Analysis of Shared Memory Priority Queues with Two Discard Levels

Shlomi Bergida and Yuval Shavitt, Tel Aviv University

Abstract

Two-rate SLAs become increasingly popular in today's Internet, allowing a customer to save money by paying one price for committed traffic and a much lower price for additional traffic that is not guaranteed. These types of SLAs are suggested for all types of traffic from best effort to QoS constrained applications. In access networks, where these SLAs are prevalent, shared memory switches are a common feature of architecture. Thus, dimensioning and management of shared memory queues for multiple priorities, each with two levels of guarantees, becomes an interesting challenge. We present a simple analysis of a multipriority multi-discard-level system controlled by a buffer occupancy threshold policy aimed at assuring SLA compliance for conforming (i.e., committed) traffic, and performance maximization for nonconforming (i.e., excess) traffic. Our analysis shows how to calculate the different system parameters: total buffer size, threshold position, and offered load control performance for the committed and excess traffic. Our suggested design enables assuring high SLA compliance for conforming traffic and performance maximization for nonconforming traffic.

In the ongoing work aimed at finding the way to transform the Internet from a single-class best effort service to providing a variety of service classes offering different performance guarantees (quality of service, QoS), simple coarse schemes and lightweight hardware support have become popular. Some such schemes are based on the concept of classification and performance level assignments at the edge of the networks. Packets are marked or tagged accordingly, and this marking is used to apply differentiated handling of the packets in the core of the network. These ideas took form in the extensive work of the differentiated services (DiffServ) working group of the Internet Engineering Task Force (IETF) [1, 2]. They were later also incorporated into the multiprotocol label switching (MPLS) world in the form of MPLS DiffServ with traffic engineering (TE) technology [3], and recently introduced into the metro Ethernet world, with the standardization efforts of the Metro Ethernet Forum [4].

A typical contract between a customer and a provider is stated in terms of a service level agreement (SLA). In its simplest form it ensures the customer a minimum or expected bandwidth for its usage and may allow additional (excess) bandwidth to be used based on availability.

We examine a typical case where several classes of services are defined. Customers requiring high performance (e.g., low delay and loss as defined in their SLAs) are assigned to the high priority class. Other customers are assigned to the lower priority classes with lower performance. The packets of a given class that conform to the agreed assured bandwidth are termed in this work *committed* bandwidth traffic of that class, and the packets that do not conform are termed *excess* traffic (these are sometimes termed *in* and *out* packets, respectively).

Typically at the ingress of the network, the provider monitors each class of traffic and marks the packets that exceed the committed rate as excess. The provider assures negligible drop probability for the committed traffic (of all classes) even during congestion periods by dropping excess traffic with higher probability. Specifically, during congestion it is preferable to drop excess traffic of high priority to dropping low-priority committed traffic. Mechanisms like Core-Stateless Fair Queuing [5] suggest using finer classification at the edge (instead of the two classes we use here) and thus allowing a fairer drop policy at the core.

Queue management has been studied extensively [6], and complete memory sharing among all classes has been shown to provide optimal throughput-delay performance and maximum utilization of available memory in the system [7]. There is a long line of work that examines threshold policies for two (or more) types of packets that share a single first in first out (FIFO) buffer [8, 9]; these deal with either a single class of packets, some which are marked as discard eligible (e.g., non-rate-conforming), or multiple classes of packets sharing a single FIFO buffer. In this article we consider, for the first time, the case of multiple priority classes of packets each having two discard levels: committed and excess packets.

The system proposed in this article can be considered a simple low-cost fast switch supporting coarse QoS differentiation. The system is based on a single shared memory space accommodating multiple FIFO queues (one per priority class). Packets are serviced according to a strict priority scheduling policy. A simple total occupancy-threshold policy is used for buffer management [8, Sec. 3.3].

