On the Effectiveness of Push-out Mechanisms for the Discard of TCP Packets

Kleber V. Cardoso
GTA/COPPE
Federal University of Rio de Janeiro
Rio de Janeiro, Brazil

José F. de Rezende
GTA/COPPE
Federal University of Rio de Janeiro
Rio de Janeiro, Brazil

Nelson L. S. Fonseca
State University of Campinas
Institute of Computing
Campinas, Brazil

***Abstract*— This paper investigates the joint use of push-out mechanisms with RED-like discarding policies to support service differentiation in the Internet. The efficiency and the degree of differentiation of a Complete Sharing with push-out, a RIO and a RIO with push-out queues are assessed. Results indicate that push-out used jointly with RED-like policies does not improve the performance.**

***Index Terms*—Selective Discard, Quality of Service (QoS), DiffServ.**

I. INTRODUCTION

The Internet Differentiated Services (DiffServ) model [1] has received a great deal of attention. Selective discard is one of the traffic control mechanisms in this model. Selective discard drops packets in congestion situations according to their priority level. It is only related to the management of the buffer space, i.e. selective discard discriminates packets without disordering them. Thus, it can be used to handle packets for the Assured Forwarding (AF) PHB [2], defined in the DiffServ model. Moreover, keeping the packet order in a traffic stream is useful for differentiating packets belonging to the same flow.

Selective packet discard policies may have a significant influence on the performance achieved by different priority classes, specially for TCP flows, which are congestion sensitive flows. The loss distributions (loss rate and loss gap length) imposed by selective discard policies determine the bandwidth allocation in network bottlenecks. In line with that, RED-n (also named as Multiclass RED [3]) buffer management mechanism was defined. RED-n signals when a discard must be performed. However, it does not determine which packet has to be dropped. Due to implementation issues, the arriving packet is the best candidate to be dropped. Another approach is to allow enqueued packets to be discarded, instead of dropping arriving packets, i.e., to use *push-out* policies. Several push-out policies have been studied, however, none of these studies has taken into account the most common Internet traffic characteristics.

This paper investigates the impact of jointly using push-out and RED-n policies on the per class performance. Different discard policies are compared under short-term and long-term TCP traffic.

This paper is organized as follows. Section 2 review some selective discard concepts. Section 3 introduces the comparison of discard policies carried out in this paper. Section 4 describes the simulation experiment, and Section 5 discusses the numerical results. Finally, in section 6, conclusions are drawn.

This work was supported by FUJB, CNPq, CAPES, COFECUB and FAPERJ.

II. SELECTIVE PACKET DISCARD

Selective packet discard includes two types of policies: buffer management policies, which define when a packet must be dropped; and push-out policies, which determine which packet should be dropped.

In the Assured Forwarding PHB, buffer management is generally performed by RED algorithms: one for each level of priority. Each RED algorithm is set less aggressively than the RED algorithm associated to immediately lower priority class, so that packets with low priority are discarded first.

RIO (RED with In/Out bit) is a discarding policy which manages two RED queues: one for packets whose traffic stream is in accordance to a profile agreed between user and service provider (In packets), and a RED queue for packets whose traffic stream does not follow the pre-established profile (Out packets). Out packets are discarded before the In packets. To accomplish that, the threshold for Out packets has to be lower than the threshold for In packets, and Out packets should have a higher maximum probability (the probability of discarding a packet in RED) than In packets.

Complete Sharing is a buffer management policy which has been widely implemented in commercial multiplexers. Under Complete Sharing any buffer slot can be occupied by any packet irrespective of the class the packet belongs to. Packets are dropped if and only if the buffer is full. In networks where packets are fixed size, such as ATM networks, no work is lost unnecessarily.

Dropping packets at different buffer positions defines different push-out policies. The most common are: First-In-First-Drop (FIFD), Last-In-First-Drop (LIFD), Modified-FIFD (M-FIFD) and Random (RAND). Under FIFD, the low priority packet nearest to the head of the queue is dropped, whereas under LIFD, the low priority packet nearest to the tail of the queue is dropped. Under M-FIFD, an arriving low priority packet removes an enqueued low priority packet closest to the head of the queue. Under RAND, a randomly selected packet is dropped.

Dropping low-priority packets at different queue positions defines different queue distributions over a certain period of time, and consequently may differently impact the performance. A drop next to the head of the queue signals faster a packet loss to the receiver. Besides, it magnifies the probability that a low priority packet is found in the buffer when a high priority packet arrives. In this way, FIFD provides a lower loss rate to high priority packets than LIFD and RAND.

