Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 30 (2007) 3859–3869
computer communications
www.elsevier.com/locate/comcom

Hop-count based probabilistic packet dropping: Congestion mitigation with loss rate differentiation

Xiaobo Zhou *, Dennis Ippoliti, Terrance Boult

Department of Computer Science, University of Colorado at Colorado Springs, CO 80918, United States

Received 19 January 2007; received in revised form 14 September 2007; accepted 19 September 2007
Available online 26 September 2007

Abstract

Network applications and users have very diverse service expectations and requirements, demanding for provisioning different levels of quality of service on the Internet. As the speed of network links has been rising at a pace that exceeds that of the growth in the buffer size, packet loss rate differentiation has been an active research topic. However, none of the existing packet dropping schemes for loss rate differentiation considered an important issue, that is, the retransmission overhead of dropped packets. In this paper, we design a hop-count based probabilistic packet dropper (HPPD) for congestion mitigation and loss rate differentiation. HPPD aims to meet a unique two-fold objective by two-dimensional loss rate differentiation: the primary one is the congestion mitigation that aims to reduce congestion in the first place by dropping intra-class packets differently based on their maturity levels to reduce retransmission cost; the other is inter-class proportional loss rate differentiation. The maturity level of a packet, the number of hops it has travelled, is inferred from its time-to-live value in the IP header. We propose a novel intra-class *n*th-root proportional dropping scheme. The scheme reduces retransmission cost by giving higher dropping probabilities to less mature packets while all packets have their forwarding chances. The *n* is a controllable parameter trading off dropping fairness for congestion mitigation. It provides great controllability to network operators. Simulation results show that HPPD can significantly mitigate the congestion by reducing the retransmission overhead of dropped packets and achieve the proportional loss rate differentiation at the same time.

Keywords: Quality-of-service differentiation; Congestion mitigation; Proportional model; Loss rate; Hop count

1. Introduction

There is an increasing demand of provisioning different levels of quality of service (QoS) on the Internet to support different types of network applications and various user requirements. Differentiated Services (DiffServ) is one of the major efforts to meet the demand [1]. It aims to provide differentiated services between classes of aggregated traffic flows within a router, rather than to offer QoS guarantees to individual flows. To receive different levels of QoS, packets are assigned with different service types or traffic classes at the network edges. DiffServ-compatible routers in the network core perform stateless prioritized packet forwarding or dropping, so called "per-hop behaviors" (PHBs), to the classified packets. Due to its per-class stateless processing, the DiffServ architecture exhibits a good scalability. There are two basic schemes to DiffServ provisioning. Absolute DiffServ aims to provide statistical assurances for a class's received performance measures, such as a minimum service rate or maximum delay. Relative DiffServ is to quantify the quality spacings between different classes. The proportional differentiation model is popular due to its proportionality fairness and predictability [3,4].

Packet delay and loss rate are two key QoS metrics considered in the DiffServ context. Both are caused by network congestion that arises when the incoming traffic is close to or exceeds the network router resources, i.e., link speed and

* Corresponding author. Tel.: +1 719 2623493.
E-mail address: zbo@cs.uccs.edu (X. Zhou).

0140-3664/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2007.09.010

buffer size. The likelihood of packet losses is a very important performance measure for most of Internet traffic, and it is especially important for peak workload control [3]. The past few years have witnessed that the speed of network links has been rising at a pace that exceeds that of the growth in buffer size [8]. As a result, packet loss rate differentiation in high workload situations has been an active research topic. There are a number of interesting differentiated buffer management and packet dropping schemes. PLR droppers in [3] aim to provide proportional loss rates to different traffic classes according to their pre-specified differentiation weights. JoBS in [15] extends the proportional loss rate model by providing both absolute loss and delay guarantees and proportional differentiations. The proportional differentiation constraint is relaxed to satisfy the absolute constraints when the two sets of constraints cannot be simultaneously satisfied. BRD dropper in [8] seeks to minimize the loss rate differences between traffic classes subject to the absolute loss constraints and the relative loss constraints. Those dropping schemes are able to effectively achieve their differentiation objectives. However, none of them considered an important issue, that is, the retransmission overhead of dropped packets.

A dropped packet might be retransmitted by protocols such as TCP or by end applications. Intuitively, dropping a packet which has travelled 20 hops results in more retransmission overhead and hence heavier congestion in networks than dropping a packet which has only travelled 2 hops. TCP's strategy is to control congestion once it happens, instead of trying to avoid congestion in the first place. An alternative to congestion control is congestion avoidance, which is to predict when congestion is about to happen and then to reduce the rate at which end nodes send data just before packets start being discarded. Previous studies have found that hop count distributions at gateways/routers are usually bell-shaped and the Gaussian distribution is a good first-order approximation; refer to [11] for the hop count distribution of a well-connected commercial server net.yahoo.com. To mitigate the congestion at the first place, our motivation is that packets should be given different dropping probabilities according to the number of hops they have travelled so as to reduce their retransmission overhead.

In this paper, we design a novel hop-count based probabilistic packet dropper (HPPD). The hop count of a packet is inferred from its time-to-live (TTL) value. HPPD provides two-dimensional loss rate differentiation. Within a traffic class, a less mature packet, which has a lower hop count value, has higher probability to be dropped than a more mature packet so as to reduce the retransmission overhead for dropped packets. We refer to this as *intra-class loss differentiation*. Between traffic classes, a packet from a low priority class has higher probability to be dropped than a packet from a high priority class. We refer to this as *inter-class loss differentiation*. Thus, HPPD aims to meet a two-fold objective: the primary one is the congestion mitigation that tries to reduce congestion in the first place by reducing retransmission overhead of dropped packets; the other is proportional loss rate differentiation for DiffServ provisioning. DiffServ is concerned with PHBs, since it is stateless. However, TTL brings some global state of a packet to the routers. The uniqueness of our work lies in the use of hop count information inferred from TTL in making differentiated packet dropping decisions so as to achieve congestion mitigation and loss rate differentiation at the same time.

There are two important considerations to the success of HPPD, i.e., *congestion mitigation*, and *fairness*. A simple way to reducing retransmission overhead is to drop packet strictly according to their maturity levels. A packet with a lower hop count will be always dropped before a packet with a higher hop count if there is backlogged one. However, this kind of strict priority dropping may lead to starvation for packets from neighbor routers. A feasible scheme should give packets with different hop count different chances to be forwarded. We propose a novel intra-class nth-root proportional dropping scheme for HPPD. The n is a controllable parameter trading off dropping fairness for congestion mitigation. Simulation results show that HPPD can significantly mitigate network congestion by reducing the retransmission overhead of dropped packets and achieve the proportional loss rate differentiation at the same time.

The structure of the paper is as follows. In Section 2, we review existing loss rate differentiation schemes. Section 3 presents the HPPD dropping strategies. Section 4 presents the design and implementation issues. Section 5 focuses on the performance evaluation. Section 6 concludes the paper with remarks in future work.