We wish to analyze and study the behavior of such a system and provide guidelines for setting optimal system parameters (thresholds and buffer sizes) given traffic conditions. Our goal is to satisfy the requirements of the SLA defined for the committed traffic (i.e., negligible drop probability and adequate delay for each priority class) while maximizing utilization of

0890-8044/07/$20.00 © 2007 IEEE

available excess bandwidth to serve revenue generating excess traffic. This is to be achieved with minimal memory requirements.

To this end we use a two-priority-queue model (Fig. 1).

- Priority queue 1 (high priority) serves two traffic types, committed and excess. The excess traffic is managed by means of a threshold, α_{1E}, which inhibits high-priority excess traffic acceptance based on total buffer space occupancy (by all priorities and discard levels).
- Priority queue 0 (low priority) has two traffic types, committed and excess. The threshold, α_{0E}, has the same meaning as that defined for priority 1 traffic. Service is non-preemptive. Extensions to allow more priorities are omitted due to space limitations.

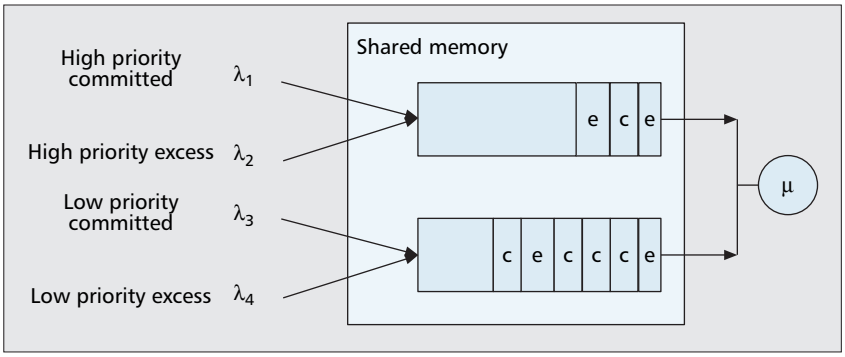

Figure 1. *System model.*

To allow simple and efficient analysis and calculation, we present an analysis that uses a simpler model, where the high-priority queue is presented with both excess and committed traffic, and the low-priority traffic is presented with committed traffic only. Thus, we have three packet types: high priority committed, high priority excess, and low priority committed. Second, we use Poisson arrival processes to model all incoming traffic types. Third, we deal with the finite nature of our queue in our model only to the extent needed to analyze committed traffic loss. Thus, in part of our analysis we assume that the headroom (i.e., the buffer space above the threshold) is infinite. This assumption is based on two facts. First, the marking process employed at the network ingress controls the committed traffic rate and characteristics. Second, the system design process is aimed at avoiding committed traffic loss. Indeed, we show that a system designed this way has a quickly dropping buffer occupancy distribution function above the threshold. This allows for the infinite headroom assumption given that the actual headroom allocated is large enough. Generalizations of the system, doing without the above mentioned simplifications, are addressed in the full version of this work [10].

Exact Analysis

The System Model

Two queues share a buffer space of n packets (or cells). The high-priority queue serves committed traffic and excess traffic packet arrivals modeled by a Poisson process of rates λ_1 and λ_2, respectively. The low-priority queue serves committed traffic packet arrivals, also modeled as a Poisson process at rate λ_3. Service rate is μ (Fig. 1). The threshold is denoted $n_{th} = \alpha_{1E}n$. When the total occupancy of the buffer is above this threshold, excess high-priority traffic is rejected and lost.

An exact analysis of the above system can be done by using a continuous-time two-dimensional Markov chain with $(n + 1)(n + 2)/2$ states, where each state is represented by the ordered pair (t, s), and t is the number of high-priority packets in the buffer and s the number of low-priority packets. The system has $O(n^2)$ states, and can be solved in time complexity of $O(n^5)$ using standard tools yielding the delay, buffer occupancy, and throughput of each traffic class. Using smart recurrence [6, Sec. III.B] we can reduce the computation complexity to $O(n^3)$.