Until recently, push-out policies were considered hard to implement because they implied queue reordering. However, the new ASICs-based technologies, which are applied in the design of high speed switches, allow the implementation of push-out policies without a high overhead [4].

III. Comparing different discard policies

In the present investigation, three different selective discard mechanisms are compared: PO (complete sharing push-out), RIO and PRIO (push-out RIO). PRIO uses RIO to signal the need to drop a packet and a push-out policy (FIFD) to choose the one to be dropped. Note that PO drops packets only when the buffer is full, whereas PRIO may drop packets even if there are available buffer slots.

To compare PO, PRIO and RIO, different performance metrics, which capture either the point of view of the service provider or that of the customers are used. Bandwidth utilization is used as a service provider's metric. If users are charged based on the consumed bandwidth, instead of being charged a flat rate, it is desirable to optimize the bandwidth utilization. Another important metric is packet loss rate, which implies wasted bandwidth and buffers. From the customers' perspective, the most important metric is average goodput, i.e. the actual obtained bandwidth. The second metric of customers' interest is related to the degree of differentiation. This metric is defined as the ratio between the goodput obtained by a service class and by the bandwidth obtained by the adjacent lower priority class. This metric thus describes the separation between classes. Customers who subscribe to high priority services, and pay more for that, expect to get a better quality than the ones who pay less.

IV. The simulation experiment

The simulation experiments were performed using ns-2 and were carried out in two steps in order to pinpoint the effects of the traffic generator used. In the first step, parameters related to the TCP traffic model are studied, using the Drop Tail (DT) as the buffer management mechanism. In the second step, different buffer management policies are compared.

For validating the simulation results, confidence intervals were derived by the independent replication method. A 95% confidence interval with the maximum width of 5% of the obtained values was adopted. The simulation experiments were run until an established number of connections per class have finished their data transfers. To eliminate transient results, only the connections whose transfers started 50 seconds after the beginning of the simulation were considered. Confidence intervals are not displayed in the graphs for the sake of visual interpretation.

The topology shown in Figure 1 was used. The bottleneck link is the one between $r1$ and $r2$, and its capacity is C. It is shared between the two classes: the high priority class, class 1, and the low priority class, class 2. The mean arrival rate of connections per class is given by λ_i ($i \in 1, 2$), i.e., at each $1/\lambda_i$ a new connection arrives to a link with capacity M connected to the node s_i. The load generated by several class i connections is ρ_i, given by $\rho_i = \lambda_i * L_i/C$, where L_i is class i mean transfer size, i.e., the amount of data to be transferred. This traffic crosses the bottleneck link and goes to node d_i. The system stability condition is given by $\rho = \rho_1 + \rho_2 < 1$.

The mean arrival rate and the transfer size are specified in order to meet the desired traffic characteristics. For example, in web traffic, the mean arrival rate is related to the aggregate dynamics of users browsing web pages, whereas the transfer size determines the TCP connection "duration". In this case, most of the connections are short-term, transferring one or more web page objects[1]. The transfer size parameter distribution is described by a log-normal with mean of 4827 bytes [5][6]. The tail (extreme values) of the transfer sizes is not considered, since it represents a small portion of the total traffic volume [6] [7] [8]. Long-term TCP connections, which appear in FTP applications, are also present in web traffic and their size can be over 5 MB. In the simulation experiments, the impact of different transfer sizes on the per class obtained bandwidth is assessed.

The access link capacity (M) determines the peak rate at which bursts generated by the sources arrive at node s_i. The aggregation nodes s_i are connected to the bottleneck $r1$ by a link with a capacity enough to avoid losses. PO, PRIO, RIO and DT are implemented at node $r1$ and at node $r2$. The buffer size in these nodes is equal to the bandwidth-latency product [9]. The TCP implementation used here is Reno since it is the most common version in the Internet.

Initially, ρ was calibrated taking into account the measured network load. When using a protocol that performs congestion control, such as TCP, the service rate is not given by the ratio $1/L/C$, since the transmission rate is regulated by the losses. In line with that, simulations were performed to identify the actual relation between average network load and ρ. This approach follows the one introduced in [10]. The transfer sizes for long-term connections was fixed at 256 KBytes. For short-term connections, the transfer size follows a log-normal distribution with mean 4827 bytes.

Figure 2 illustrates the mentioned relationship. The functions that best fit the results were obtained by using the method of non-linear minimum square. In the case of fixed transfer size, $load = 1,00001 * \rho - 4,50\text{E-}6$. In the case of HTTP traffic, $load = 1,06581 * \rho - 2,07\text{E-}3$. Figure 2 also depicts

[1]HTTP 1.0 uses a TCP connection per object. HTTP v1.1 reuses TCP connections.

