Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2012, Article ID 827137, 16 pages
doi:10.1155/2012/827137

Research Article

A Fuzzy Dropper for Proportional Loss Rate Differentiation under Wireless Network with a Multi-State Channel

Yu-Chin Szu

Department of Information Management, St. John's University, 499, Section 4, Tam King Road, Tamsui District, New Taipei City 25135, Taiwan

Correspondence should be addressed to Yu-Chin Szu, belinda@mail.sju.edu.tw

Received 18 October 2011; Revised 25 January 2012; Accepted 3 February 2012

Academic Editor: Jung-Fa Tsai

The proportional loss rate differentiation (PLD) model was proposed to provide controllable and predictable loss rate for different classes of wired network connections. However, these algorithms cannot be directly applied to wireless networks, because of the location-dependent and time-varying wireless channel capacity. This paper proposes a novel packet dropper for fuzzy controlling of the proportional loss rate differentiation in a wireless network with multistate channel. The proposed dropper, fuzzy proportional loss rate dropper (FPLR), prefers to drop the small packets destined to a poor condition channel to improve the network performance. The loss rate debts of the poor channel will be compensated later to keep PLD. From simulation results, FPLR does achieve accurate loss rate proportion, lower queuing delay and loss rate, and higher throughput, compared with other methods in the wireless environment.

1. Introduction

The Internet becomes an important infrastructure of global communication. However, many multimedia applications require some guarantee of the quality of services (QoS). Thus the best-effort service is not suitable to such applications because it cannot promise any guarantee about packet loss, delay, or jitter. The Internet Engineering Task Force (IETF) first proposed the Integrated Service (IntServ) as a QoS architecture for IP networks [1]. Because IntServ suffers from the scalability problem, the Differentiated Service (DiffServ) [2] was then proposed. Diffserv has proceeded in two directions—absolute differentiated service and relative differentiated service. The absolute differentiated service ensures that an admitted user can enjoy certain and steady performance, while the relative differentiated service ensures that users with a higher service class experience better performance than the

users with a lower one. The proportional differentiation model, a branch of relative service differentiation, has received a lot of attention because it can perform the controllable and predictable differentiation [3, 4]. That is, the proportional differentiation model offers the network manager a means of varying quality spacing between service classes according to the given pricing or policy criteria and ensures that the differentiation between classes is consistent in any measured timescale.

The proportional services can be differentiated according to different performance metrics, such as throughput, delay, loss rate, or jitter. When adopting packet loss rate as the performance metric, the proportional differentiation model is referred to as the proportional loss rate differentiation (PLD) model. To provide PLD, some methods, such as Proportional Loss Rate (PLR) [5], Average Drop Distance (ADD) [6], Debt Aware [7], Weighted Random Early Detection (WRED) [8], Weight Simple Adaptive Proportion (WSAP) [9], Hop-count Probabilistic Packet Dropper (HPPD) [10], and RED with maximum drop probability adjustment, were proposed.

As wireless technologies rapidly advance and lightweight portable computing devices become popular, wireless networks have become pervasive. Accordingly, the PLD model is also urgently required for wireless environments, just as it was for wired networks. However, the above approaches designed in a wired network are not applicable in a wireless environment, which has some distinct characteristics, such as high error rate, location-dependent and time-varying capacity, and scarce bandwidth [11].

In a wireless network with a multirate channel, when many packets encountering a good channel are dropped and many packets encountering a poor channel are kept in the buffer, the system performance becomes poor because most packets are transmitted in a poor-capacity channel. Actually, dropping the packet having the poorest channel will cause the optimal performance, but this behavior completely ignores to maintain the PLD model. Therefore, how to keep the PLD model and raise the overall performance in a wireless network with a multirate channel is a challenge and will be investigated in this paper.

This paper proposes a novel algorithm, named fuzzy proportional loss rate dropper (FPLR). In the FPLR dropper, we utilize the concept of fuzzy control to make an optimal decision for dropping the small packets destined to a poor channel and reserving the large packets destined to a good channel, causing a lot of large packets to be transmitted via a good channel. Therefore, FPLR can provide high performance and keep PLD in a wireless with a multistate channel.

Organization of the remainder of the paper is as follows. Section 2 describes the background, including the proportional differentiation model, some previous related works, structure of fuzzy controller systems, and what problems occur when these works are directly applied into a wireless network. Section 3 describes in more detail the FPLR and the design of the fuzzy controller. In Section 4, some simulation results and their implications are exhibited. The conclusion of this work is given in Section 5.

2. Background

2.1. Proportional Differentiation Model

The proportional differentiation model was proposed first by Dovrolis and Ramanathan [5] and Bobin et al. [6], and its structure is shown as Figure 1. The arrival traffic is classified into N service classes where each class has a dedicated queue. Let q_i denote the measured

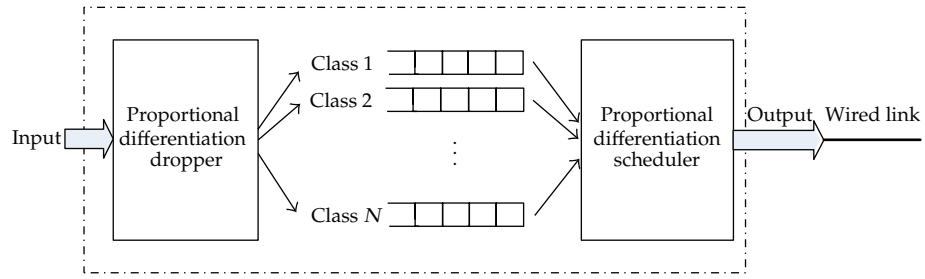


Figure 1: The proportional differentiation model.

performance of class i. For proportional differentiation model, the following equation should be satisfied for all pairs of classes, i and j,

$$\frac{q_i}{q_j} = \frac{c_i}{c_j} \quad (1 \le i,\ j \le N), \tag{2.1}$$

where $c_1, c_2, \ldots, c_N$ are the quality differentiation parameters (QDP). A network operator could manipulate the service quality spacing between classes by adjusting QDPs.

