Drop Probability Assurance for Delay-insensitive Applications

Changhee Joo°, Jaesung Hong, and Saewoong Bahk
School of Electrical Engineering and Computer Science, Seoul National University
{cjoo, hjs, sbahk}@netlab.snu.ac.kr

***Abstract*—Loss differentiation is recommended as a service differentiation provided by an Assured Forwarding (AF) Per-Hop Behavior (PHB) in Differentiated Service (DiffServ) architecture. An Active Queue Management (AQM) technique is addressed as a suitable alternative to realize the service differentiation because the AF PHB should attempt to minimize long-term congestion while permitting short-term congestion in order to accommodate traffic bursts. In this paper, we introduce a desirable property of *assured drop probability*. We modify an existing AQM algorithm for the property so that it assures a target drop probability in a properly provisioned network. We evaluate it with other comparable schemes through simulation.**

I. INTRODUCTION

The DiffServ architecture is a scalable model to provide Quality of Service (QoS) in IP networks, especially in the Internet. Due to its simplicity and scalability, DiffServ has taken a considerable attention to achieve QoS in Internet. The DiffServ model introduces two packet-handling schemes based on PHBs besides the basic best-effort delivery mechanism used in the current Internet. The *Expedited Forwarding (EF)* PHB is used to build services that require low delay, low jitter, low loss, and assured bandwidth [1], while the *Assured Forwarding (AF)* PHB is used to build more elastic services that impose requirements only on throughput without any delay or jitter restrictions [2]. The idea behind the AF PHB is to differentiate packets by marking them based on the conformance to their target throughputs. Non-conformant packets are called out-of-profile while conformant packets are called in-profile.

The IETF recommends that an AQM technique be used to realize the multiple levels of drop precedence required in the AF PHB [1] because it can minimize long-term congestion while permitting short-term congestion to accommodate traffic bursts. Random Early Detection (RED) is one of the most well-known AQM schemes [3]. Stochastic packet drop allows RED to avoid global synchronization and bias against burst traffic. However, a high degree of sensitivity toward its operating parameters and unfairness to flows with different round-trip times prevent it from being widely deployed [4, 5]. Hence, it is not a good idea to employ a pure RED for each level of drop precedence.

Support for multiple levels of drop precedence also characterizes an AQM scheme for the AF PHB. An AQM scheme is expected to have two properties of *sheltering* and *load tolerance* [6]. A drop precedence level is said to be sheltered if higher drop precedence levels have a minor effect upon loss rate of the level. A queuing mechanism is said to be load-tolerant if it prevents starvation of high drop precedence traffic and preserves hierarchy among drop precedence levels. Both properties are required for AQM to provide service differentiation while preventing resources from being monopolized by the lower drop precedence level.

In this paper, we introduce another attractive property for an AQM scheme and propose a modification of Hybrid RED (HRED) [4], which overcomes weakness of the pure RED, for the property. The paper is organized as follow. Section II addresses related work. We introduce an *assured drop probability* property in section III. We discuss required modifications of AQM for the property and extend the modified AQM for the three-color AF PHB. We evaluate our proposal and compare it with RIO and WRED through simulations in section IV. We conclude our paper in section V.

II. RELATED WORK

The RED In and Out (RIO) scheme [7] was initially proposed as a basis for two levels of drop precedence. Drop probability of in-profile packets is calculated independently from out-of-profile packets. RIO is equivalent to running two pure REDs. One calculates drop probability of in-profile packets from queue length of in-profile packets, and the other calculates drop probability of out-of-profile packets from queue length of both in-profile and out-of-profile packets. RIO can be modified for three-color AF PHB [8].

Weighted RED (WRED) [9] is similar to RIO except that it has a single average of queue length. For a packet arrival, WRED updates the single average queue length without considering precedence levels of packets. However, multiple RED threshold parameters are maintained to provide service differentiation – one set for each precedence level.

In RIO and WRED, the parameters for different precedence levels can be set in three ways: overlapped, partially overlapped, and staggered [10]. As discussed in [6], the setting of multiple RED parameters affects the characteristic of AQM in RIO and WRED. Overlapped WRED should be modified to offer sheltering, and staggered RIO and WRED are hard to meet requirements for load tolerance. In this paper, overlapped WRED and RIO are evaluated for comparison because our proposal is closer to overlapped one than others.

