An Enhanced Dropping Scheme for Proportional Differentiated Services

Jingdi Zeng and Nirwan Ansari
Center for Communications and Signal Processing Research
Electrical and Computer Engineering Department
New Jersey Institute of Technology
Newark, New Jersey 07102, USA

***Abstract*— As the DiffServ architecture is gaining ground, traffic engineering requires major adjustments. In addition, the measurement-based strategy has been widely adopted owing to its advantages of flexibility and easy adaptation. This article reviews measurement-based dropping schemes. Based on the definition and investigation of the "packet shortage" phenomenon, an enhanced dropping scheme for the proportional differentiated packet loss, referred to as "debt-aware," is proposed. Simulation results show that the scheme partially curbs negative effects of "packet shortage" and closely approximates loss differentiation parameters, as compared to a typical proportional dropping mechanism. More simulations have been applied to demonstrate the merits of this improved method.**

I. Introduction

As compared to Integrated Service (*IntServ*), the Differentiated Service (*DiffServ*) model defines an architecture for implementing the scalable service differentiation in the Internet. It achieves scalability by implementing the classification function only at network boundary nodes, and by applying Per-Hop Behaviors (*PHB*s) to traffic aggregates which have been marked using the Differentiated Services (*DS*) field in Internet Protocol (*IP*) headers [1]. The shift from the individual packet flow-oriented IntServ to traffic aggregate-oriented DiffServ model has had wide effects on traffic engineering, such as buffer management.

A generic buffer/queue management system handles delay and delay jitter with scheduling strategies, and guarantees packet/cell loss criteria with dropping mechanisms. Most dropping strategies require *a priori* knowledge about characteristics of all traffic streams, such as the sustained rate, peak rate, or burst length, and how they interact with each other. One type of dropping strategy, referred to as measurement-based, becomes rather attractive because of two major factors. First, with the booming of Internet services and changing of users' surfing patterns, a traffic model, which can be used to analyze preset parameters for traffic engineering, demands significant preciseness and may not be effective. Second, making dropping decisions by measured data, not by the statistical analysis and estimation, the measurement-based dropping strategy may be incorporated with any scheduling mechanism. Under certain circumstances, therefore, the measurement-based dropping is more flexible and robust as compared to others.

The intent of this article is to discuss the class-oriented and measurement-based dropping for proportional differentiated packet loss guarantees. In Section II, background related to the proportional dropping is presented. System and traffic models are given in Section III. Based on the "packet shortage" phenomenon defined and investigated in Section IV, an enhanced proportional dropping scheme, referred to as "debt-aware," is proposed. As compared to a typical proportional dropping scheme, its performance is then intensively simulated and evaluated in Section V. Concluding remarks are given in Section VI.

II. Measurement-based dropping mechanisms

Previous work [2][3][4][5][6] on the measurement-based dropping strategy has been proposed for IntServ, where Quality of Service (*QoS*) criteria are quantified by the explicit end-to-end delay and packet/cell loss. Corresponding to the absolute QoS provided by IntServ, a proportional differentiation service model was suggested [7] to provide the relative QoS. This model groups network traffic into M service classes which are ordered, such that class i is better or at least no worse than class $i-1$ for $1 < i \leq M$, in terms of per-hop metrics such as queuing delay and packet loss. Considering the loss aspect of the proportional differentiation model, a dropping strategy [8] was proposed to target for a controllable, predictable, and differentiated packet loss guarantee. It states that even in short time scales, per-hop packet drops should be proportional to the corresponding differentiation parameters chosen by network engineers, such that $\frac{l_i}{l_j} = \frac{\delta_i}{\delta_j}$, $1 \leq i, j \leq M$, where l_i is the average loss rate for class i, and $\delta_i, i = 1, 2, ..., M$, are differentiation parameters in terms of the loss rate, ordered as $\delta_1 > \delta_2 > \delta_3 > ... > \delta_M > 0$.