2. Related work

Congestion control and avoidance has been studied extensively in computer communications and networks. Early studies in packet networks included slow start [10], early random drop [7], and random early detection [6]. There are recent studies in buffe management for congestion mitigation and avoidance in wireless sensor networks [2,5,9]. There are also recent studies that propose to utilize the loss information to design robust active queue management [13,19]. Our approach is different that it utilizes the hop count information, inferred from TTL information, to execute loss rate differentiation so as to reduce the retransmission cost of dropped packets and mitigate the congestion in networks. It is complementary to previous work on congestion mitigation and control.

Loss rate differentiation in packet networks is an active research topic. Representative differentiated packet dropping schemes include PLR(∞), PLR(M), JoBS, and BRD. PLR(∞) and PLR(M) [3] were proposed for providing proportional loss rate differentiation. A packet class is assigned a loss differentiation parameter. The PLR schemes adjust the normalized loss rates so that they are eventually

equal for all classes and proportional loss rate differentiation is achieved. Two schemes differ in the time interval over which the loss rates are measured and proportionally adjusted. For PLR(∞), the loss rate of a class is estimated based on a long history of packet arrivals. This makes it less adaptive for changing class load. In PLR(M), the loss rate is measured over last M arrived packets.

JoBS in [15] extends the proportional loss rate model by providing both absolute loss and delay guarantees and proportional differentiations. The loss ratio and delay differentiation are combined into a cost function. The service rate of a class is adjusted in order to minimize the cost function. The proportional differentiation constraint is relaxed to satisfy the absolute constraints when the two sets of constraints cannot be simultaneously satisfied. A non-linear programming algorithm was proposed to solve the optimization problem on every packet arrival. It significantly increases the overhead of the approach.

BRD dropper in [8] seeks to minimize the loss rate differences between traffic classes subject to the absolute loss constraints and the relative loss constraints. To reduce the packet dropping overhead, BRD utilizes a random drop mechanism for loss rate differentiation. In the mechanism, the loss ratio is calculated for every sampling period. At the end of each sampling period, the loss probability of every class is calculated for next sampling period. Upon arrival of each packet, it is randomly dropped according to its dropping probability calculated from last sampling period. Our work aims to achieve congestion mitigation and proportional loss rate differentiation at the same time and therefore extends the scope of those previous efforts.

Loss rate differentiation aside, delay differentiation has been studied extensively in packet networks and end-point computer systems. Many algorithms were proposed for providing proportional delay differentiation (PDD) services. The main objective is to keep the ratio of average delay of a higher priority class to that of a lower priority class equal to the pre-specified value. Existing algorithms can be classified into two categories: rate-based, as exemplified by BPR [4] and JoBS [15], and time-dependent priority based, as exemplified by WTP [4], A-WTP [14], PAD [4], HPD [4], MDP [16], and LAD [20]; see [23] for a taxonomy. There are also many efforts for service differentiation on Internet and multimedia servers; see [17,24] for representatives.

In packet forwarding, a router discards a packet if its TTL value reaches zero. The work in [21] proposed heuristic QoS-aware routing algorithms which essentially divide the end-to-end path into at most two super-edges that are connected by a relay node. Routers that lie on the same super-edge use either the cost metric or the delay metric to forward packets. A packet will be forwarded along the least-cost path if the path can deliver the packet before its due-date, inferred from the TTL field. Otherwise, the router will try the least-delay path, or the packet will be dropped or forwarded with no guarantees [21]. Recently, TTL value is also used for security purposes. The work in [11] proposed a novel hop-count filtering scheme as a defense against spoofed DDoS traffic. The rationale is that an attacker can forge any field in the IP header, but not falsify the number of hops an IP packet takes to reach its destination. Using a mapping between IP addresses and their hop-counts to an Internet server, the server can distinguish spoofed IP packets from legitimate ones. The hop-count information is inferred from the TTL value in the IP header.

3. HPPD: a hop-count based probabilistic packet dropper

3.1. Inter-class proportional packet dropping scheme for diffserv provisioning

For DiffServ provisioning, packets are classified into multiple classes in the network edges according to their desired QoS levels. DiffServ-enabled network core routers perform prioritized packet forwarding and dropping. An effective relative DiffServ scheme must satisfy two basic properties: *predictability* and *controllability*. Predictability requires that higher classes receive better or no worse service quality than lower classes, independent of the class load distributions. Controllability requires that the network operators contain a number of controllable parameters that are adjustable for the control of quality spacings between classes. An additional requirement is *fairness*. Fairness is a quantitative extension of the predictability, which describes how much better QoS metric received by a class compared with that received by another class. For example, proportionality is a popular fairness metric.

The proportional differentiation model has been widely accepted as an important relative DiffServ model because of its inherent differentiation predictability, controllability, and proportionality fairness [3,4]. It states that the quality spacing between class i and class j to be proportional to their pre-specified differentiation parameters δ_i and δ_j, that is, $q_i/q_j = \delta_i/\delta_j,\ 1 \leqslant i,\ j \leqslant N$ where q_i and q_j are the QoS factor of class i and class j, respectively. So it is up to network applications and clients to select appropriate QoS levels in terms of differentiation parameters that best meet their requirements, cost, and constraints.

HPPD is a probabilistic packet dropper. Its basic idea is to divide the dropping process into a sequence of short periods. In each period, based on the measured resource utilization and the predicted workload, the packets are dropped with different probabilities. HPPD inter-class dropping scheme aims to control the ratios of the average loss rate of classes based on their normalized differentiation parameters. Let L_i denote the total loss probability of packets in class i. HPPD requires that the ratio of total loss probability of class i to class j is fixed to the ratio of their differentiation parameters

$$\frac{L_i}{L_j} = \frac{\delta_i}{\delta_j} \qquad 1 \leqslant i, \quad j \leqslant N. \tag{1}$$

The differentiation predictability requires that higher classes receive better services, i.e., lower loss rate. Without loss of generality, we assume that class 1 is the'highest class' and set $0 < \delta_1 < \delta_2 < \cdots < \delta_N$. Let λ_i denote the packet arrival rate of class i, which is normalized based on the router's forwarding capacity. When the total arrival rate exceeds the forwarding capacity (C), we have

$$\sum_{i=1}^{N} \lambda_i L_i = \sum_{i=1}^{N} \lambda_i - C. \tag{2}$$

The set of equations (1), together with the dropping constraint (2), lead to a linear equation system. It follows the HPPD inter-class probabilistic dropping scheme as

$$L_i = \frac{\delta_i \left(\sum_{i=1}^{N} \lambda_i - C\right)}{\sum_{i=1}^{N} \delta_i \lambda_i}, \qquad 1 \leqslant i \leqslant N. \tag{3}$$

3.2. Intra-class nth-root proportional packet dropping scheme for congestion mitigation

To reduce the retransmission overhead of dropped packets, HPPD provides intra-class loss rate differentiation to packets with different hop counts. Let k denote the hop count of a packet when it arrives at the router; $p \leqslant k \leqslant q$ where p and q are the lower bound and the upper bound of the hop-count, respectively. Let $f_i(k)$ be the probability of a packet from class i with hop count k, and ℓ_i^k be the dropping probability of a packet from class i with hop-count k. For packet class i, given its overall dropping probability L_i, we have the intra-class dropping constraint

$$\sum_{k=p}^{q} f_i(k) \ell_i^k = L_i, \qquad 1 \leqslant i \leqslant N. \tag{4}$$

The feasibility of the intra-class dropping scheme is important to the success of HPPD. As mentioned above, a simple way to reducing retransmission overhead is to drop packet strictly according to their maturity levels. A packet with a lower hop count will be always dropped before a packet with a higher hop count. However, this strict priority dropping may lead to starvation for packets from neighbor routers. A feasible scheme should give packets with different hop counts fair chances of forwarding while giving differentiated packet dropping probabilities according to hop count values. One possible approach is an intra-class proportional dropping scheme, i.e., $\ell_i^{k_1}/\ell_i^{k_2} = k_2/k_1$, $p \leqslant k_1$, $k_2 \leqslant q$. However, our preliminary experiments found that this intra-class proportional dropping scheme could only gain very limited retransmission saving because there are only few packets with low hop count values due to the bell-shaped hop count distribution. The scheme is also lack of controllability for network operators.