WRED with Threshold (WRT) [6] is designed by combining RIO with WRED to achieve both sheltering and load tolerance. When an out-of-profile packet arrives, drop probability is calculated and applied like RIO. When an in-profile packet arrives, WRT calculates the average queue length of in-profile packets and compares the average with a threshold. If the average is less than the threshold, the packet is accepted, and otherwise, the packet is dropped with a probability that is calculated as WRED. WRT offers sheltering by calculating a separate average queue length for packets of low precedence level, and prevents starvation of high precedence level by calculating drop probability from queue length of all packets.

Apart from research about DiffServ, Morris proposed Flow-Proportional Queuing (FPQ) [11] that controls drop probability by varying router's queue length in proportion to the number of active TCP connections. Though FPQ produces more delay in queuing, it is observed that the average transfer delay is the same due to lack of TCP timeout, and transfer delays are more predictable and fairer. However, FPQ requires per-flow information to calculate the number of active TCP connections, and it is not designed for DiffServ architecture.

III. Assured Drop Probability

The AF PHB is designed to provide forwarding service that delivers IP packets with high probability as long as traffic does not exceed a profile. The excess traffic is acceptable but it is not delivered with as high probability as traffic in the profile. Keeping in mind the intention, it is reasonable that a target drop probability of a precedence level be assured in a properly provisioned DiffServ network. In that case, the drop probability for the level and the penalty for the excess become clear to users. In the remainder of paper, we assume three-color AF PHB and present each level of drop precedence as one of colors of Green, Yellow, and Red: Green refers to the lowest level of drop precedence, and Red refers to the highest.

The sending rate of the TCP protocol is governed by loss late and round-trip time (RTT) with a simple approximation [4, 10]. Since the bottleneck link is shared, the loss rate can be assured by limiting the sending rate of each connection or by controlling RTT. Per-flow information can be used to limit the sending rate of each connection, and buffer management can control RTT. Though pre-flow information is more useful for providing QoS of delay bound, bandwidth as well as drop probability, it is not acceptable in the DiffServ architecture due to scalability problem.

Buffer management can be used to lower down loss rate of TCP connections [11]. However, since the queuing delay significantly increases from additional buffers, it is suitable for delay-insensitive traffic like FTP, in which throughput is more important than delay or jitter.

Considering finite physical buffer of a router and hierarchy among colors, an AQM scheme is said to *assure drop probability* when it satisfies following two requirements,

- Drop probability of a color is less than its targeted drop probability if there are additional buffers.
- If a color has a larger drop probability than its target, drop probabilities of colors with higher precedence also exceed their target, and queues of colors with lower precedence do not occupy additional buffers unnecessarily.

Most AQM schemes control drop probability with given buffer space. We modify one of them to control buffer space rather than drop probability. Among a number of AQM schemes, we choose Hybrid RED, or simply HRED [4]. Its local stability is analytically proven and response time is designed to be independent of the amount of traffic changes.

```
/* get expected drop probability */
calculate p from qlen and Q_lim

/* adjust Q_lim from expected p */
if (p > p_ref & Q_lim < Q_max)
    increase Q_lim
else if (qlen < min_th & Q_lim > Q_min)
    decrease Q_lim

/* get new drop probability */
calculate p from qlen and Q_lim
```

Figure 1. Modified HRED Algorithm.

HRED is also easily configured to achieve demanded performance of queuing delay, jitter, and response time through its parameters.

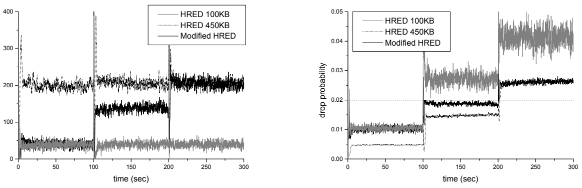
A. Modification of HRED

We modify HRED to adjust the size of buffer size Q_{lim} and corresponding threshold parameters max_{th} and min_{th}, which are configured in proportional to Q_{lim}, while regulating the drop probability p less than a target p_{ref}. The modified algorithm is shown in Fig. 1. The drop probability is calculated from current queue length *qlen* using the linear mapping function of HRED.

