, and the loss characteristics for in-profile and out-profile packets. They suggest that, once W_a is chosen, W_{max} should be chosen carefully so that the amount of out-profile packets does not exceed the range beyond which the throughput is affected.

While the first result indicates that there are *added benefits* in using TD with edge-marking, the second result requires the sender to *judiciously choose the fraction of packets* that should be sent as out-profile.

6 Conclusions

In this study, we quantitatively examined two key issues for a diffserv architecture: (1) how an internal router should treat packets of different classes, and (2) whether an edge-router should forward packets that fall outside of the service profile. In doing so, we derived analytical models to compare the loss and delay behavior that can be provided using services derived from the combination of two router mechanisms, *threshold dropping* and *priority scheduling* and two packet marking mechanisms, *edge-dropping* and *edge-marking*.

The first part of our study compared the loss and delay behavior of service classes derived by combining threshold dropping and priority scheduling with edge-discarding. Our examination of these services under a wide range of traffic models show that priority scheduling provides lower expected delays to preferred packets. In addition, we observed that a considerable additional link bandwidth is needed with threshold dropping to provide the same delay behavior as priority scheduling. We found that both router mechanisms provide similar loss rates to preferred packets with an exception for extremely bursty sources, in which case threshold dropping had better performance. These findings suggest that *priority scheduling should be an important mechanism* in a diffserv architecture for supporting delay sensitive applications.

The second part of our study examined the effect of forwarding out-profile packets into the network on the throughput of a TCP connection. We derived a simple Markov model for determining throughput of a TCP connection when in-profile packets observe different loss rates than out-profile packets. We found that it is possible to improve the throughput significantly even when a small portion of traffic is sent as in-profile packets. However, we observed that in order to fully utilize the benefit of out-profile packets, a TCP source must carefully determine the amount of out-profile packets it will send in addition to the in-profile packets. This result suggests that it is *beneficial to combine edge-marking with threshold dropping for throughput sensitive applications*. However, intra-session service differentiation can adversely affect the performance of a feedback-based network protocol if proper attention is not given to the choice of out-profile packets that a source should send.

References

[1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss. An Architecture for Differentiated Services, RFC-2475, December 1998.

[2] L. Braslau, S. Shenker. Is Service Priority Useful in Networks?, *Proceedings of ACM Sigmetrics*, June 1998, pp. 66-77.

[3] L. Braslau, S. Shenker. Best-Effort versus Reservations: A Simple Comparative Analysis, *Proceedings of ACM Sigcomm*, October 1998, pp. 3-16.

[4] D.D. Clark, J. Wroclaski. An Approach to Service Allocation in the Internet, IETF Draft, July 1997.

[5] D.D. Clark, and W. Fang. Explicit Allocation of Best-Effort Packet Delivery Service, *ACM Transaction on Networking,* Vol. 6, Number 4, August 1998, pp. 362-373.

[6] S. Deering. Communication on Diffserv Mailing List.

[7] A. Elwalid, D. Mitra. Analysis and Design of Rate-Based Congestion Control of High Speed Networks, I: Stochastic Fluid Models, Access Regulation, *Queuing Systems*, 9, pp. 29-64, 1991.

[8] A. Elwalid, D. Mitra. Fluid Models for the Analysis and Design of Statistical Multiplexing With Loss Priorities on Multiple Classes of Bursty Traffics, *IEEE INFOCOM 92*, pp. 415-425.

[9] A. Elwalid, D. Mitra. Analysis, Approximations and Admission Control of a Multi-Service Multiplexing System with Priorities, *IEEE Infocom 95*, pp. 463-472.

[10] W. Feng, D.D. Kandlur, D. Saha, and K.G. Shin. Adaptive Packet Marking for Providing Differentiated Services in the Internet, IBM Research Report, June 1997.

[11] S .Floyd. Communication in Diffserv Mailing List, Aug. 1998.

[12] R. Guerin, S. Kamat, and R. Rajan. Scalable QoS Provision Through Buffer Management, *Proceedings of ACM Sigcomm*, October 1998, pp. 29-40.

[13] J. Ibanez, and K. Nichols. Preliminary Simulation Evaluation of an Assured Service, IETF Draft, August 1998.

[14] V. Jacobson, K. Nichols, and K. Poduri. An Expedited Forwarding PHB, IETF Draft, November 1998.

[15] M. Handley, S. Floyd. TCP-Friendly Simulations, Unpublished Work.

[16] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured Forwarding PHB Group, IETF Draft, January 1999.