We propose a novel nth root proportional intra-class dropping scheme. We find a function $g(x)$ that offers a desirable intra-class dropping property. That is, the dropping probability of a packet maintains high before its hop count approaches the mean of the hop-count distribution, and maintains low when its hop count is greater than the mean. The function $g(x)$ is given as

$$g(x) = \begin{cases} (m-x)^{1/n} & \text{if } x \leqslant m, 1 \leqslant n; \\ -(|m-x|)^{1/n} & \text{if } m \leqslant x, 1 \leqslant n; \end{cases} \tag{5}$$

where x is hop count variable, and m is the mean of the hop count distribution. Since the dropping probability should be positive, we normalize it by $(q-m)^{1/n}$ and have a function $g'(x) = g(x) + (q-m)^{1/n}$. It leads to the intra-class nth-root proportional scheme. That is

$$\frac{\ell_i^{k_1}}{\ell_i^{k_2}} = \frac{g'(k_1)}{g'(k_2)}, \qquad p \leqslant k_1, \quad k_2 < q. \tag{6}$$

The rationale is its feasibility, controllability, differentiation predictability, and nth-root proportional fairness. Interestingly, when $n = 1$, the scheme is reduced to a proportional scheme. Actually, n is a controllable parameter trading off dropping fairness for congestion mitigation. It provides nice controllability to network operators. Note that when a packet with hop count q (the upper bound) arrives at a router, its dropping probability is either zero (the packet reaches its destination) or 1 (the packet expires). Fig. 1 shows the intra-class dropping probability of packets with different hop counts due to a uniform loss scheme, a proportional loss scheme, and the nth-root proportional loss scheme ($n = 3$) when the workload is 125%. Note that the overall loss rate (dropping probability) is 20% of the workload.

The set of equation (6), together with the dropping constraint (4), lead to a linear equation system. It follows the intra-class packet dropping probability is calculated as

$$\ell_i^k = \frac{g'(k) L_i}{\sum_{j=p}^{q} f_i(j) g'(j)}. \tag{7}$$

Note that the calculated loss probability could be greater than 1 for packets with low hop count values if the overall

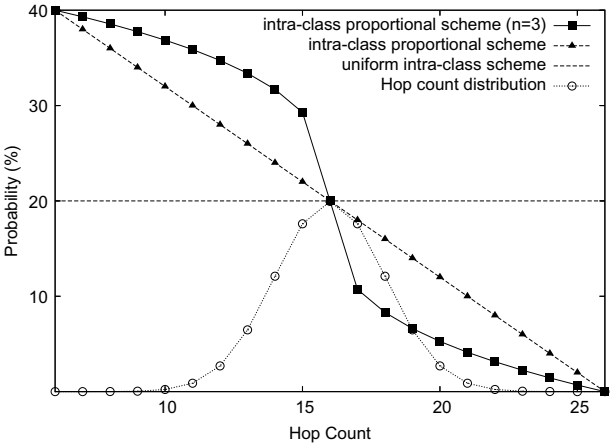


Fig. 1. Loss probability of packets due to the intra-class dropping schemes.

workload is high enough. The dropping scheme is reduced to strict priority dropping for those packets.

If a packet with hop count k is dropped by the router, its retransmission cost is k hops. Because packets from different classes have different differentiation weights, the function of the overall retransmission cost due to a dropping scheme is

$$\text{Cost} = \sum_{i=1}^{N} \delta_i \cdot \lambda_i \sum_{k=p}^{q} k f_i(k) \ell_i^k. \quad (8)$$

4. Design and implementation issues

4.1. Calculation and estimation of hop count distributions

The hop-count value of an arrived packet is inferred from its TTL value. TTL is an 8-bit field in the IP header. TTL was originally introduced to specify the maximum lifetime of a packet in the Internet. Each intermediate router decrements the TTL value of an IP packet by one before forwarding it to the next-hop router. If the TTL reaches zero, the packet will be dropped without forwarding. The hop-count value of the packet is therefore calculated as the initial TTL subtracted by the current TTL value. However, a technical issue is that there is no consensus on the initial TTL value. Different operating systems may set the initial TTL value differently. Fortunately, most modern operating systems use only a few initial TTL values, 30, 32, 60, 64, 128, and 255 [18]. These initial TTL values cover most of the popular operating systems, including MS Windows, Mac OS, Linux, SUN OS, Solaris, and many other commercial Unix systems. The work in [11] proposed a novel way to approximate hop-count of a packet according to its TTL value. It determines the initial TTL value of a packet by selecting the smallest initial value in the set that is larger than the TTL when the packet arrives at the router. Thus, the hop count of the packet can be calculated and its distribution is estimated in each simulation period. Note that it is even possible that the hop count information will be available in a packet header in the future Internet.

4.2. Estimation of request arrival rate

The request arrival rate of each class (λ_i) is estimated by counting the number of packets from each class occurring in a moving window of certain immediate past periods. The moving window estimation approach has been used in many similar experiments [8,23,24]. A smoothing technique based on a decaying function is applied to take weighted averages over past estimates. We have also examined the performance of the moving window method with different window size settings and other prediction methods, such as the exponential moving average method. Because the results show no qualitative differences, we only present those due to the moving window method with size as 5.

4.3. Loss rate differentiation feasibility

The PLR schemes in [3] drop a packet whenever the buffer of a router is full. However, limiting dropping decisions to such cases also limits the differentiation flexibility. PLR schemes maintain multiple queues, one queue per a class. When a packet is to be dropped, the schemes look for the head-of-line packet in a specific queue. However, when the workload is light, there may not exist such a packet in the specific queue. Thus, there is a feasibility issue in PLR schemes. HPPD dropper does not have this feasibility issue. Its dropping scheme is similar to the random early drop approach for congestion avoidance [6] and to the BRD dropper in [8]. An arriving packet is randomly dropped with a probability that depends on its traffic class, its hop-count value, the buffer state, the loss requirements, and the input traffic intensities of all traffic classes.