In controlling buffer space, we notice two restrictions: first, the link utilization should not be hurt by excessively small buffer space and secondly, buffer space is limited by the physical memory Q_{max}. Two restrictions increase complexity of the modified algorithm. For the first restriction, Q_{lim} is not allowed to get less than Q_{min} and for the second, the modified algorithm adjusts p_{min} and p_{max} exactly as HRED does when $Q_{lim} = Q_{max}$. Obeying two restrictions, Q_{lim} increases only when additional buffer is available ($Q_{lim} < Q_{max}$) and the drop probability needs regulation ($p > p_{ref}$) in Fig. 1. It should be noticed that $p > p_{ref}$ also implies from HRED algorithm that the queue length is larger than max_{th} ($qlen > max_{th}$) when $Q_{lim} < Q_{max}$. The second if-statement is used to decrease Q_{lim}. Even if it is possible that Q_{lim} decreases when $p > p_{ref}$, Q_{lim} will increase again on next packet arrival making the first if-statement true because decreased Q_{lim} is less than Q_{max}.

Fig. 2 illustrates behaviors of HREDs having the full buffer space of 100KB and 450KB, and the modified HRED (Q_{min} = 100KB, Q_{max} = 450KB) through simulation, in which the bottleneck link has capacity of 45Mbps and delay of 20ms. At traffic increase at 100 seconds, the modified HRED queues more packets by adjusting buffer space in order to assure the target drop probability 0.02 (a dashed horizontal line in Fig. 2 (b)). After another traffic increase at 200 seconds, it utilizes full buffer space and starts adjusting drop probability to not lose stability. HREDs of full buffer space keep queue length at a constant level as their aim.

For each time interval of 100 seconds, we measure transfer times of 20 packets. We omit the detailed results due to lack of space. Smaller buffer space contributes to decreasing transfer time by reducing queuing delay. However, when buffer space is not enough in heavy load, it is observed that some TCPs experience excessive delay due to timeout, and make a biased distribution of transfer time. Though average delays are all similar in the interval of 200~300 seconds, the 90th percentile of delay of HRED with 100KB is definitely larger than others. The modified HRED removes this bias by increasing buffer



(a) Queue Length (b) Drop Probability

Figure 2. Behaviors of HRED and Modified HRED.

space in heavy load, while keeping as small average transfer time as HRED with 100KB in light load.

B. *Extension for Three-color AF PHB*

In the DiffServ architecture, the AF PHB provides forwarding of IP packets in four independently forwarded AF classes and, within each AF class, IP packets are assigned one of three different levels of drop precedence [2].

We extend the modified HRED for the three-color AF PHB. The extension, named XHRED, has three modified HREDs for each color. Since each queue length of HREDs is decoupled, it is very similar with three independent HREDs running for each color. However, queues are related through *prioritized sharing* of a single physical buffer space, which improves resource utilization. The priority of sharing is given as drop precedence: a color with lower drop precedence has a higher priority.

When a packet arrives at the router, it is classified into one of three colors based on information of IP header. If expected drop probability of the color is larger than the target, queue of the color tries to get buffers from the extra buffer space, and if the attempt fails, it tries to preempt buffers from lower priority colors. If all attempts to get buffers turn out futile, it has no choice but increases drop probability over the target. Then, after additional buffers are allocated or drop probability increases over the target, the incoming packet is stochastically dropped with the lowest probability of it and its higher priority colors. The buffer release can be also traced similarly.

It should be noticed that packets in queues are not preempted although buffers can be preemptively allocated. This is accomplished by using virtual queues. XHRED has three virtual queues for each color and keeps a single output queue for transmission. It prevents out-of-ordered packets from different queues as well as preemption of queued packets.

IV. SIMULATION

In this section, we evaluate XHRED comparing it with overlapped RIO and WRED among the variants of RED. We use the ns-2 simulator [12].

The simulations are based on a dumbbell topology. All connections traverse a single bottleneck link with capacity 150Mbps. We assign a random round-trip time of 20 to 100 ms to each connection (uniform distribution), which includes everything except queuing delay at the bottleneck. One set of random RTT's is used in all simulations. Average packet length is 500 bytes. The bottleneck link queue operates in byte mode with buffer size of 400KB. Overlapped WRED and RIO are used with $min_{th} = max_{th}/2 = 100$KB. The weight for moving average is chosen automatically from the link capacity, and the maximum drop probability of each color (p^G_{max}, p^Y_{max}, p^R_{max}) is set to 0.02 (Green), 0.03 (Yellow) and 0.04 (Red). XHRED is configured with $min_{th} = max_{th}/2 = Q_{lim}$, $\kappa = 2$, $T_\alpha = 1.78\text{x}10^{-6}$, and $T_\beta = 7.12\text{x}10^{-6}$ for each color. The target drop probability of each color (p^G_{ref} ,p^Y_{ref} ,p^R_{ref}) is set as the same as p^G_{max}, p^Y_{max}, p^R_{max} of WRED and RIO.