Approximated Analysis

The complexity of the exact numerical solution may be too high to allow solving the system for buffer sizes exceeding a few tens of packets. Although one can use numerical methods, such as stochastic petri nets [12], to model and solve such large systems efficiently, they do not produce compact closed form expressions that allow us to understand how the system parameters affect its behavior. Thus, we suggest instead to first model the system by a single parameter, its total occupancy, as explained below. Using occupancy distribution we derive the other system parameters with a finer analysis. This solution is shown to have negligible error but produces compact and easy-to-understand expressions for the system behavior.

Analysis of Total System Occupancy

Thus, we start by looking at the one-dimensional state space representing the total occupancy of the shared memory buffer: the number of packets (of all types) present in the system at a given moment.

In this analysis the system can be modeled by a continuous-time birth-death Markov chain with $n + 1$ states. The state transition probabilities are given by

$$q_{u,u+1} = \begin{cases} \lambda_1 + \lambda_2 + \lambda_3 & \text{if } u \le n_{th} \\ \lambda_1 + \lambda_3 & \text{if } n_{th} < u < n \\ 0 & \text{if } u = n \end{cases}$$
$$q_{u,u-1} = \begin{cases} u & \text{if } 0 < u \le n \\ 0 & \text{if } u = 0 \end{cases}. \quad (1)$$

To find the steady state probabilities, π_u, we solve the system equilibrium equations, together with the probability conservation relation:

$$\pi_u = \begin{cases} \pi_0(\rho_u)^u & \text{if } u \le n_{th}+1 \\ \pi_{(n_{th}+1)}(\rho_r)^{(u-n_{th}-1)} & \text{if } n_{th}+1 < u \le n \end{cases} \quad (2)$$

$$\pi_0 = \frac{(1-\rho_u)(1-\rho_r)}{(1-\rho_r)(1-\rho_u^{n_{th}+2}) + \rho_u^{n_{th}+1}\rho_r(1-\rho_u)(1-\rho_u^{n-n_{th}-1})}, \quad (3)$$

where ρ_u, the unrestricted full load case, and ρ_r, the restricted load case (when occupancy is over the threshold), are defined as

$$\rho_u = (\lambda_1 + \lambda_2 + \lambda_3)/\mu, \quad (4)$$
$$\rho_r = (\lambda_1 + \lambda_3)/\mu.$$

The drop probability for the committed traffic, η_1 (high priority) and η_3 (low priority), is the probability the system is full, or finding n packets in the buffer. The drop probability for the excess traffic, η_2, is the probability that more than n_{th} packets are in the buffer. Substituting the expressions from Eq. 2 we obtain

$$\eta_1 = \pi_n = \pi_0\rho_u^{n_{th}+1}\rho_r^{n-n_{th}-1}$$
$$\eta_2 = \pi_0\rho_u^{n_{th}+1}\frac{1-\rho_r^{n-n_{th}}}{1-\rho_r} \quad (5)$$
$$\eta_3 = \eta_1.$$