[17] M. May, J.-C. Bolot, C. Diot, and A. Jean-Marie. Simple Performance Models of Differentiated Services Schemes for the Internet, *Proceedings of IEEE Infocom'99*, March 1999, pp. xx-yy.

[18] D. Mitra. Stochastic Theory of a Fluid Model of Producers and Consumers Coupled by a Buffer, *Adv. App. Prob.*, 20, pp. 646-676, 1988.

[19] H. Naser, and O. Aboul-Magd. Voice over Differentiated Services, *IETF Draft*, 1999.

[20] K. Nichols, V. Jacobson, L. Zhang. A Two-bit Differentiated Services Architecture for the Internet, *IETF Draft*, Nov. 1997.

[21] J. Padhye, V. Firoiu, D. Towsley, J. Kurose. Modeling TCP Throughput: A Simple Model and its Empirical Validation, *Proceedings of ACM Sigcomm*, October 1998, pp. 303-314.

[22] J. Padhye, V. Firoiu, D. Towsley. A Stochastic Model of TCP Reno Congestion Avoidance and Control, Computer Science Tech-report-99-02, Univ. of Massachusetts, January 1999, available from: `ftp://gaia.cs.umass.edu/pub/Padhey99-markov.ps`.

[23] S. Sahu, D. Towsley, J. Kurose. A Quantitative Study of Differentiated Services for the Internet, Computer Science Tech-report-99-09, Univ. of Massachusetts, February 1999, available from: `ftp://gaia.cs.umass.edu/pub/Sahu99_Diffserv-TR-09.ps`.

[24] I. Stoica, S. Shenker, and H. Zhang. Core-Stateless Fair Queuing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks, *Proceedings of ACM Sigcomm 98*, October 1998, pp. 118-130.

[25] I. Stoica, H. Zhang. LIRA: An Approach for Service Differentiation in the Internet, *Proceedings of Nossdav*, June 1998.

[26] J. Wroclawski. The Use of RSVP with IETF Integrated Services, RFC 2210, September 1997.

[27] I. Yeom, and A.L.N. Reddy. Realizing Throughput Guarantees in a Differentiated Services Network, In Submission.

A Markov Model for TCP Reno with Two Loss Regimes

A recent work [22] models TCP Reno as a Markov process for a single service, where all packets incur the same loss rate. This model [22] improves the approximate analysis in [21] via a Markov process. We modify this Markov model to account for two loss regimes that packets of a TCP connection incur when threshold dropping is used in conjunction with edge marking. Note that the in-profile and out-profile packets incur different loss rates. The first W_a packets of a window are marked in-profile and the excess as out-profile. We also assume that once a packet gets lost, all packets that follow it in the current window also get lost. In addition, we assume delayed ack model [22] for the TCP implementation, i.e., one acknowledgment is sent to the sender for every two correctly received packets. First we start with the formal definitions of various random variables that are needed.

- $\{W_i\}_{i=0}^{\infty}, \quad 1 \leq W_i \leq W_{max}$.
  W_i denotes the sequence of window sizes of the round.

- $\{C_i\}_{i=0}^{\infty}, \quad C_i = 0, 1$.
  C_i allows us to model the increment of the window by one every two rounds during no-loss period. $C_i = 0$ indicates the first of these two rounds and $C_i = 1$ indicates the second. We impose the following constraint on the value of C_i: If $W_i = W_{max}$, then $C_i = 0$. This captures the fact that the window size cannot grow beyond W_{max}.

- $\{L_i\}_{i=0}^{\infty}, \quad 0 \leq L_i < W_{i-1}$.
  L_i is the number of packets lost in the previous round. $L_i = 0$ indicates either that no packet was lost in the previous round, or that the previous round was a short round and the effects of packet loss have already been accounted for via a triple duplicate event or a timeout event. $L_i > 0$ indicates that this round is a short round, and is a result of L_i packets being lost in the previous round. For short rounds, C_i is always equal to 0. In other words, if $L_i > 0$, then $C_i = 0$.

- $\{T_i\}_{i=0}^{\infty}, \quad 0 \leq T_i \leq 7$.
  T_i denotes the timeouts. $T_i = 0$ if transmission of packets in this round was not a result of a retransmission timer expiration. In other words, $T_i = 0$ if a timeout did not occur as a result of packet loss in the previous round. Moreover, $T_i = 1, \ldots, 7$, if a timeout did occur in the previous round, and the back-off value was $1, 2, 4, \ldots, 64$, respectively.