For each colored-traffic, we monitored drop probability used in AQM, actual loss rate, and bandwidth share. We start our simulation with 100 persistent TCP connections for each color. After 100 seconds, we increase Red traffic. Yellow and Green traffic are also increased in order after another 100 seconds. Hence, for 300~400 seconds, there are total 600 TCP connections, which heavily overload the bottleneck link.

Fig. 3 illustrates drop probability of each color. The three horizontal lines present three targeted drop probabilities. Fig. 3 (a) and (b) appear as if WRED and RIO succeed in assuring target drop probabilities, but they do not take account of drop probability when average queue length is larger than max_{th}, in which case drop probability is 1. Actual loss rate shown in Fig. 4 confirms it. Green experiences larger loss rate than its drop probability in WRED, and so does Yellow and Red in RIO.

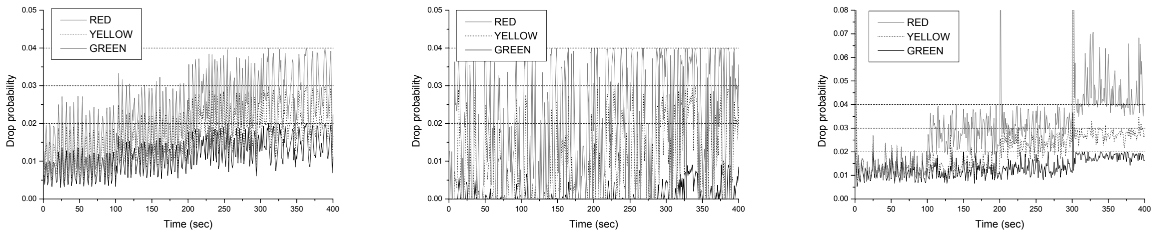
WRED offers sheltering in part and load tolerance but fails to assure drop probability. Accumulated bandwidth share of Fig. 5 (a) shows that traffic increase of Red at 100 seconds decreases bandwidth shares of Yellow and Green, and traffic increase of Yellow at 200 seconds decreases bandwidth share of Green. Though the amount of decreases is not significant in our simulation, it can be if Red traffic excessively increases. Again from Fig. 5 (a), neither Yellow nor Red starve when Green traffic increases at 300 seconds. However, Fig. 4 (a) shows that Green has larger loss rate than its target, but Yellow and Red achieve its target. It breaks the second condition for assured drop probability.

Though RIO provides sheltering, it is not load-tolerant. It is evident from Fig. 5 (b) that bandwidth share of Green is not affected by traffic increase of Red at 100 seconds nor Yellow at 200 seconds. However, accumulated bandwidth share also reveals that Red experiences starvation by Green. RIO does not assure drop probability either. Although it is not presented in our results, additional buffers are allocated to queues of Yellow and Green even if they already have less drop probabilities than their targets. This breaks the second condition for assured drop probability.

We can also observe that XHRED assures drop probability as well as offers sheltering and load tolerance. In Fig. 3 (c) and Fig. 5 (c), traffic increase of Red and Yellow at 100 seconds and 200 seconds has minor effects on drop probability and bandwidth share of Yellow and Green. Comparing with WRED, it is evident that XHRED offers better sheltering. Fig. 5 (c) also shows that Red no longer starve as RIO because queue of Red has at least its minimum buffer space Q^R_{min}. Finally, assured drop probability is observed in Fig. 3 (c) and Fig. 4 (c). The drop probability satisfies two conditions for assured drop probability in section III. Especially in heavy load of the last 100 seconds, XHRED assures drop probability of Green at the expense of drop probability of Red (over its target 0.04).