Two Proportional Loss Rate (*PLR*) droppers, namely $PLR(\infty)$ and $PLR(M)$, were introduced [8] to closely approximate the loss differentiation parameters δ_i. In $PLR(\infty)$, the loss rate estimation l_i is the long-term fraction of packets that have been dropped from class i, being measured by counters for arrivals and drops in each class. Denote A_i, D_i, and $B(t)$ as the counter of packet arrivals for class i, the counter of packet drops from class i, and the set of backlogged classes at time t, respectively. Whenever the buffer overflows, $PLR(\infty)$ drops a packet from the class whose index is determined by $min_{i \in B(t)}(\frac{D_i}{\delta_i A_i})$ (i.e., $min_{i \in B(t)}(\frac{l_i}{\delta_i})$). Instead, in $PLR(M)$, the loss rate of class i is estimated as the fraction of dropped packets from class i in the last M arrivals.

0-7803-7802-4/03/$17.00 © 2003 IEEE

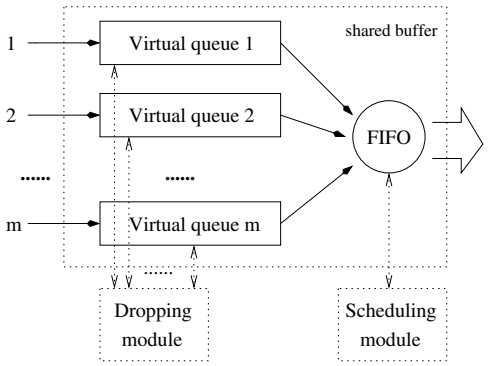


Fig. 1. The buffer/queue system model.

Its dropping mechanism is the same as $PLR(\infty)$ except with different parameter values.

III. SYSTEM AND TRAFFIC MODELS

It is assumed that a buffer/queue manager supports m ($m = 3$ in this article) service classes, one for each class selection PHB, as depicted in Fig.1. A dropping module decides when and which packets to be dropped. In addition, a First In First Out (*FIFO*) module determines which class shall be served next.

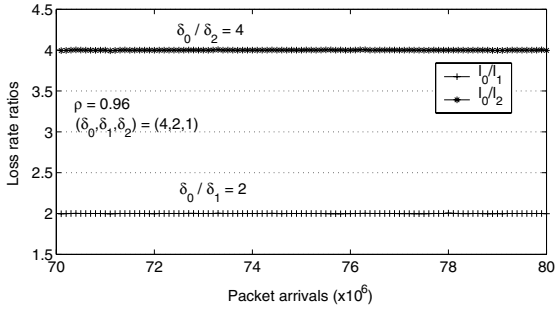
A typical deployment scenario is that an Internet Service Provider (*ISP*) classifies the traffic at the core edge by application types, customer service requirements, etc. When the load of aggregates is low, all classes experience the same level of QoS; otherwise, the differentiated QoS (e.g., packet loss in this article) have to be maintained by ISP. Provided with a means of monitoring the service performance, users can either adjust their service criteria or switch to another class.

To model popular self-similar traffic aggregates (e.g., Ethernet traffic), it is assumed that each traffic class is aggregated by Pareto-distributed ON-OFF sources, respectively. A scale parameter of 1.2, which was suggested [9][10] by early empirical studies, is applied to all classes. Moreover, the minimum values of ON and OFF time periods for the three respective classes of traffic, are defined as $0.5ms, 1ms, 1.5ms$ and $1.61ms, 2.9ms, 4.85ms$, respectively. Traffic aggregate traces generated under these assumptions, which are not shown here due to the limited space, exhibit the self-similar characteristic at different time scales and easily pass the "visual" test.