4.4. Differentiation overhead

A multi-queue system introduces greater implementation complexity than a single-queue system. As the work in [8], HPPD dropper uses a single-queue system. Furthermore, our scheme drops packets on their arrivals only, which is easy to be implemented compared to the push-out technique used in the PLR schemes [3]. The loss rate probability of an arriving packet is calculated according to (7). The calculation is done in the end of each simulation period and used for lookup in the next period. In our simulation model, the period size is the processing time of 1000 packets.

5. Performance evaluation

We built a Modular Network Simulation Tool (MoNSTer) for the evaluation of the HPPD dropping schemes. Its design was modelled following the design of the Click software router [12]. The simulator consists of a number of packet generators, hop count generators, waiting queues, an arrival rate predictor, a hop count distribution estimator, and a dropping probability table. The simulation process was divided into a sequence of short periods and performs packet dropping based on the dropping probability table in each period.

As the work in [11], we assumed that hop count distributions follow the bounded Gaussian distribution. Note that we are not making definitive claim that if hop-count distributions are Gaussian or not. Instead, we are interested in using the Gaussian distribution to study if HPPD can utilize the intra-class loss differentiation to achieve congestion mitigation by saving retransmission cost of dropped packets and further study the impact of various hop-count distributions on the performance of HPPD. As the work in [3,8], in our simulation, the packets have the same length

and the packet transmission time is normalized to one time unit. The traffic is generated from Pareto sources, one for each class, with a shape parameter $\alpha = 1.5$. We note that we have experimented with variable packet length and we have not found any qualitative differences. Each representative result reported in this section is an average of 200 runs. In the following, we study the impact of HPPD intra-class loss differentiation on congestion mitigation due to retransmission saving and the impact of HPPD inter-class loss differentiation on DiffServ provisioning with respect to loss rate. The evaluation focuses on the short-term high workload scenarios. Handling sudden spike or flash crowds is an ubiquitous problem for Internet services. Networks usually over-provision their bandwidth and CPU to handle the spike load. However, these networks sometimes still face unexpectedly high workloads during unforeseeable events such as terror attacks and Mars landing. Therefore, as others work in [15], this work studies the capability of HPPD under short-term spike workloads. The approach itself is also feasible in other workload situations in which congestion occurs.

5.1. HPPD Intra-class loss differentiation on congestion mitigation

Figs. 2 and 3 show the impact of the HPPD intra-class nth-root proportional dropping scheme on congestion mitigation when $n = 3$ and $n = 10$, respectively. The experiments examine the HPPD performance with 20 different bell-shaped hop-count distributions and the uniform distribution. As in [11], for the bell-shaped hop-count distributions, the lower bound and the upper bound of the distributions are 6 and 26, respectively. The mean of the distributions is 10, 13, 16, and 19. The variance, the girth of the distributions, varies from 2 to 6. A hop count distribution M16-V3 means that the mean is 16 and the variance is 3. Both figures illustrate the effectiveness of the HPPD approach because the simulated results closely agree with

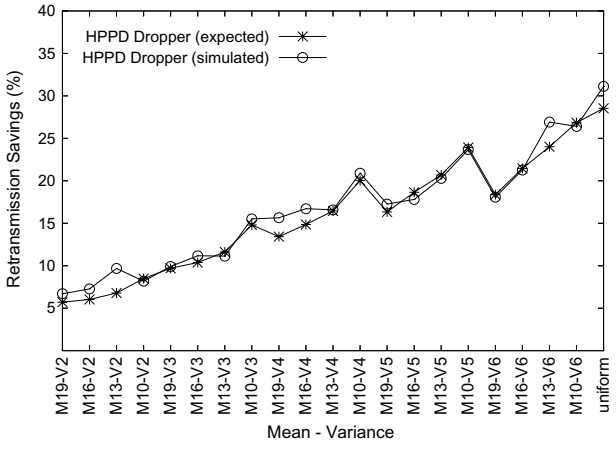


Fig. 2. The impact of HPPD intra-class loss differentiation on congestion mitigation ($n = 3$).

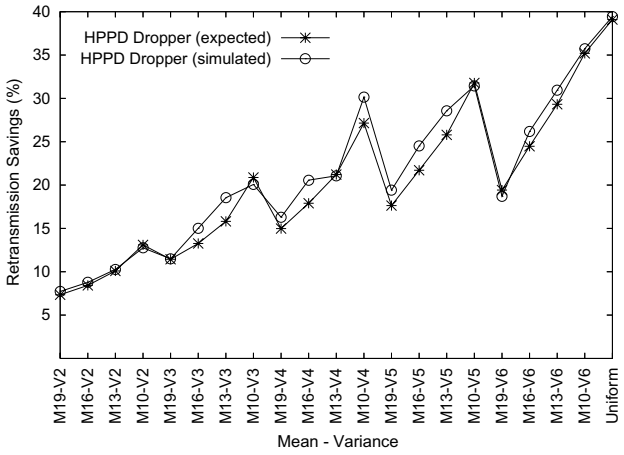
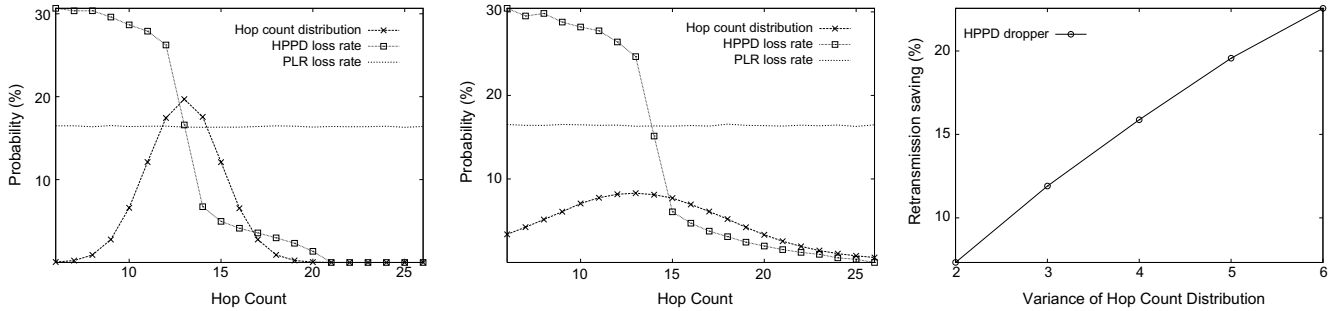


Fig. 3. The impact of HPPD intra-class loss differentiation on congestion mitigation ($n = 10$).

the expected results under various load conditions. More importantly, we found that the HPPD intra-class dropping scheme is able to reduce the overhead of retransmission nearly 25% when $n = 3$ and 35% when $n = 10$, compared to PLR droppers [3] and BRD dropper [8]. Both savings occur with the hop count distribution M10-V6. In case of the uniform distribution (though unrealistic), the saving would be about 30% and 40%, respectively. The saving is defined as $(\mathrm{Cost}_{\mathrm{UNI}} - \mathrm{Cost}_{\mathrm{HPPD}})/\mathrm{Cost}_{\mathrm{UNI}}$ where $\mathrm{Cost}_{\mathrm{UNI}}$ stands for the cost of any uniform intra-class dropping schemes such as PLR [3] and BRD [8] and the cost function is given by (8). Note that the saving is achieved in a single hop when it is overloaded. The impact on congestion mitigation will be accumulated polynomially in a route with multiple hops. In the experiments, we also found that the percentage of retransmission cost saving is independent on the workload situations, but on the hop count distributions.