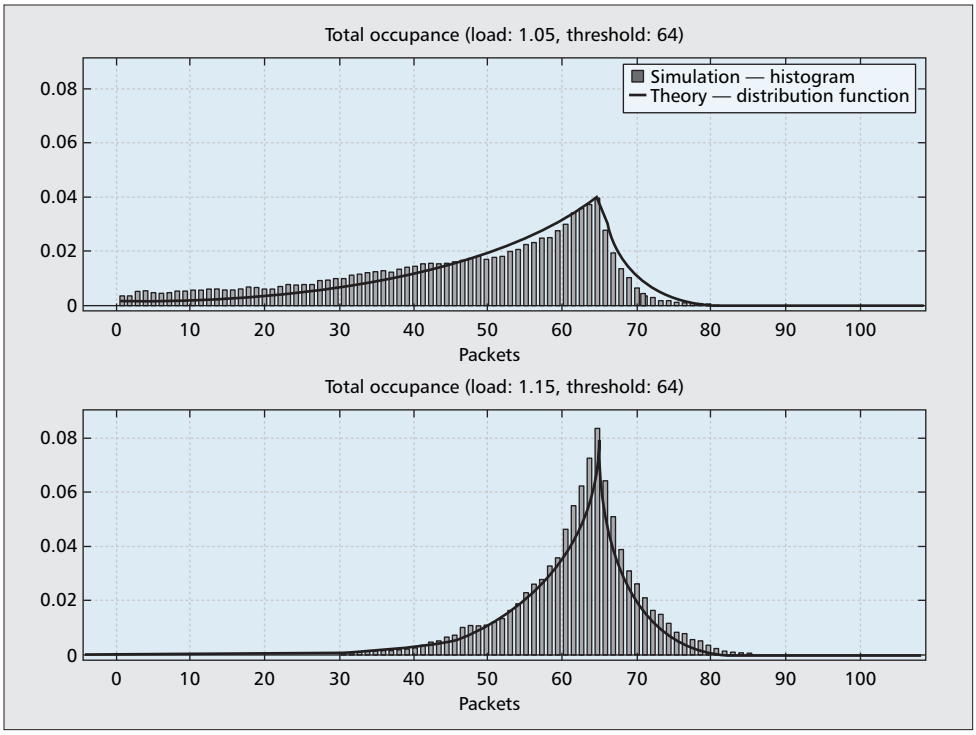

Figure 2. *Total system occupancy distribution function at two load points.*

We are interested in the regime where $\rho_r = (\lambda_1 + \lambda_3)/\mu < 1$ or else committed traffic will be lost with probability 1. Furthermore, we look mainly at the case when the total load, $(\lambda_1 + \lambda_2 + \lambda_3)/\mu$, exceeds unity as it represents periods of congestion.

Figure 2 shows total occupancy distribution for two load points of 1.05 and 1.15 (unless otherwise specified, the system load is defined as the aggregate load of all packet types. Also, unless otherwise specified the rate is equal for all types (i.e., $\lambda_1 = \lambda_2 = \lambda_3$). This figure demonstrates that in the cases of interest the total occupancy probability distribution drops fast above the threshold. This allows the infinite buffer assumption, given that the actual space above the threshold is large enough.

Figure 3 shows the acceptance ratio of the various packet types as a function of the threshold value at the same two load values (committed traffic in these figures is not lost according to our infinite capacity assumption). Both graphs include simulation results for comparison.

Delay Analysis

For our approximated analysis of the delay in the threshold governed priority queue, we use an approach based on the multipriority with infinite capacity analysis of Kleinrock [13].

We now claim that under the above assumptions we can approximate the average waiting time of the high-priority queue in our system using the results from the infinite case presented above.

The waiting time equation for priority 1 for our case is

$$W_1 = \hat{W}_0 + \bar{x}_1 N_1.$$

Substituting $\hat{\lambda}_1 W_1$ for N_1 by Little's theorem and rearranging yields:

$$W_1 = \frac{\hat{W}_0}{1 - \bar{x}_1 \hat{\lambda}_1}. \tag{6}$$

We adapt this result to the model under consideration by recalculating $\hat{\lambda}_i$ and $\hat{W}_0$ using the steady state distribution of the total system occupancy obtained above. In our case (exponential service at rate μ for all priorities) both $\bar{x}_i$ and $\bar{x}_i^2/2\bar{x}_i$ reduce to $1/\mu$ for every i.[1] The chance of the server being free is $1 - \pi_0$ (Eq. 3). Therefore, $\hat{W}_0$ can be written as

$$\hat{W}_0 = \frac{1}{\mu}(1 - \pi_0). \tag{7}$$

Since we consider infinite total capacity, the total system occupancy distribution function (Eq. 2) is

$$\begin{aligned} \pi_u &= \pi_0(\rho_u)^u && \text{if } u \le n_{th} + 1 \\ \pi_u &= \pi_{n_{th}}(\rho_r)^{(u - n_{th})} && \text{if } n_{th} + 1 < u, \end{aligned} \tag{8}$$

where

$$\pi_0 = \frac{(1-\rho_u)(1-\rho_r)}{(1-\rho_r)(1-\rho_u^{n_{th}+2}) + \rho_u^{n_{th}+1}\rho_r(1-\rho_u)}. \tag{9}$$