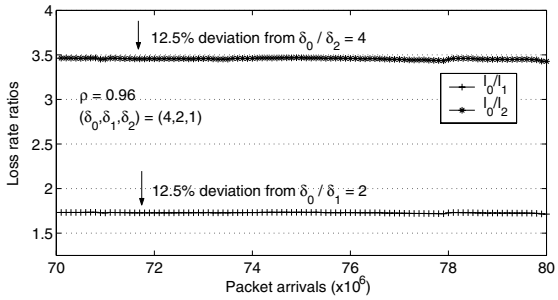
IV. "PACKET SHORTAGE" AND AN ENHANCED DROPPING METHOD

A. The "packet shortage" phenomenon

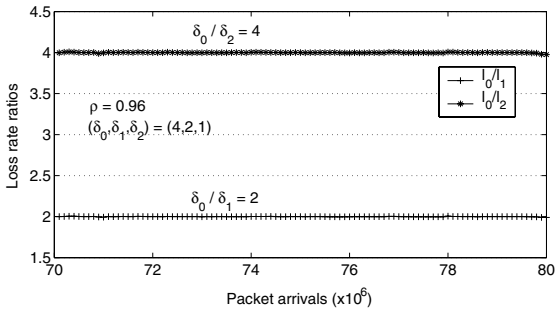
With the normalized traffic load distribution of three classes $(L_0, L_1, L_2) = (56\%, 30\%, 14\%)$, $PLR(\infty)$ closely approximates differentiation parameters $\delta_0 : \delta_1 : \delta_2 = 4 : 2 : 1$, as shown in Fig. 2(a). Assuming a 10% QoS deviation defined in Service Level Agreement (*SLA*), we show that the agreement will be violated under certain circumstances. For instance, when picking up another load distribution $(L_0, L_1, L_2) = (14\%, 30\%, 56\%)$, the ratios $\frac{l_0}{l_1}$ and $\frac{l_0}{l_2}$ of $PLR(\infty)$ exhibit a



(a) $PLR(\infty)$ with a normalized load $(L_0, L_1, L_2) = (56\%, 30\%, 14\%)$.



(b) $PLR(\infty)$ with a normalized load $(L_0, L_1, L_2) = (14\%, 30\%, 56\%)$.



(c) "Debt-aware" with a normalized load $(L_0, L_1, L_2) = (14\%, 30\%, 56\%)$.

Fig. 2. "Packet shortage" phenomenon: (a) with an appropriate traffic load distribution, $PLR(\infty)$ approximates its differentiated ratios well; (b) "packet shortage" caused by another traffic load distribution, however, induces an about 12.5% deviation to both rate ratios of $PLR(\infty)$; (c) alleviating the "packet shortage" problem, "debt-aware" closely approximates the required rate ratios.

12.5% deviation from $\frac{\delta_0}{\delta_1}$ and $\frac{\delta_0}{\delta_2}$, respectively, as illustrated in Fig. 2(b). The obvious cause is that in the latter case, the load distribution does not synchronize with differentiation parameters, that is, classes with bigger differentiation parameters (i.e., higher tendency to have smaller values of $\frac{l_i}{\delta_i}$ and therefore heavier dropping demands) have lighter traffic loads as compared to others. Referred to as "package shortage," this problem has two possible consequences: the dropping module may not be able to drop a packet from a designated class if this class happens to be lightly loaded; it in turn induces unnecessary losses of other backlogged classes, although rate

ratios among classes may be regained after several rounds of overflows. All these are against the original goal of the differentiated dropping: closely approximate differentiation parameters even at short time scales.

Seeking to alleviate the negative effects of "packet shortage," enlarging the buffer size is considered as a potential solution. Being long enough, a buffer shall more likely be able to accommodate packets from every class, and therefore help the dropping module identify packets to drop at the moment of overflow.

Assume there are m traffic classes, each of which is superpositioned by n ON-OFF sources. The ON and OFF periods of source j in class i are Pareto distributed with the scale parameter $\alpha_{i,j}$, and lower cut-offs of $b_{on_{i,j}}$ and $b_{off_{i,j}}$, respectively. Then Probability Density Functions (*PDF*s) of an ON period $X_{on_{i,j}}$ and an OFF period $X_{off_{i,j}}$ follow

$$f_{X_{on}}(x_{on_{i,j}}) = \frac{\alpha_{i,j}(b_{on_{i,j}})^{\alpha_{i,j}}}{(x_{on_{i,j}})^{\alpha_{i,j}+1}}, \quad x_{on_{i,j}} \geq b_{on_{i,j}} \qquad (1)$$