We next study the impact of the mean and the variance of the hop count distributions on the performance of HPPD and the impact of the parameter of HPPD on congestion mitigation. Fig. 4 shows the impact of the variance of hop-count distributions on the performance of HPPD. The experiment was conducted when the incoming workload was 120% of the outgoing link capacity of a network node. Fig. 4(a) shows the loss probability of packets with different hop counts. The mean and the variance of the hop-count distribution are 13 and 2, respectively. We can see that packets with hop-count values less than the mean received higher dropping probability than those with hop-count values greater than the mean. Meanwhile, all packets got chances of forwarding. We next vary the variance of the hop-count distribution to 5. Fig. 4(b) shows that as the variance increases, the bell-shaped hop-count distribution is more flat. It means that the percentage of packets with hop count values lower than the mean increases and HPPD achieves more retransmission cost saving by giving higher dropping probabilities to packet with lower hop



(a) Loss probability (variance = 2). (b) Loss probability (variance = 5). (c) Impact of variance (mean = 16).

Fig. 4. Impact of hop-count variance on loss probability and congestion mitigation ($n = 3$).

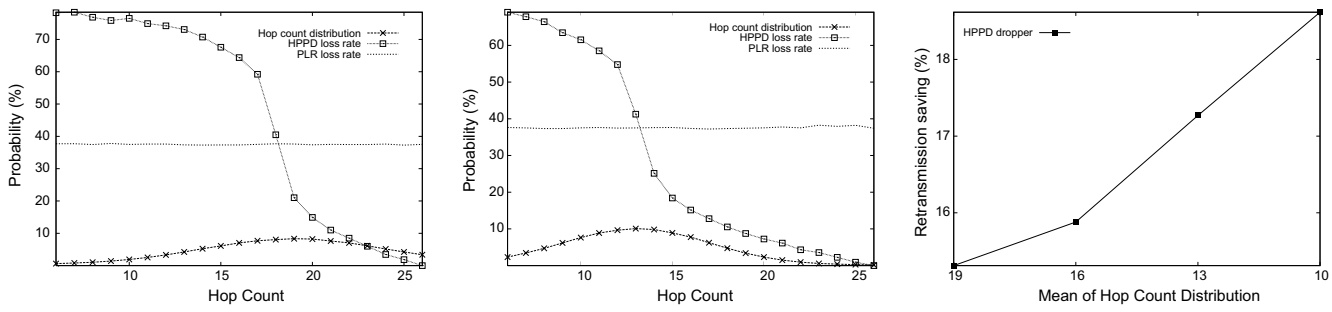
count values. Fig. 4(c) gives the correlation between the variance of hop-count distribution and the retransmission saving. The variance changes from 2 to 6 while the mean is fixed to be 16. We find that the retransmission saving increases as the variance increases. The result is explained by the fact that as the variance increases, the bell-shape of hop-count distribution is more flat. The percentage of packets with low hop count values increases. This gives the nth-root proportional dropping scheme more room to drop packets with low hop counts and increases the retransmission saving.

Fig. 5 shows the impact of the mean of hop-count distributions on the performance of HPPD. The workload is 160%. Fig. 5(a) shows the loss probability of packets with different hop counts. The mean and the variance of the hop-count distribution are 16 and 5, respectively. We next change the mean to 13. Fig. 5(b) shows that as the mean decreases, the percentage of packets with low hop-count values increases. Thus, HPPD achieves more retransmission cost saving by giving higher dropping probabilities to packet with lower hop count values. Fig. 5(c) depicts the correlation between the mean of the hop-count distribution and the retransmission saving. The mean changes from 19 to 10 while the variance is fixed to be 4. We find that the retransmission saving increases as the mean decreases. The result is explained by the fact that as the mean decreases, the percentage of packets with low hop count values increases. This gives the nth-root proportional dropping scheme more room to drop packets with low hop count values and increases the retransmission saving for congestion mitigation.

Previous experiments set n to 3 for the HPPD nth-root proportional dropping scheme. We next study the impact of n on loss probability and congestion mitigation. Fig. 6 shows that the saving of retransmission cost increases as n increases. The overall workload was 160% of the link capacity. The mean and the variance of the hop count distribution are 16 and 4, respectively. Fig. 7 illustrates the reason. As n increases, the packets with hop count values less than the mean got higher dropping probability and the packets with hop count values less than the mean got lower dropping probability. The retransmission saving is due to the various degree of intra-class differentiation. Due to the property of nth-root proportional function, the gain due to the incremental of n decreases as n increases. The results show that parameter n is a controllable parameter for congestion mitigation. Essentially, it is a tradeoff between congestion mitigation and the fairness of packet dropping. It provides great controllability to network operators.

5.2. HPPD Inter-class loss rate differentiation on diffserv provisioning

In the following, we study the HPPD inter-class dropping scheme for proportional DiffServ provisioning with respect to loss rate. Fig. 8 depicts the experienced loss rate



(a) Loss probability (mean = 19). (b) Loss probability (mean = 13). (c) Impact of mean (variance = 4).

Fig. 5. Impact of hop-count mean on loss probability and congestion mitigation ($n = 3$).

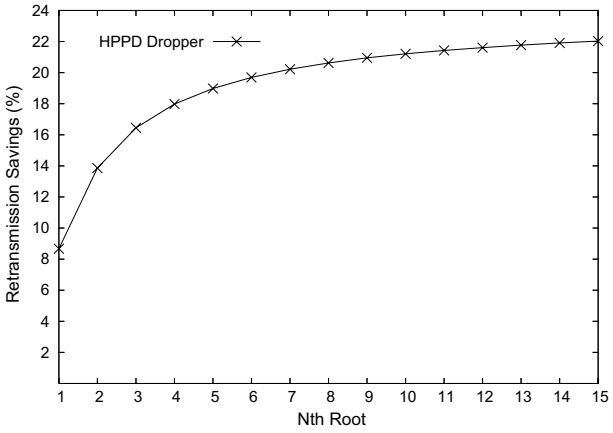


Fig. 6. Impact of n on retransmission saving.