The average input rate for the high-priority queue can now be calculated as

$$\begin{aligned} \hat{\lambda}_1 = {} & (\lambda_1 + \lambda_2) \cdot P(occupancy \le n_{th}) \\ & + \lambda_1 \cdot P(n_{th} < occupancy), \end{aligned} \tag{10}$$

yielding

$$\hat{\lambda}_1 = \pi_0 \left[\frac{1 - \rho_u^{n_{th}+1}}{1 - \rho_u}(\lambda_1 + \lambda_2) + \frac{\rho_u^{n_{th}+1}}{1 - \rho_r}\lambda_1 \right]; \tag{11}$$

thus, we have

$$W_1 = \frac{\hat{W}_0}{1 - \frac{1}{\mu}\hat{\lambda}_1}. \tag{12}$$

Using Eqs. 7, 9, 11, and 12 together with Little's theorem, we can calculate the waiting time of the low-priority queue.

[1] *Clearly, the model allows for traffic in each priority to have different stochastic characteristics.*

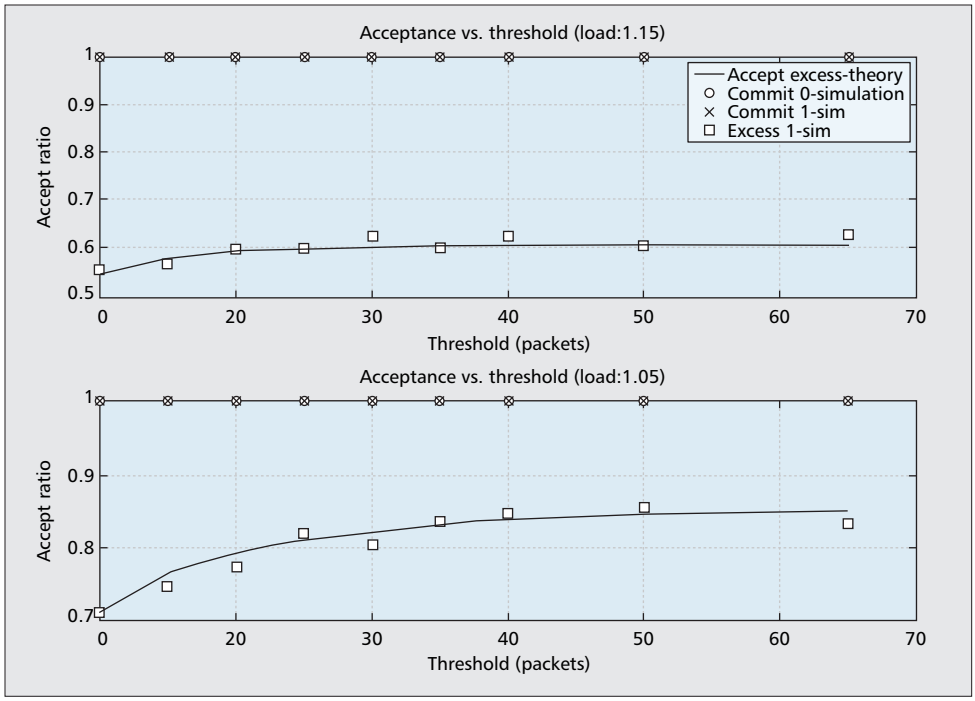

Figure 3. *Acceptance vs. threshold at two load points. The buffer size is large enough so that committed traffic loss probability is negligible.*

The average occupancy of the low priority queue is

$$N_0 = \hat{\lambda}_0 W_0 = N - N_1. \quad (13)$$

$N_1 = W_1 \hat{\lambda}_1$ and N can be calculated using the results of the total system occupancy distribution (Eq. 8), yielding

$$N = \frac{\pi_0}{(1-\rho_u)^2}\left[\rho_u^{n_{th}+2}((n_{th}+1)\rho_u - (n_{th}+2)) + \rho_u\right] - \frac{\pi_0}{(1-\rho_r)^2}\left[\rho_r\rho_u^{n_{th}+1}((n_{th}+1)\rho_r - (n_{th}+2))\right]. \quad (14)$$