and

$$f_{X_{off}}(x_{off_{i,j}}) = \frac{\alpha_{i,j}(b_{off_{i,j}})^{\alpha_{i,j}}}{(x_{off_{i,j}})^{\alpha_{i,j}+1}}, \quad x_{off_{i,j}} \geq b_{off_{i,j}}, \qquad (2)$$

respectively, where $i = 1, 2, ..., m, j = 1, 2, ..., n$. Denote $L_{i,j}$ and $R_{i,j}$ as the average load and peak rate of the j_{th} ON-OFF source in class i, respectively. It can be shown that the buffer length B, which is sufficient to hold at least one ON period (i.e., one burst) from *every* class, is

$$B \geq \max_i \min_j (b_{on_{i,j}} + b_{off_{i,j}}) \quad \times \sum_{j=\arg\min_j (b_{on_{i,j}} + b_{off_{i,j}})} R_{i,j} L_{i,j}, \qquad (3)$$

where $i = 1, 2, ..., m, \quad j = 1, 2, ..., n$.

For the j_{th} source in class i, moreover, the traffic load and the length of a pair of ON and OFF periods are $L_{i,j} = \frac{X_{on_{i,j}}}{X_{on_{i,j}} + X_{off_{i,j}}}$ and $Z_{i,j} = X_{on_{i,j}} + X_{off_{i,j}}$, respectively. Since both the ON and OFF periods have the same scale parameter, the function of the buffer size B, which has at least one ON period from *every* class, can be accordingly expressed as:

$$F_B(Z_{i,j}, L_{i,j}) = \max_i \min_j Z_{i,j} \quad \times \sum_{j=\arg\min_j z_{i,j}} R_{i,j} L_{i,j}. \qquad (4)$$

Given an upper bound of the buffer size x, the probability of at least one ON period from *every* class accommodated in the buffer is derived as the following:

$$P(B \leq x) = \int_0^x f_B(x) dx. \qquad (5)$$

This expression is based on the availability of a closed form of $f_B(x)$, the PDF of $F_B(Z_{i,j}, L_{i,j})$. The characteristic function of the Pareto distribution, however, is not integrable in a closed algebraic form [11][12]; thus, inversion methods of obtaining $f_Z(x)$ and $f_L(x)$, that is, PDFs of $Z_{i,j}$ and $L_{i,j}$, are not immediately applicable. This rules out an explicit expression of $f_B(x)$. In other words, "packet shortage" cannot be eliminated by simply enlarging the buffer.

D_i: the number of packets dropped from class i.
A_i: the number of packet arrivals of class i.
δ_i: the loss differentiation parameter of class i.
d_i: the "drop debt" carried by class i.

A class i packet arrives, A_i++;
if (the buffer overflows)
{ sort $\frac{D_i}{\delta_i A_i}$, $i = 1, 2, ..., m$, in an ascending order;
find an eligible class j, $j = \arg min_{i \in B(t)}(\frac{D_i}{\delta_i A_i})$, where $i = 1, 2, ..., m$, and $B(t)$ is the set of backlogged classes;
update the "drop debt" counters, that is, d_k++, where $k = 1, 2, ..., j-1$;
drop the packet at the tail of class j, D_j++;
}
else
{ looping through "drop debt" counters, pick up class k which is backlogged;
d_k--;
drop the packet at the tail of class k, D_k++;
}
Accept the incoming packet;

Fig. 3. The pseudo code of "debt-aware."

B. An enhanced "debt-aware" dropping scheme

The previous discussion and analysis highlight three features an enhanced dropping method shall have: closely approximating loss differentiation parameters by relieving the "packet shortage" phenomenon, dropping packets whenever it is necessary, not just at overflow moments, and still being based on simple on-line measurements.

The enhanced proportional dropping method with a "drop debt" memory, referred to as "debt-aware," is therefore suggested. Instead of only considering backlogged classes, this method monitors all classes and adopts corresponding actions. Its pseudo code is listed in Fig. 3. Taking the cheap memory and the fast access speed of digital circuits nowadays into consideration, the complexity of the system does not significantly increase, with an extra register for each of the limited number of service classes.