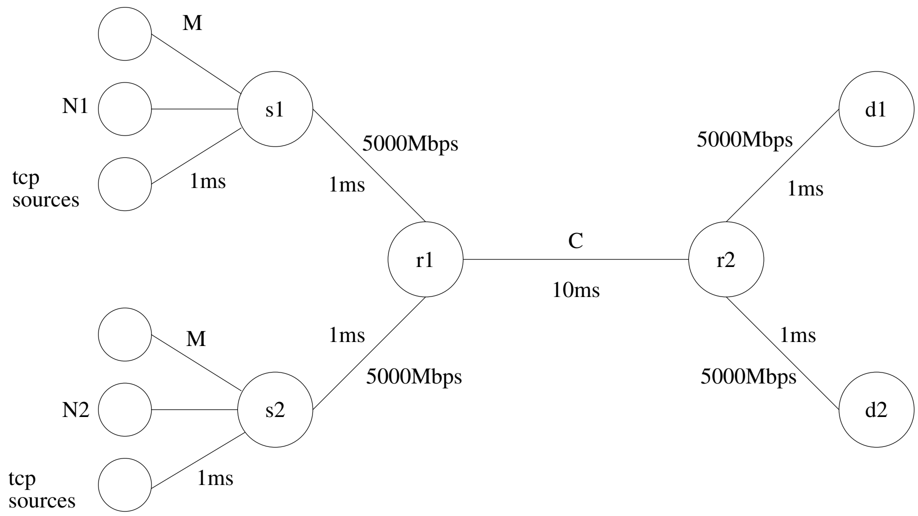

Fig. 1. Topology.

the straight line where $load = \rho$.

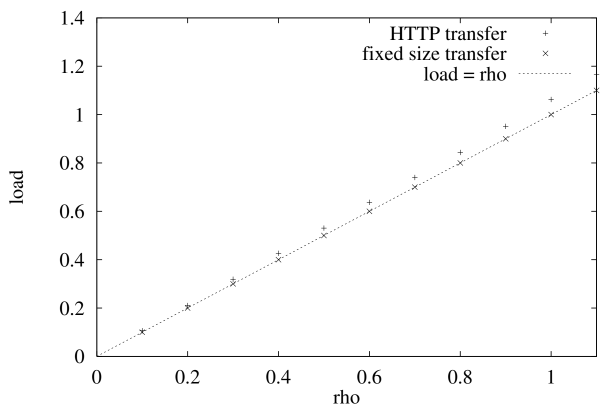

Fig. 2. Load and ρ relationship.

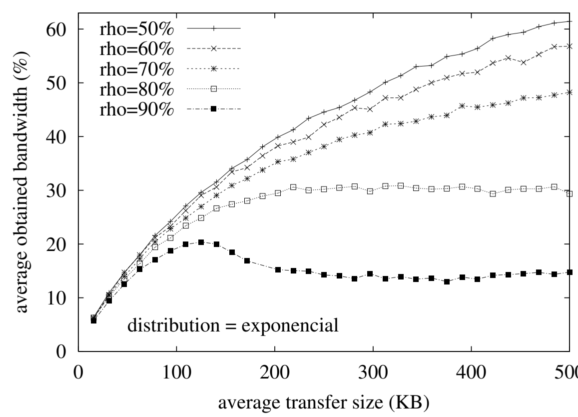

Fig. 3. Bandwidth obtained as a function of average transfer size.

V. Numerical results

A. *The Impact of the Transfer Size on the Obtained Bandwidth*

The relationship between the transfer size and the obtained bandwidth was investigated under different load conditions. Figure 3 shows the aggregate obtained bandwidth as a function of the average transfer size (in bytes) for different loads (ρ), for $\rho_1 = \rho_2 = \rho/2$. The bandwidth is normalized by the bottleneck link bandwidth. Drop Tail (DT) was used, so that there is no class differentiation. The transfer size follows an exponential distribution with mean L.

The obtained bandwidth decreases as the average transfer size decreases. This happens because the majority of the short-term TCP connections operates only in the slow-start phase (transient period), and thus do not reach the congestion avoidance phase (steady state). Since slow-start performs a slow ramping up of the congestion window, there is an upper limit on the bandwidth which can be obtained by short-term connections.