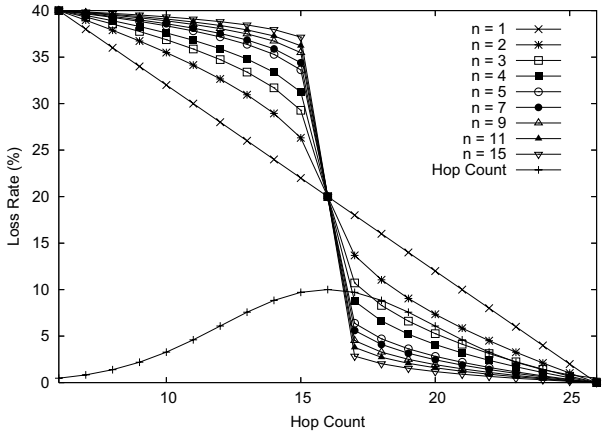


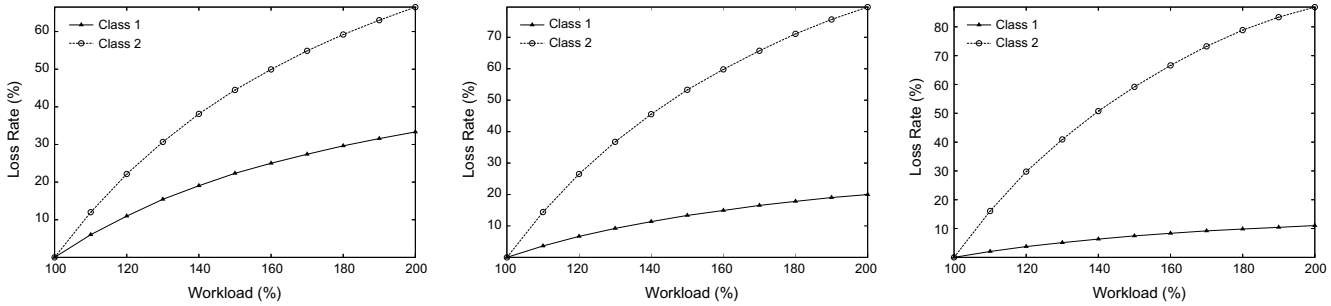
Fig. 7. Impact of n on loss probability.

of two classes due to the HPPD inter-class dropping scheme under various load conditions. Without loss of generality, let Class 1 be the high priority class and Class 2 be the low priority class. The arrival rate ratio of two classes ($\lambda_1 : \lambda_2$) is fixed to be 1:1. The differentiation weight ratio of two classes ($\delta_1 : \delta_2$) varies as 1:2, 1:4, and 1:8. The workload changes from 100% to 200%. The mean and the variance of the hop count distribution are 16 and 4, respectively. The results show that HPPD can achieve proportional loss rate differentiation at different workload conditions. We also note that the inter-class differentiation of HPPD is independent on the hop count distribution, which means that HPPD is able to achieve predictable and consistent inter-class differentiation and mitigate congestion by intra-class differentiation at the same time.

To give more sensitivity analysis of the HPPD inter-class dropping scheme, we vary the arrival rate ratio of two classes. Fig. 9 depicts the experienced loss rate of two classes. The overall workload is 120% of the link capacity. The traffic of class 1 contributes 10–90% of overall workload. The differentiation weight ratio of two classes ($\delta_1 : \delta_2$) varies as 1:2, 1:4, and 1:8. It shows that HPPD dropper can achieve proportional loss rate differentiation as well, independent of workload variations.

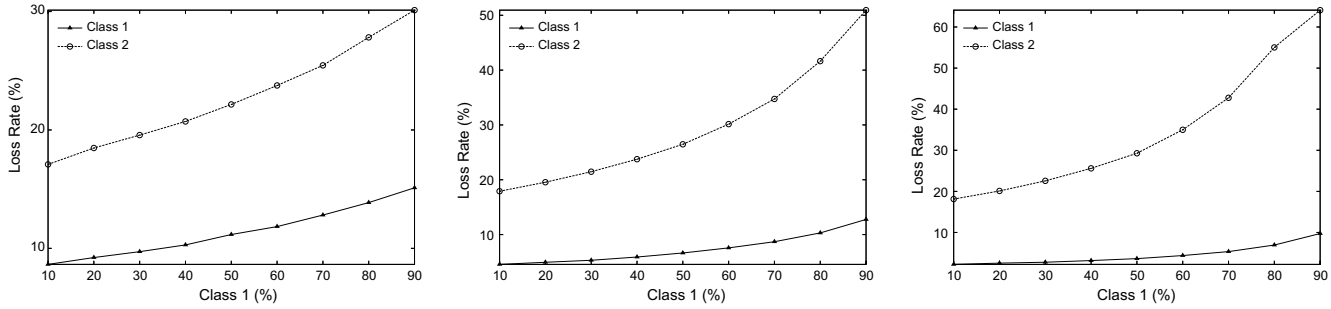
We then use a three-class workload to address the sensitivity of the HPPD inter-class dropping scheme to the number of classes. Note that the number of classes for DiffServ provisioning is often limited, varying from 2 to 3 [4,8,15,23,24]. Fig. 10(a) and (b) depict the experienced loss rate of three classes when their arrival rates are fixed to be same. The overall workload varies from 100% to 200% of the link capacity. The differentiation weight ratio ($\delta_1 : \delta_2 : \delta_3$) is set to 1:2:4 and 1:2:6. The results show that HPPD can achieve predictable proportional loss rate differentiation with three classes. We then vary the arrival rate ratio of three classes. Fig. 10(c) depicts the experienced loss rate of three classes. The overall workload is 150% of the link capacity. The traffic of class 1 contributes 10–50% of overall workload. The differentiation weight ratio of three classes $\delta_1 : \delta_2 : \delta_3$ is 1:2:4. It shows that HPPD can achieve predictable proportional loss rate differentiation as well, independent of classes' workload variations.

Finally, we discuss the effectiveness of the HPPD inter-class dropping scheme by investigating its differentiation robustness and short-term behaviors. Fig. 11 quantitatively depicts the variance of the loss rate of two classes ($\delta_1 : \delta_2 = 1 : 2$) and Fig. 12 depicts that of three classes ($\delta_1 : \delta_2 : \delta_3 = 1 : 2 : 4$). At each workload condition, the upper line is the 95th percentile; the bar is the mean; and the lower line is the 5th percentile. The results demonstrate the differentiation robustness of the HPPD inter-class dropping scheme. Figs. 13 and 14 fur-



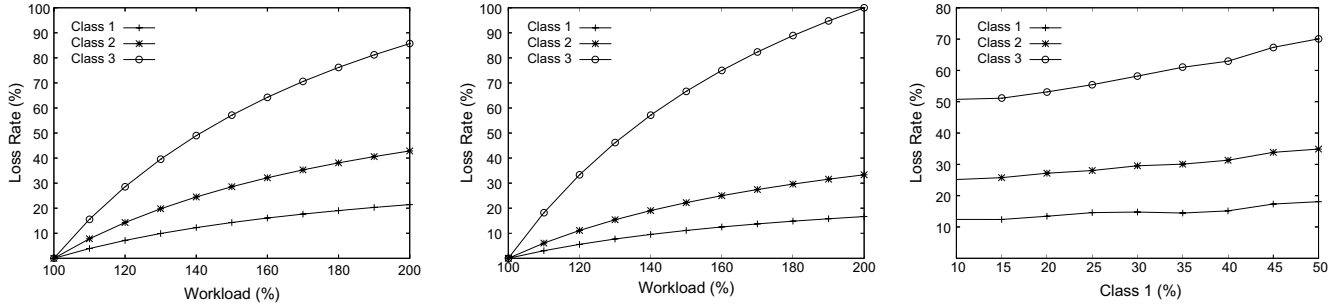
(a) $\delta_1 : \delta_2$ = 1:2. (b) $\delta_1 : \delta_2$ = 1:4. (c) $\delta_1 : \delta_2$ = 1:8.