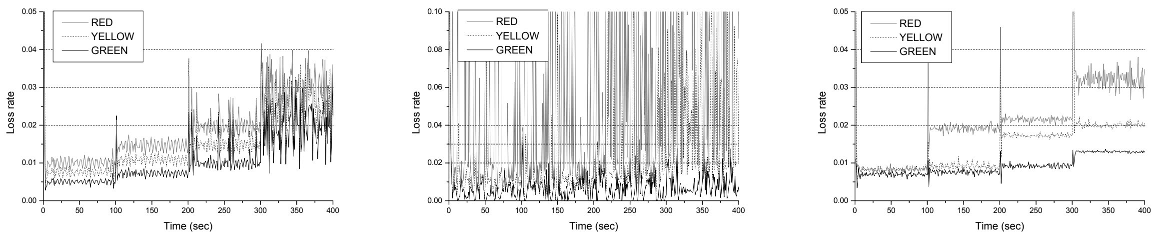
V. CONCLUSION

We propose an extended AQM scheme to assure drop probability in the Differentiated Service architecture while



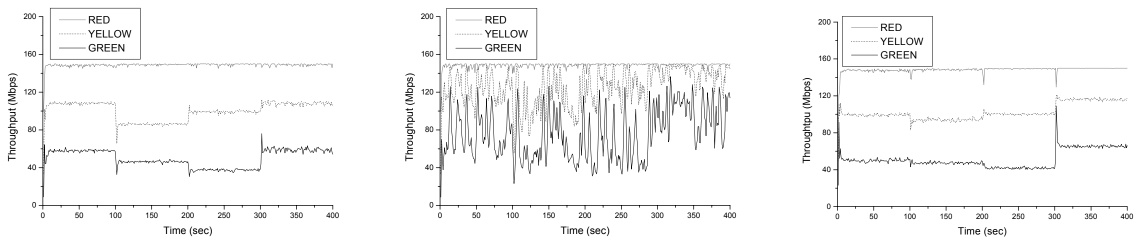
(a) WRED (b) RIO (c) XHRED

Figure 3. Drop probability. WRED and RIO can not have larger drop probability than p_{max} of each precedence level, while XHRED allows higher precedence level (Red) to over its reference when the link is overloaded.



(a) WRED (b) RIO (c) XHRED

Figure 4. Actual Loss rate. Only XHRED assures the loss rate for all precedence levels. WRED fails for low precedence levels (Green and Yellow), and RIO fails for high precedences levels (Yellow and Red).



(a) WRED (b) RIO (c) XHRED

Figure 5. Accumulated bandwidth share. In RIO, high precedence level (Red) experiences the starvation by low precedence level (Green).

offering sheltering and load tolerance. A differentiating queuing scheme is said to *assure drop probability* when it controls buffer space for target drop probabilities and it allocates and releases shared buffers with priority following levels of drop precedence.

We employ HRED as a proper AQM algorithm, modify it to control buffer space for a target drop probability, and extend it to support three levels of drop precedence (XHRED) without losing sheltering and load-tolerant property.

Throughout simulation, we compared XHRED with overlapped WRED and RIO observing that while RIO does not have load-tolerant property and WRED does not assure drop probability, XHRED assures drop probability as well as offers sheltering and load tolerance.

REFERENCES

[1] V. Jacobson, K. Nichols, K.Poduri, "An Expedited Forwarding PHB", RFC 2598, June 1999.

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

[3] S. Floyd and V. Jacobson, "Random Early Detection Gateways for congestion Avoidance", IEEE/ACM Transactions on Networking, vol.1, no.4, pp.397-413, August 1993.

[4] C. Joo, and S. Bahk, "Hybrid Random Early Detection", ISCC, 2003.

[5] M. May, J. Bolot, C. Diot, and B. Lyles, "Reasons not to deploy RED", IWQoS, March 1999.

[6] U. Bodin, O. Schelen, and S. Pink, "Load-tolerant Differentiation with Active Qeueu Management", Computer Communication Review, 2000.

[7] D. Clark and W. Fang, "Explicit Allocation of Best-effort Packet Delivery Service", IEEE/ACM Transactions on Networking, vol.1, no.4, pp.362-373, August 1998.

[8] M. Goyal, A. Durressi, R. Jain, C. Liu, and P. Misra, "Effect of number of Drop Precedences in Assured Forwarding", GLOBECOM, December 1999.

[9] "Distributed Weighted Random Early Detection", URL http://www.cisco.com/univercd/cc/td/doc/product/software/ios111/cc111/wred.htm

[10] R. Makkar et al., "Empirical Study of Buffer Management Schemes for Diffserv Assured Forwarding PHB", ICCN, May 2000.

[11] R. Morris, "Scalable TCP Congestion Control", INFOCOM, March 2000.

[12] The Network Simulator (NS), URL "http://www.isi.edu/nsnam/ns"