B. *Short-term connections*

In this section, the effectiveness of selective discard mechanisms for short-term transfers is evaluated. In order to assess the sensitivity of the discarding mechanisms, class 1 load was kept constant and the total load was increased up to 0.9. A desirable result would be the remaining of class 1 to keep performance constant with the increase of the total load. Figure 4 shows the average bandwidth per class when ρ_1 is fixed at 0.2. Simulation experiments with different values of ρ, ρ_1 and ρ_2 were run, and similar results were observed.

As can be seen in Figure 4, class 1 obtained bandwidth does not significantly change under PRIO and RIO policies. It is worth noting that the joint use of push-out and RIO does not

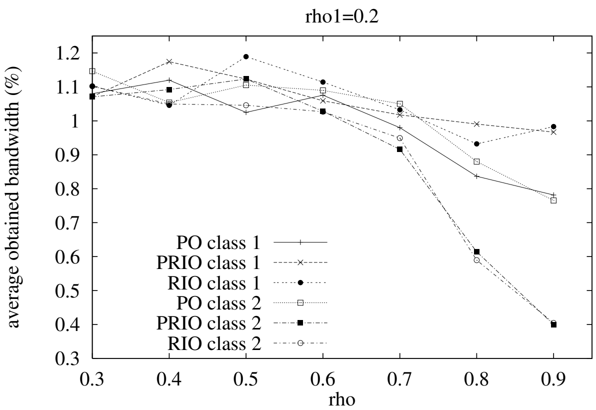

Fig. 4. Bandwidth of classes 1 and 2 as a function of network load.

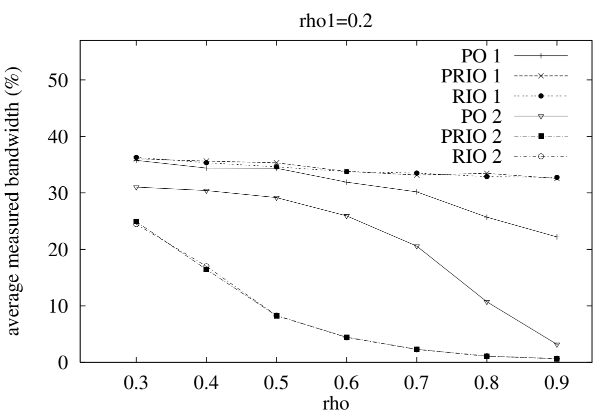

Fig. 5. Bandwidth of classes 1 and 2 as a function of network load.

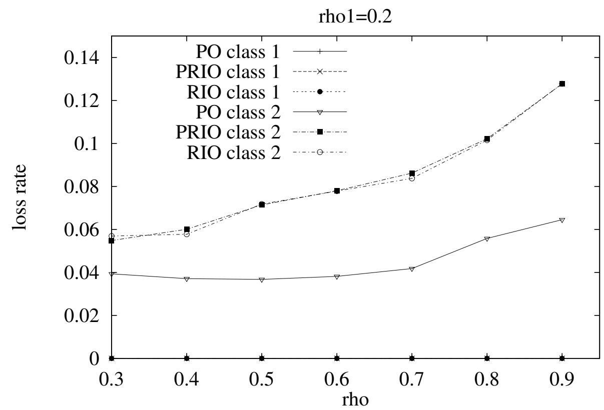

Fig. 6. Classes 1 and 2 loss rate as a function of network load.

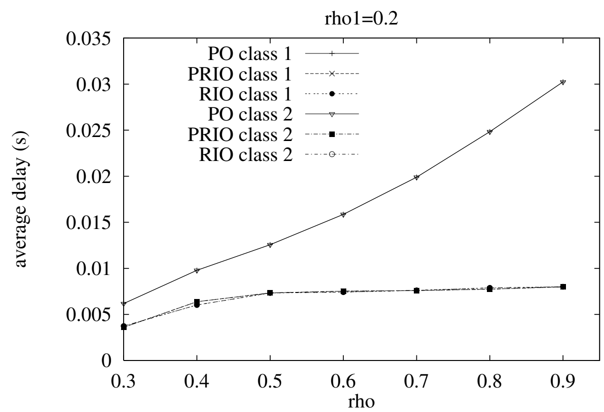

Fig. 7. Queue delay as a function of network load.

offer any improvement to both classes. PO is the most sensitive mechanism to the load increase, and it does not significantly differ the priority classes. Under PRIO and RIO, a significant decrease of class 2 bandwidth is observed.