Fig. 8. HPPD two-class proportional loss rate differentiation with fixed class workloads.



(a) $\delta_1 : \delta_2$ = 1:2. (b) $\delta_1 : \delta_2$ = 1:4. (c) $\delta_1 : \delta_2$ = 1:8.

Fig. 9. HPPD two-class proportional loss rate differentiation with different class workloads.



(a) $\delta_1 : \delta_2 : \delta_3$ = 1:2:4. (b) $\delta_1 : \delta_2 : \delta_3$ = 1:2:6. (c) $\delta_1 : \delta_2 : \delta_3$ = 1:2:4.

Fig. 10. HPPD three-class proportional loss rate differentiation with fixed and different class workloads.

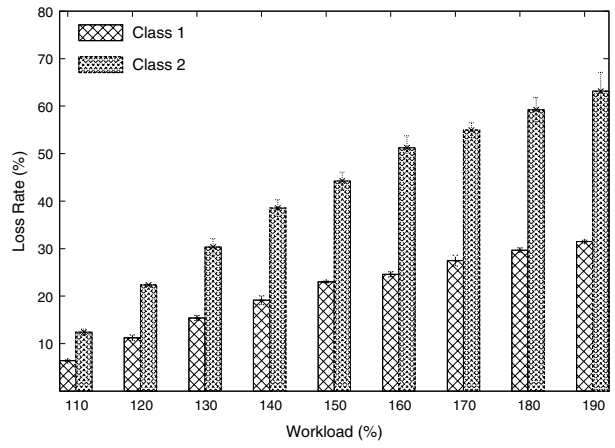


Fig. 11. Two-class DiffServ robustness.

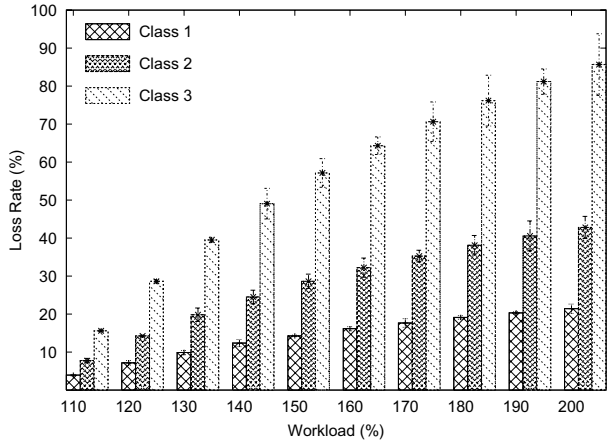


Fig. 12. Three-class DiffServ robustness.

ther show a short-term view of the loss rate of the two-class case and the three-class case, respectively. The experiment was run for 100 sampling periods for warming up and then the data was collected for 50 sampling periods at each of three workload conditions. For the first 50 periods, the system ran at 160% capacity. It was then lowered to 130% capacity. At the 100th period, the workload was increased back to 160%. A loss rate is measured for each class in a sampling period, i.e., every 1000 packet arrivals. Obviously, the HPPD inter-class scheme achieves consistent DiffServ results during different sampling periods.

We performed a wide range of sensitivity analysis. We varied the sampling period size, the number of classes, the arrival rate ratio of the classes, and the differentiation weight ratio of the classes. We note that we did not reach any significantly different conclusions regarding to the congestion mitigation capability and the loss rate differentiation controllability of HPPD.

6. Conclusion

Packet loss rate differentiation has been an active research topic in the DiffServ context. However, none of

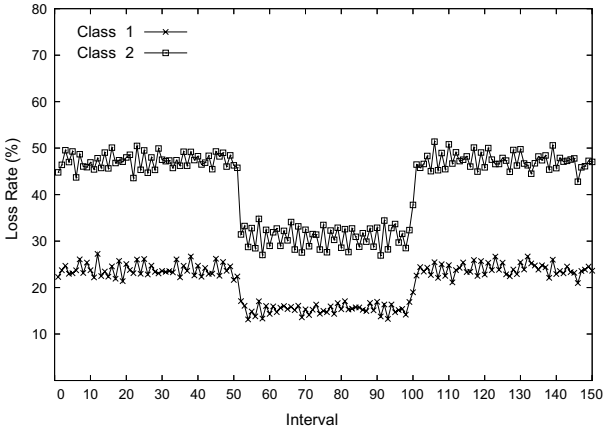


Fig. 13. Two-class short-term behaviors.

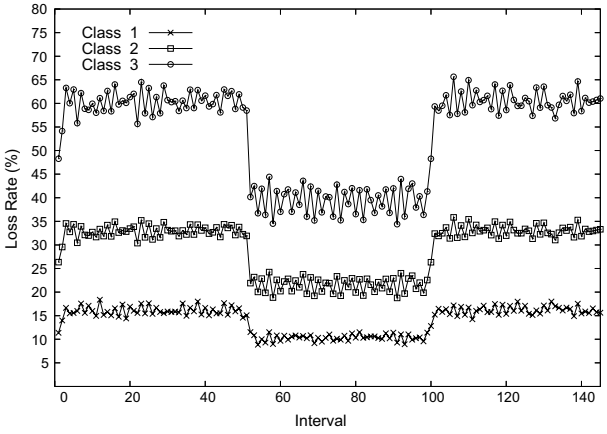


Fig. 14. Three-class short-term behaviors.

the existing packet dropping schemes for loss rate differentiation considered an important issue, that is, the retransmission overhead of dropped packets. In this paper, we have designed a novel hop-count based probabilistic packet dropper (HPPD). HPPD provides two-dimensional loss rate differentiation. Within a traffic class, a less mature packet, which has lower hop count, has higher probability to be dropped than a more mature packet so as to reduce the retransmission cost for dropped packets. We have proposed a unique intra-class nth-root proportional dropping scheme. Simulation results found that the scheme is able to reduce retransmission cost of dropped packets significantly on a single hop while packets with different hop counts got forwarding chances. This saving is even more significant considering that saving will be accumulated in a network route with many hops. Simulation results also found that HPPD inter-class dropping scheme is able to achieve predictable and controllable DiffServ provisioning with respect to loss rate. The major contributions of this work are using hop count information in making differentiated packet dropping decisions so as to achieve congestion mitigation and loss rate differentiation at the same time, and giving a controllable parameter for trading off dropping fairness for congestion mitigation.

Our future work is to implement the HPPD two-dimensional packet dropper in the Click software router and to study the implementation overhead of the dropper and the efficiency of the HPPD-enabled router.

Acknowledgement

This research was supported in part by US National Science Foundation Grant CNS-0720524. A preliminary version of this paper appeared in the Proceedings of the 2006 IEEE International Conference on Communications (ICC), General Symposium [22].