$\hat{\lambda}_0$ in this case is equivalent to λ_3; thus,

$$W_0 = \frac{N - W_1\hat{\lambda}_1}{\lambda_3}. \quad (15)$$

In Fig. 4 the expected delay is shown (theoretical calculations and corresponding simulated results) as a function of the aggregate load for all priorities. The figure shows that our analysis agrees with the simulation we conducted.

System Behavior and Trade-offs

Using the above results, we can now study the system behavior and trade-offs presented as a function of load and threshold selection. We are interested in the regime where the aggregate input rate is close to the service rate (i.e., time of congestion). We assume that the committed traffic is allocated enough resources to keep loss low (namely, the committed aggregate input rate, $\lambda_1 + \lambda_3$, is lower than the service rate, and adequate buffer space above the threshold, $n - n_{th}$, is allocated). This is a logical common policy. Low loss can be verified by checking that expected committed traffic loss ratios (η_1, or equivalently η_3, of Eq. 5) are negligible.

Figures 5, 6, and 7 show the effect of different load and threshold values on the service level obtained for the committed

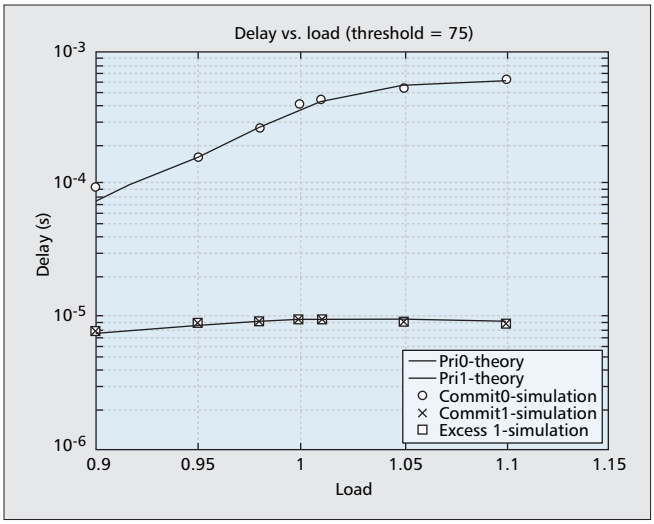

Figure 4. *Delay vs. load.*

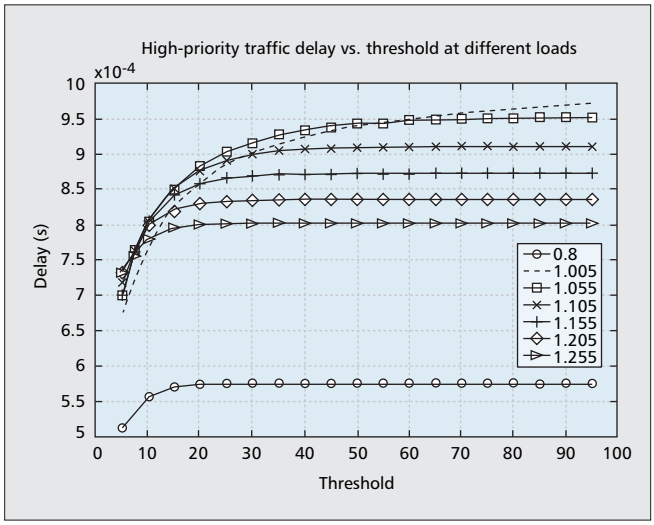

Figure 5. *High priority delay.*

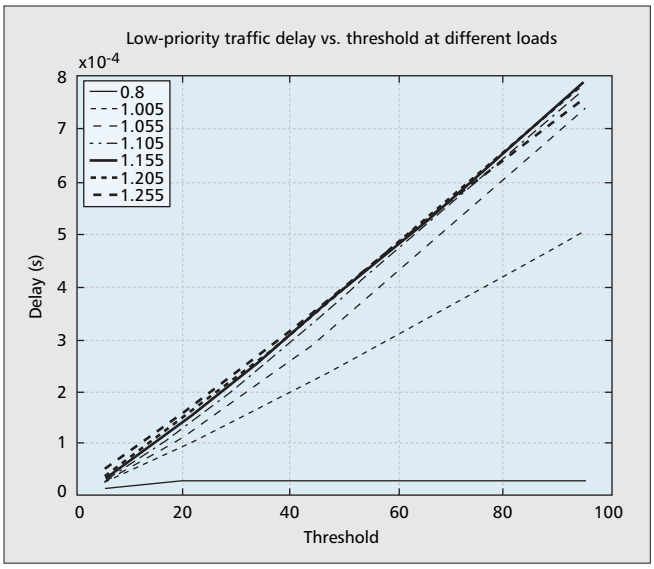

Figure 6. *Low-priority delay.*

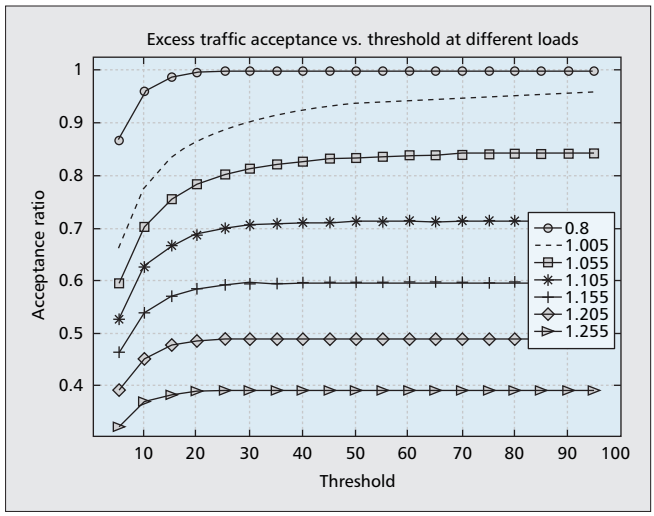

Figure 7. *Excess acceptance.*

and excess traffic of both priorities. Looking first at high priority (committed and excess) traffic delay (Fig. 5) we observe that the strict priority service scheme promises low delay that is affected by the threshold only when it becomes too low. This phenomenon is due to higher rejection of excess traffic as the threshold is lowered. This reduces the total load on the high priority queue and thus lowers the average delay of high priority packets. On the other hand (Fig. 6), low priority traffic