Before looking into simulation results, essential characteristics and advantages of "debt-aware" are summarized as follows: first, it expands the reach of the dropping module to incoming packets as well as backlogged ones, and thus partially curbs the adverse effect of the traffic load on the system performance. Second, in addition to overflow moments, a necessary dropping takes place when there is a "debt." This "debt" memory is exactly the effort to immediately identify packets which stay in the buffer but are eventually pushed out. Dropping these packets at an earlier stage can not only avoid causing the loss of other packets, but can also improve the queuing delay performance, as will be demonstrated in the following section.

V. SIMULATIONS AND DISCUSSION

Unless otherwise noted, all simulations utilize a buffer length $B = 2K\ packets$. All packets have a constant length of $1K\ bits$. The link utilization, which is defined as the ratio of the aggregated traffic arrival to the service rate, is denoted

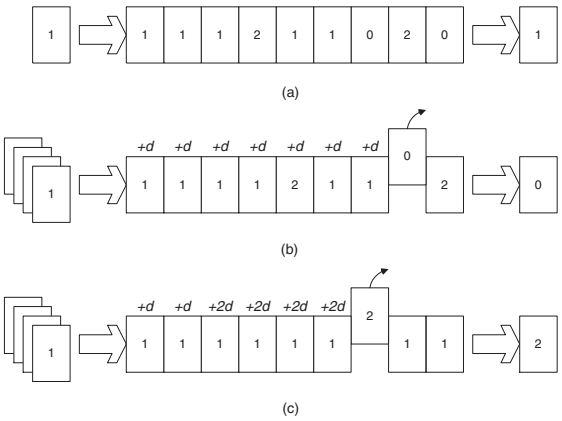


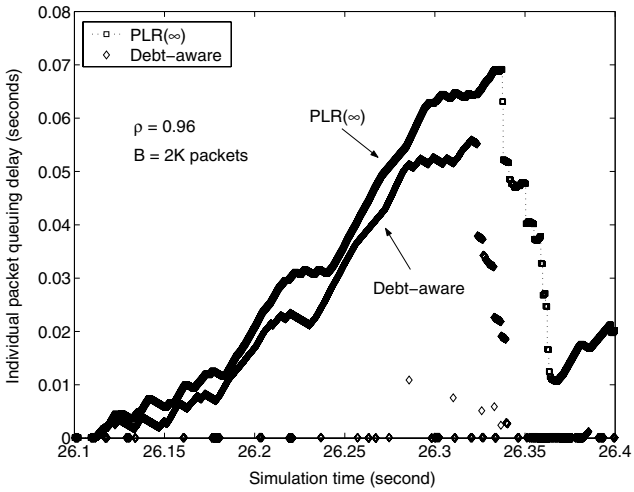
Fig. 4. Snapshots demonstrating the excess queuing delay which can be regained by "debt-aware."

as ρ and given in each figure. All samples are measured based on a window of $100K$ $packets$.

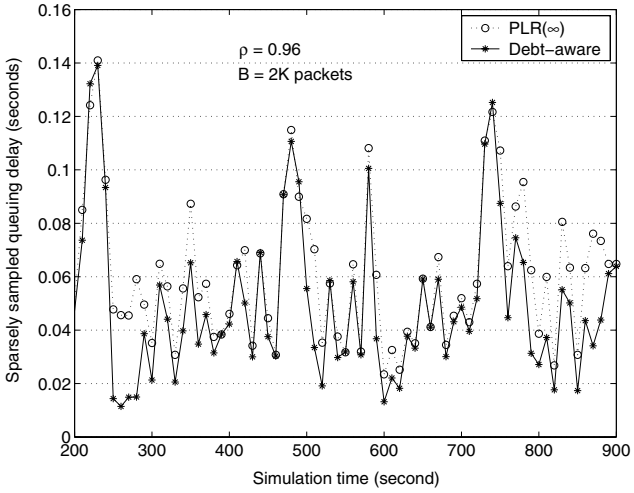
As compared to $PLR(\infty)$ that only drops backlogged packets when buffer overflows, "debt-aware" provides more "operation space." It remembers packets which are supposed to be dropped, and blocks out the eligible one when it comes in. With the same load distribution that induces a 12.5% performance deviation of $PLR(\infty)$ in Fig. 2(b), "debt-aware" curbs the rate ratios back to their criteria, as shown in Fig. 2(c). Furthermore, "debt-aware" is able to achieve the equivalent performance of $PLR(\infty)$ under normal load distribution. The corresponding simulation result is not shown due to the limited space.