- $\{R_i\}_{i=0}^{\infty}, \quad RTT \leq R_i \leq 64 * T_0$.
  R_i denotes the duration of each round. RTT denotes the round trip time, and T_0 denotes the base timeout value. During a timeout sequence R_i can have value of $T_0, 2T_0, \ldots 64 * T_0$, as explained earlier. All other rounds have a length of RTT.

- $\{N_i\}_{i=0}^{\infty}, 1 \leq N_i \leq W_{max}$.
  N_i denotes the number of packets transmitted in round i.

The sequence of random variables, $\{(W_i, C_i, L_i, T_i)\}_{i=0}^{\infty}$ is a finite state Markov chain with probability transition matrix, $P = [q_{v,a,k,s;w,c,l,t}]$, which we will soon calculate.

Let $r = (v, a, k, s)$ and $r' = (w, c, l, t)$ be the indices of two successive states in the sequence $\{(W_i, C_i, L_i, T_i)\}_{i=0}^{\infty}$. The number of packets sent during this transition is denoted by N_r and the time taken for the transition is denoted by R_r. Define the conditional expectations:

$$n_{v,a,k,s;w,c,l,t} = E[N_r | W_r = v, C_r = a, L_r = k, T_r = s, W_{r'} = w, C_{r'} = c, L_{r'} = l, T_{r'} = t]$$

$$x_{v,a,k,s;w,c,l,t} = E[R_r | W_r = v, C_r = a, L_r = k, T_r = s, W_{r'} = w, C_{r'} = c, L_{r'} = l, T_{r'} = t]$$

These conditional expectations, along with the transition probabilities $q_{v,a,k,s;w,c,l,t}$, capture the behavior of TCP's congestion control and avoidance algorithm. Define:

$$N = \lim_{i \to \infty} N_i \tag{16}$$

$$R = \lim_{i \to \infty} R_i \tag{17}$$

For given time $t > 0$, define N_t to be the number of packets transmitted in interval $[0, t]$, and M_t to be the number of packets that make it to the receiver. Then the throughput during that interval is given by N_t/t.

The long term throughput of a TCP connection is then defined by:

$$\begin{aligned}
\text{Throughput} &= \lim_{t \to \infty} \frac{N_t}{t} \\
&= \frac{E[N]}{E[R]} \\
&= \frac{\sum_{(v,a,k,s) \in S} \pi_{v,a,k,s} \sum_{(w,l,c,t) \in S} q_{v,a,k,s;w,c,l,t} n_{v,a,k,s;w,c,l,t}}{\sum_{(v,a,k,s) \in S} \pi_{v,a,k,s} \sum_{(w,l,c,t) \in S} q_{v,a,k,s;w,c,l,t} r_{v,a,k,s;w,c,l,t}}
\end{aligned} \tag{18}$$

We now calculate the transition probabilities and the conditional expectations of packets sent in each round and time required to complete the round.

A.0.1 No packets are lost

When no packets are lost, the TCP window size increases by one every two round trip times. During each such round, a window worth of packets are sent in one round trip time. Therefore:

$$\begin{array}{lcll}
q_{w,0,0,0;w,1,0,0} & = & (1-p_1)^w, & 1 \le w < W_a \\
 & = & (1-p_1)^{W_a}(1-p_2)^{w-W_a}, & W_a + 1 \le w < W_{max} \\
q_{w,1,0,0;w+1,0,0,0} & = & (1-p_1)^w, & 1 \le w < W_a \\
 & = & (1-p_1)^{W_a}(1-p_2)^{w-W_a}, & W_a + 1 \le w < W_{max} \\
q_{w,0,0,0;w,0,0,0} & = & (1-p_1)^{W_a}(1-p_2)^{w-W_a}, & w = W_{max} \\
 & & & \\
n_{w,0,0,0;w,1,0,0} & = & w, & 1 \le w < W_{max} \\
n_{w,1,0,0;w+1,0,0,0} & = & w, & 1 \le w < W_{max} \\
n_{w,0,0,0;w,0,0,0} & = & w, & w = W_{max} \\
 & & & \\
x_{w,0,0,0;w,1,0,0} & = & RTT, & 1 \le w < W_{max} \\
x_{w,1,0,0;w+1,0,0,0} & = & RTT, & 1 \le w < W_{max} \\
x_{w,0,0,0;w,0,0,0} & = & RTT, & w = W_{max}
\end{array} \tag{19}$$

A.0.2 One or more packets are lost in a round

When one or more, but not all packets are lost in a round, the following round is referred to as a "short round" [22]. This transition is accompanied by sending of a window worth of packets in a round trip time. When all packets in a window are lost, there is no following "short" round, instead, the sender waits for a retransmission timeout, and the window size goes to one.