References

[1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, An architecture for differentiated services, IETF RFC 2475 (1998).
[2] S. Chen, N. Yang, Congestion avoidance based on lightweight buffer management in sensor networks, IEEE Trans. Parallel Distrib. Syst. 17 (9) (2006) 934–946.
[3] C. Dovrolis, P. Ramanathann, Proportional differentiated services, part II: Loss rate differentiation and packet dropping, in: Proc. of the Int. Workshop on Quality of Service (IWQoS), 2000.
[4] C. Dovrolis, D. Stiliadis, P. Ramanathan, Proportional differentiated services: delay differentiation and packet scheduling, IEEE/ACM Trans. Networking 10 (1) (2002) 12–26.
[5] A. Eryilmaz, R. Srikant, Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control, in: Proc. of the IEEE INFOCOM, 2005, pp. 1794–1803.
[6] S. Floyd, V. Jacobson, Random early detection gateways for congestion avoidance, IEEE/ACM Trans. Networking 1 (4) (1993) 397–413.
[7] E. Hashem, Analysis of random drop for gateway congestion control, Technical report LCS-TR-465, Lab for Computer Science, MIT, 1989.
[8] Y. Huang, R. Gu. A simple fifo-based scheme for differentiated loss guarantees, in: Proc. of Int. Workshop on Quality of Service (IWQoS), 2004.
[9] B. Hull, K. Jamieson, H. Balajrishnan, Mitigating congestion in wireless sensor networks, in: Proc. ACM SenSys, 2004
[10] V. Jacobson, Congestion avoidance and control, Comput. Commun. Rev. 18 (4) (1988) 314–329.
[11] C. Jin, H. Wang, K. G. Shin, Hop-count filtering: an effective defense against spoofed DDoS traffic, in: Proc. of ACM Conf. on Computer and Communications Security (CCS), 2003.
[12] E. Kohler, R. Morris, B. Chen, J. Jannotti, M.F. Kaashoek, The click modular router, ACM Trans. Comput. Syst. 18 (3) (2000) 263–297.
[13] L. Le, K. Jeffay, F.D. Smith, A loss and queueing-delay controller for router buffer management, in: Proc. IEEE Int'l Conf. on Distributed Computing Systems (ICDCS), 2006.
[14] M.K.H. Leung, J.C.S. Lui, D.K.Y. Yau, Adaptive proportional delay differentiated services: characterization and performance evaluation, IEEE/ACM Trans. Networking 9 (6) (2001) 817–908.
[15] J. Liebeherr, N. Christin, JoBS: joint buffer management and scheduling for differentiated services, in: Proc. Int. Workshop on Quality of Service (IWQoS), 2001, pp. 404–418.
[16] T. Nandagopal, N. Venkitaraman, R. Sivakumar, V. Bharghavan, Delay differentiation and adaptation in core stateless networks, in: Proc. of IEEE INFOCOM, 2000, pp. 421–430.
[17] M.M. Rashid, A.S. Alfa, E. Hossain, M. Maheswaran, An analytical approach to providing controllable differentiated quality of service in Web servers, IEEE Trans. Parallel Distrib. Syst. 16 (11) (2005) 1022–1033.

[18] The Swiss Education Research Network. Default TTL values in TCP/IP. http://secfr.nerim.net/docs/fingerprint/en/ttl_default.html(Date of access: March 3, 2005).
[19] C. Wang, J. Liu, B. Li, K. Sohraby, Y.T. Hou, LRED: a robust and responsive AQM algorithm using packet loss ratio measurement, IEEE Trans. Parallel Distrib. Syst. 18 (1) (2007) 29–43.
[20] J. Wei, C.-Z. Xu, X. Zhou, A robust packet scheduling algorithm for proportional delay differentiation services, Proc. IEEE Globecom 2 (2004) 697–701.
[21] B. Zhang, M. Krunz, Algorithms and protocols for stateless constrained-based routing, Comput. Commun. J., Elsevier 26 (14) (2002) 1570–1580.
[22] X. Zhou, D. Ippoliti, T. Boult. HPPD: a hop-count based probabilistic packet dropper, in: Proc. IEEE Int. Conf. on Communications (ICC), 2006.
[23] X. Zhou, J. Wei, C.-Z. Xu, Quality-of-service differentiation on the internet: a taxonomy, J. Network Comput. Appl., Elsevier 30 (1) (2007) 354–383.
[24] X. Zhou, C.-Z. Xu, Harmonic proportional bandwidth allocation and scheduling for service differentiation on streaming servers, IEEE Trans. Parallel Distrib. Syst. 15 (9) (2004) 835–848.

Xiaobo Zhou received the BS, MS, and PhD degrees in computer science from Nanjing University, in 1994, 1997, and 2000, respectively. He is an assistant professor in the Department of Computer Science at the University of Colorado at Colorado Springs. His research interests are in scalable Internet services and architectures, end-to-end quality of service, and computer networks. He was a founding program cochair of the IEEE International Workshop on Security in Systems and Networks (SSN), the workshops chair of the 16th IEEE Conference in Computer Communications and Networks (ICCCN-2007), a program vice chair of the International Conference on Autonomic and Trusted Computing (ATC-2007), and a program committee member of numerous IEEE conferences including ICDCS'2008, ICC'2007 and AINA'2007. He was a guest editor of Elsevier Journal of Parallel and Distributed Computing, Computer Communications, and Journal of Network and Computer Applications. Dr. Zhou's research was supported in part by the US National Science Foundation, Army Medical Research, and Air Force Research Laboratory. He was a visiting scientist in 1999 and a Post doctorate research associate in 2000 at the Paderborn Center for Parallel Computing, University of Paderborn, Germany. From January 2001 to August 2003, he was a visiting assistant professor in the Department of Computer Science at Wayne State University, Detroit. He was the recipient of the Outstanding Researcher of the Year of the EAS College and the research award of the University of Colorado at Colorado Springs in 2005. He is a member of the IEEE Computer Society.

Dennis Ippoliti received the MA degree from Bellevue University, 2001, and the MS degree in computer science from the University of Colorado at Colorado Springs, 2006. He is a PhD student in the Department of Computer Science at the University of Colorado at Colorado Springs. His research interests are mainly in quality-of-service and network communications.

Terrance Boult received the BS in Applied Mathematics in 1983, and the MS and PhD degrees in computer science in 1984 and 1986 respectively from Columbia University. He is the El Pomar Professor of Innovation and Security at University of Colorado at Colorado Springs and director of the Vision and Security Technology lab. He is also a Co-founder and CEO/CTO of Securics. He was on the faculty of Columbia University from 1986 until joining Lehigh University in 1994. At Lehigh he became the Weismann Chair Professor and Founding Chair of the Computer Science and Engineering Department. He received an NSF Presidential Young Investigator award, and has won teaching and research awards at both Columbia University and Lehigh University. He has published over 150 papers and holds 6 patents, with 8 pending. He was the University Colorado Inventor of the year for UCCS and a finalist in the 2007 EE Times ACE awards for Innovator of the year. He has been the Chair of the IEEE Technical Pattern Analysis and Machine Intelligence (PAMI) since 2005, and a member of the IEEE Golden Core Society since 2006. He has served as an associate editor for IEEE Trans. on Pattern Analysis and Machine Perception (PAMI) and played major organizational roles in dozens of conferences/workshops in IEEE, ACM and SPIE. He has directed 23 students to their PhD in the areas of computer vision, networking, distributed systems, biometrics and security.