One may argue that "debt-aware" drops packet too aggressively; this is not completely true. Since both $PLR(\infty)$ and "debt-aware" aim to curb loss rate ratios even at a short time scale, $PLR(\infty)$ will eventually push out whatever is supposed to be dropped. Therefore, we argue that "debt-aware" does not over-drop, but just do so at an earlier stage. For the policing purpose, $PLR(\infty)$ may not be strict enough because a biased ratio can go on for an unknown period of time until it is regulated in one of the succeeding overflow moments or a subscriber changes its service class, whichever comes first.

Another enhancement of "debt-aware" is that it improves the performance of packet queuing delay, by distinguishing certain packets at an earlier stage. Since $PLR(\infty)$ only drops at the moments of overflow, certain packets may stay in the buffer and delay other packets during the interval of two successive overflows, until they either are finally dropped or leave the queue. Fig. 4 demonstrates three snapshots of a buffer, where every packet is marked with its class index. In the snapshot shown in Fig. 4(a), the arriving and serving processes are keeping a dynamic balance, whereby the buffer is full but does not overflow. Then a burst comes and an overflow is about to happen, as shown in Fig. 4(b). Denote the service time of one packet as d. Assume that the serving process of the last packet has finished and class 0 has the minimum value of $\frac{D_i}{\delta_i A_i}, i = 1, 2, ..., m$. Under this circumstance, the dropping



(a)



(b)

Fig. 5. Packet queuing delay with different sample density for $PLR(\infty)$ and "debt-aware": (a) demonstrates individual packet queuing delay in a very short but typical time period, where the trace of "debt-aware" is below that of $PLR(\infty)$, and shows smaller queuing delay; (b) sparsely plots 70 samples in a 700-second simulation period, where "debt-aware" frequently exhibits smaller queuing delay than that of $PLR(\infty)$.

module of $PLR(\infty)$ will drop the tail packet of class 0; "debt-aware" case, however, could have blocked out this very packet upon its arrival, if class 0 carries a "debt." The consequence of the $PLR(\infty)$ scenario, as demonstrated in Fig. 4(b), is that all packets behind the discarded one are penalized with an extra delay of d. Before the overflow is over, the same situation could happen again. As illustrated in Fig. 4(c), the tail packet of class 2 happens to be the next one eligible to be dropped, and therefore packets lining up behind this one suffer from another queuing delay of d in the $PLR(\infty)$ case.

To further illustrate the regained delay, Fig. 5(a) plots the

individual packet queuing delay in a very short but typical period. Both traces are quite thick due to the very high sample density. With its sample trace completely under that of $PLR(\infty)$, "debt-aware" exhibits much less queuing delay than $PLR(\infty)$ does, confirming to the previous explanation. Another two traces with sparse samples are presented in Fig. 5(b), both of which contain 70 samples in a 700-second simulation period. Frequently, "debt-aware" exhibits smaller queuing delay as compared to $PLR(\infty)$, except at very few sample points.

One may argue that the decreased queuing delay is just resulted by a more aggressive dropping, instead of the early stage dropping feature of "debt-aware." However, the existence of exceptional sample points in Fig. 5(b), although very few, is exactly a good counterevidence: if the performance was simply gained by an aggressive dropping, all queuing delay values of "debt-aware" must have been lower than or at least equal to those of $PLR(\infty)$. The possible reason of these few exceptional values, moreover, can be explained as follows: since "debt-aware" does the early dropping in a round robin manner which treats all classes equally, it may not always follow the dynamic dropping order among classes as $PLR(\infty)$ does. Consequently, both schemes may see different buffer contents at the same moment of overflow. Assume that "debt-aware" picks up packet A and $PLR(\infty)$ chooses packet B when the buffer overflows. If packet A is behind packet B in the queue, all packets between packet A and B will experience one more measure of queuing delay (i.e., d) in "debt-aware;" this contributes to a longer delay experienced by certain sample points in "debt-aware."