$$
\begin{array}{llllll}
q_{w,c,0,0;w,0,l,0} & = & p_1(1-p_1)^{w-l}, & 2 \le w, w-l \le W_a, & c=0,1, & 1 \le l < w \\
 & = & p_2(1-p_1)^{W_a}(1-p_2)^{w-W_a-l}, & 2 \le w, W_a \le w-l \le W_{max}, & c=0,1, & 1 \le l < w \\
q_{w,c,0,0;1,0,0,1} & = & p_1, & 1 \le w \le W_a, & c=0,1 & \\
 & = & p_2, & W_a \le w \le W_{max}, & c=0,1 & \\
 & & & & & \\
n_{w,c,0,0;w-l,0,l,0} & = & w, & 2 \le w \le W_{max}, & c=0,1, & 1 \le l < w \\
n_{w,c,0,0;1,0,0,1} & = & w, & 1 \le w \le W_{max}, & c=0,1 & \\
 & & & & & \\
 & c=0,1 & & & & \\
 & & & & & \\
x_{w,c,0,0;w-l,0,l,0} & = & RTT, & 2 \le w \le W_{max}, & c=0,1, & 1 \le l < w \\
x_{w,c,0,0;1,0,0,1} & = & TO, & 1 \le w \le W_{max}, & c=0,1 &
\end{array}
\tag{20}
$$

A.0.3 One or more packets are lost in a short round

The following are the transition probabilities when one or more packets are lost in a short round.

$$
\begin{array}{lll}
q_{w,0,l,0;1,0,0,1} & = 1, & 1 \le w < 3 \\
q_{w,0,l,0;1,0,0,1} & = \sum_{i=0}^{2} p_1(1-p_1)^i, & 3 \le w < W_{max} \\
q_{w,0,l,0;\lfloor (w+l)/2 \rfloor,0,0,0} & = \sum_{i=3}^{w-1} p_1(1-p_1)^i + (1-p_1)^w, & 3 \le w, w-l < W_a \\
 & & \\
q_{w,0,l,0;\lfloor (w+l)/2 \rfloor,0,0,0} & = \sum_{i=3}^{W_a} p_1(1-p_1)^i + \sum_{i=W_a+1}^{w-1} p_2(1-p_1)^{W_a}(1-p_2)^{i-W_a} & \\
 & \quad +(1-p_1)^{W_a}(1-p_2)^{w-W_a}, & 3 \le w, W_a \le w-l < W_{max} \\
 & & \\
n_{w,0,l,0;1,0,0,1} & = w, & 1 \le w < 3 \\
n_{w,0,l,0;1,0,0,1} & = w, & 3 \le w+l < W_{max} \\
n_{w,0,l,0;\lfloor (w+l)/2 \rfloor,0,0,0} & = w, & 3 \le w < W_{max} \\
 & & \\
x_{w,0,l,0;1,0,0,1} & = T_0 - RTT, & 1 \le w < 3 \\
x_{w,0,l,0;1,0,0,1} & = T_0 - RTT, & 3 \le w < W_{max} \\
x_{w,0,l,0;\lfloor (w+l)/2 \rfloor,0,0,0} & = RTT, & 3 \le w < W_{max}
\end{array}
\tag{21}
$$

A.0.4 Exponential Backoff

If a packet is retransmitted as a result of a timeout, the retransmission timer backs off exponentially, but not exceeding 64 times the base timeout value. The timeout sequence ends when the retransmitted packet is successfully ACK'd.

$$
\begin{array}{lcll}
q_{1,0,0,i;1,0,0,\min(i+1,7)} & = & p_1, & 1 \le i \le 7 \\
q_{1,0,0,i;2,0,0,0} & = & 1-p_1, & 1 \le i \le 7 \\
\\
n_{1,0,0,i;1,0,0,\min(i+1,7)} & = & 1, & 1 \le i \le 7 \\
n_{1,0,0,i;2,0,0,0} & = & 1, & 1 \le i \le 7 \\
\\
x_{1,0,0,i;1,0,0,\min(i+1,7)} & = & 2^{\min(6,i-1)}T_0, & 1 \le i \le 7 \\
x_{1,0,0,i;2,0,0,0} & = & RTT, & 1 \le i \le 7
\end{array}
\qquad (22)
$$

Using the above transition probability matrix, we solve for $\pi P = \pi$ to compute the throughput of a TCP connection with two loss regimes.