C. Long-term connections

To produce long-term connections, the transfer size was drawn from an exponential distribution with mean of 512 KBytes. The load variation follows the same pattern of the one used for short-term connections. Figure 5 shows the average bandwidth keeping ρ_1 (load of class 1) fixed at 0.2. In despite of the difference in the bandwidth obtained by class 2, which is even lower than the one for short-term connections, RIO and PRIO showed the same trend observed for short connections.

To better understand results shown in Figure 5, the loss rate, the delay and the average number of timeouts are displayed in Figures 6, 7, and 8, respectively, since these are the major factors impacting the bandwidth obtained by a priority class.

As can be seen, RED-based discard policies (RIO and to PRIO) provides more bandwidth to the high priority class, which is mainly due to the significant increase in the low priority loss rate (Figure 6). However, class 1 bandwidth slightly decreases due to a small delay increase (Figure 7), since neither losses nor timeouts occur (Figure 8). From Figure 5 it is also possible to observe that under PO the loss rates per class do not significantly differ, and they are smaller than those obtained under RIO and PRIO, as it can be observed in Figure 6. The reason for that is that PO discards packets only when the buffer of the bottleneck link overflows. The differentiation under PO is not depicted in Figure 4 because short-term connections do not give the TCP opportunity to reach its steady state. The bandwidth decrease under PO is due to the significant increase in queue delay, according to Figure 7. On the other hand, PRIO and RIO policies keep a low queuing delay, which is a property of early discard policies.

Results given by both RIO and PRIO do not significantly differ. A possible reason for that is that both policies are based on an early discard mechanism which overlaps the push-out behavior of making room to a recently arrived packet by removing an enqueued one. Consequently, differentiated loss patterns per

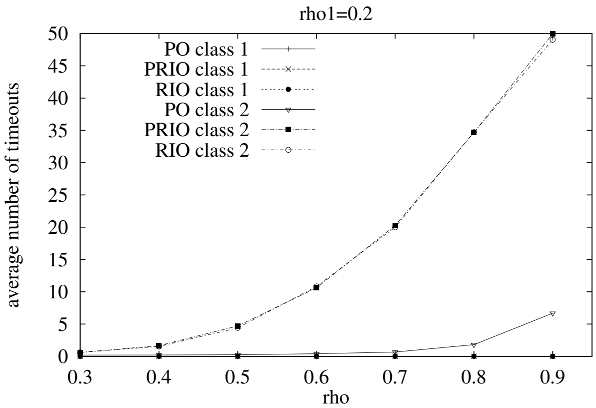

Fig. 8. Average number of timeouts for classes 1 and 2 as a function of network load.

class are not observed.

VI. Conclusion

In this paper, the joint effect of push out and early discard policies was investigated, under short-term and long-term TCP traffic. The analysis took into consideration the differentiation degree and the bandwidth utilization under different network load conditions and transfer sizes. The joint use of push-out and early packet discard policies is not attractive, since early packet discard mechanisms overshadows push-out main effect.

References

[1] S. Blake, D. L. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services," *RFC 2475*, Dec. 1998.

[2] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB Group," *RFC 2597*, June 1999.

[3] M. Goyal, A. Durresi, P. Misra, C. Liu, and R. Jain, "Effect of Number of Drop Precedences in Assured Forwarding," in *IEEE/Globecomm'99*, (Rio de Janeiro), Dec. 1999.

[4] R. Cohen, N. Fonseca, and M. Zukerman, "Traffic management and control," in *Multimedia Communications Networks* (B. Khasnabish and M. Tatipamula, eds.), Artech House, 1998.

[5] M. Molina, P. Castelli, and G. Foddis, "Web Traffic Modeling Exploiting TCP Connections' Temporal Clustering through HTML-REDUCE," *IEEE Network Magazine*, vol. 14, no. 3, pp. 46–55, 2000.

[6] P. Barford and M. Crovella, "Generating representative web workloads for network and server performance evaluation," in *Proc. ACM SIGMETRICS Conference*, pp. 151–160, July 1998.

[7] M. E. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: Evidence and possible causes," in *Proc. of the ACM SIGMETRICS Conference on Measurement & Modeling of Computer Systems*, (Philadelphia), pp. 160–169, 1996.

[8] B. Mah, "An empirical model of http network traffic," in *Proc. INFOCOM'97*, Apr. 1997.

[9] R. Morris, "TCP Behavior with Many Flows," in *International Conference on Network Protocols*, Oct. 1997.

[10] M. Christiansen, K. Jeffay, D. Ott, and F. D. Smith, "Tunning RED for Web Traffic," in *Proc. ACM/SIGCOMM'00*, (Stockholm), 2000.