suffers a delay that roughly grows linearly with the threshold value. This is the case in the regime of interest: where the aggregate load is higher than unity. This is understood, since, in this regime, the total queue length is stabilized around the threshold by the occupancy dependent acceptance policy. Since low-priority traffic dominates the buffer space, it is directly affected.

Finally, looking at the excess traffic acceptance (Fig. 7), we see that for each load value there is a maximum acceptance ratio that can be reached by raising the threshold. This maximum ratio represents full utilization of service bandwidth left over after all committed traffic is served. It can be seen that for higher load values, low threshold values suffice to reach maximum utilization. At lower loads the server's idle probability (π_0) is significant (Fig. 2). Increasing the effective queue length for the excess traffic (raising the threshold), lowers this probability and **increases the utilization of the server**, resulting in more excess traffic throughput.

Putting our observations together, we suggest the following design procedure. First, set the committed traffic bandwidth share and loss probability targets (these would be in compliance with the various SLA commitments to the customers sharing this link, and monitored by a marking scheme). These performance targets for the committed traffic can be achieved by allocating sufficient headroom above the threshold so that even at high congestion periods loss probability remains low. Next the value of the threshold (and thus the total memory space allocated to the queue) can be set. The threshold selected is set to achieve the desired trade-off between excess traffic acceptance and low priority delay. In addition, the above analysis shows that for a given expected maximum aggregate link load, there is a threshold value region above which raising the threshold does not significantly improve acceptance ratio for excess traffic.