It is worth mentioning that neither "debt-aware" nor $PLR(\infty)$ can improve the situation when the overall (not only backlogged) traffic load of a class is not sufficient to approximate its loss rate ratios to other classes, as regulated by differentiation parameters. This is the infeasible region investigated in [8]. It implies that it is critical for ISPs and users to be able to adjust differentiation parameters and service criteria, respectively.

VI. Conclusions

This article discusses the dropping strategy for proportional differentiated packet loss guarantees. The "packet shortage" phenomenon, frequently revealed in classical proportional dropping schemes, has been defined and investigated. Referred to as "debt-aware," an enhanced measurement-based dropping method is then suggested and evaluated. By simply adding one register/counter to each service class, "debt-aware" improves the queuing delay performance, and partially curbs the "packet shortage" phenomenon to closely approximate loss differentiation parameters.

Future work could be threefold: besides different traffic load distributions, the effects of the traffic load fluctuation on the dropping performance need further investigation; when coupling with other scheduling mechanisms rather than FIFO, the overall performance could be different; the possible effects of the restrict dropping in "debt-aware" on specific loss-sensitive applications require additional attention.

Acknowledgment

This work has been supported in part by the New Jersey Commission on Higher Education via the NJI-TOWER project and the New Jersey Commission on Science and Technology via the NJ Center for Wireless Telecommunications.

References

[1] S. Blake, S. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for Differentiated Services," *IETF RFC 2457*, December 1998.

[2] D. P. Heyman, A. Tabatabai, and T. V. Lakshman, "Statistical analysis and simulation study of video teleconference traffic in ATM networks,"*IEEE Trans. on Circuits and System for Video Technology*, Vol. 2, No. 1, pp. 49-59, March 1992.

[3] T. Yang, D. Tsang, and S.Li, "Cell scheduling and bandwidth allocation for a class of VBR video connections," *IEEE Workshop on Visual Signal Processing and Commun.*, New Brunswick, NJ, pp. 95-101, Sep. 1994.

[4] T. Yang, D. H. K. Tsang, and P. McCabe, "Cell scheduling and bandwidth allocation for heterogeneous VBR video conferencing traffic," *Proc. IEEE GLOBECOM'95*, Vol.1, Singapore, pp. 371-377, Nov. 1995.

[5] T. Yang and J. Pan, "A measurement-based loss scheduling scheme," *Proc. IEEE INFOCOM'96*, Vol.3, San Francisco, CA, pp. 1062-1071, Mar. 1996.

[6] H. J. Chao and H. Cheng, "A new QoS-guaranteed cell discarding strategy:self-calibrating pushout," *Proc. IEEE GLOBECOM'94*, Vol.2, San Francisco, CA, pp. 929-934, Nov. 1994.

[7] C. Dovrolis and P. Ramanathan, "A case for relative differentiated services and the proportional differentiation model," in *IEEE Network*, Vol. 13, No. 5, pp. 26-34, September-October 1999.

[8] C. Dovrolis and P. Ramanathann, "Proportional differentiated services, part II: loss rate differentiation and packet dropping," in *Proc. 8th Int. Workshop on Quality of Service (IWQoS'00)*, Pittsburgh, PA, pp. 53-61, Jun. 2000.

[9] W. Willinger, M. S. Tappu, R. Sherman, and D. V. Wilson, "Self-similarity through high-variability: statistical analysis of Ethernet LAN traffic at the source level," *IEEE/ACM Trans. on Networking*, Vol.5, No.1, pp. 71-86, February 1997.

[10] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of Ethernet traffic," *Proc. ACM SIGCOMM'93*, pp. 183-193, San Francisco, CA, Sep. 1993.

[11] H. L. Seal, *Survival probabilities: the goal of risk theory*, Wiley, Chichester, 1978.

[12] H. Buhlmann, *Mathematical methods in risky theory*, Spring-Verlag Berlin, Heidelberg, 1970.