Concluding Remarks

This article is a first step in understanding the management of multipriority queues where traffic for each priority queue comprises two discard levels. Here we have used queuing theory to derive expressions for the expected delay and throughput of a simple threshold policy, and obtained very good approximation. One must acknowledge, though, the limitation of our approach that is based on the Poisson arrival process; indeed, we see deviation from the analysis when simulating more bursty traffic [2]. We believe that continuing in this direction, as well as using other tools such as competitive analysis, will help us better understand how to engineer and dimension such systems.

References

[1] S. Blake *et al.*, "An Architecture for Differentiated Services," IETF RFC 2475, Nov. 1998.
[2] D. Grossman, "New Terminology and Clarifications for DiffServ," IETF RFC 3260, Apr. 2002.
[3] F. L. Faucheur *et al.*, "Multi-Protocol Label Switching (MPLS) Support of Differentiated Services," IETF RFC 3270, May 2002.
[4] "Ethernet Services Attributes Phase 1," Metro Ethernet Tech. Specs., Nov. 2004.
[5] I. Stoica, S. Shenker, and H. Zhang, "Core-Stateless Fair Queuing," *IEEE/ACM Trans. Net.*, vol. 11, no. 1, Feb. 2003, pp. 33–46.
[6] A. K. Choudhury and E. L. Hahne, "Dynamic Queue Length Thresholds for Shared-Memory Packet Switches," *IEEE/ACM Trans. Net.*, vol. 6, no. 2, 1998, pp. 130–40.
[7] S. Iyer, R. R. Kompella, and N. McKeown, "Analysis of a Memory Architecture for Fast Packet Buffers," *IEEE Wksp. High Perf. Switching and Routing*, Dallas, TX, May 2001, pp. 368–73.
[8] I. Cidon, R. Guerin, and A. Khamisy, "On Protective Buffer Policies," *IEEE/ACM Trans. Net.*, vol. 2, no. 3, June 1994, pp. 240–46.
[9] Z. Lotker and B. Patt-Shamir, "Nearly Optimal FIFO Buffer Management for DiffServ," *PODC '02*, Monterey, CA, July 2002.
[10] S. Bergida and Y. Shavitt, "Analysis of Shared Memory Priority Queues with Two Discard Levels," Technical Report EE60, Tel Aviv Univ., School of Elec. Eng., Sept. 2006.
[11] I. Cidon, R. Rom, and Y. Shavitt, "Analysis of Multi-Path Routing," *IEEE/ACM Trans. Net.*, vol. 7, no. 6, Dec. 1999, pp. 885–96.
[12] M. A. Marsan *et al.*, *Modelling with Generalized Stochastic Petri Nets*, Wiley, 1995.
[13] L. Kleinrock, *Queuing Systems*, vol. 2: Computer Applications, Wiley, Inc., 1976.

Biographies

SHLOMI BERGIDA holds an M.Sc. (Cum Laude) in electrical engineering from Tel-Aviv University, Israel; and a B.Sc. (Summa Cum Laude) in electrical engineering from the Technion — Israel Institute of Technology, Haifal. He has 10 years of industry experience in high technology system design and leadership, specifically in the field of high-speed networking hardware and systems.

YUVAL SHAVITT (shavitt@eng.tau.ac.il) received a B.Sc. in computer engineering (cum laude), an M.Sc. in electrical engineering, and a D.Sc. from the Technion in 1986, 1992, and 1996, respectively. Between 1997 and 2001 he was an MTS at the Networking Research Laboratory at Bell Labs, Lucent Technologies, Holmdel, New Jersey. Since October 2000 he has been a faculty member in the School of Electrical Engineering at Tel Aviv University, Israel. He has served as a Technical Program Committee member for many networking conferences. He was an Editor of *Computer Networks* 2003–2004, and served as a Guest Editor for *IEEE JSAC* and *JWWW*. His recent research focuses on Internet measurement, mapping, and characterization, and QoS in networks.