Let $\overline{L}_i$ be the average loss of the class-i packets with σ_i as the loss differentiation parameters (LDPs). For all pairs of service classes, i and j, the proportional loss differentiation model is specified by

$$\frac{\overline{L}_i}{\overline{L}_j} = \frac{\sigma_i}{\sigma_j} \quad (1 \le i,\ j \le N). \tag{2.2}$$

Let $\overline{D}_i$ be the average queuing delay of the class-i packets with δ_i as the delay differentiation parameters (DDPs). The proportional delay differentiation model has the following constraint for any pair of classes:

$$\frac{\overline{D}_i}{\overline{D}_j} = \frac{\delta_i}{\delta_j} \quad (1 \le i,\ j \le N). \tag{2.3}$$

2.2. Proportional Loss Rate (PLR)

Dovrolis et al. proposed the proportional loss rate dropper (PLR) to offer the PLD model [5]. In order to determine which packet should be dropped, PLR uses a loss history table (LHT) to record the loss rate of each class at present. Let $\overline{L}_i(t)$ be the average loss rate and $\widetilde{L}_i(t)$ be the normalized average loss rate of class i at time t. Also $L_i(t)$ and $A_i(t)$ denote the numbers of dropped packets and arrived packets, respectively, of class i until time t. $B(t)$ denotes the set of backlogged classes. PLR chooses class J to drop its tail packet as follows:

$$\overline{L}_i(t) = \frac{L_i(t)}{A_i(t)} \quad (1 \le i \le N), \tag{2.4}$$

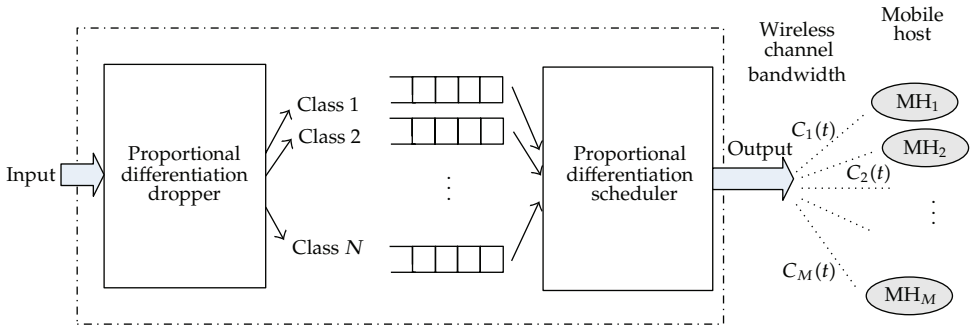


Figure 2: The proportional differentiation model in a wireless network.

$$\widetilde{L}_i(t) = \frac{\overline{L}_i(t)}{\sigma_i} \quad (1 \le i \le N), \tag{2.5}$$

$$J = \arg \min_{i \in B(t)} \widetilde{L}_i(t) \quad (1 \le i \le N). \tag{2.6}$$

PLR aims to maintain a unanimous normalized average loss rate among all classes. According to the number of packets that PLR estimates, the calculated average loss rate is different; so there are two kinds of algorithms, namely, PLR with infinite memory, PLR (∞), and PLR with memory M, PLR(M). When calculating the average loss rate, PLR(∞) counts packets from initial to present, while PLR(M) observes the last M packets of every class at present. From the long-term observation, the result of PLR(∞) is closer to targeted proportion than that of PLR(M). However, when the class load significantly oscillates, adopting PLR(M) is preferred because of its adaptation, but determining an optimal M is difficult.

2.3. *The Problem of Achieving PLD in a Wireless Network*

Previous studies on achieving the PLD model focused on increasing the accuracy of the achieved loss proportion between classes in wired networks. These proposed algorithms did not consider any characteristics in a wireless network.

Figure 2 depicts the PLD model applied in a wireless network. In this environment, all mobile hosts share one wireless link. Since each host could be located at different places, different capacities exist when the scheduler transmits data to different mobile hosts via this wireless link. Also as the mobile host moves, the capacity to this destined host varies. Thus a wireless link has a location-dependent and time-varying capacity, and it is called as a multistate link herein. For simplicity, the term channel j means the wireless link at transmitting the packet to the mobile host j. Let $C_j(t)$ denote the encountered capacity when the scheduler transmits packets via channel j at time t.

In a wireless network with a multirate link, when many packets encountering a good channel are dropped and many packets encountering a poor channel are kept in the buffer, the system performance becomes bad because most packets will be transmitted via a poor-capacity channel. Actually, dropping the packet having the poorest channel will cause the optimal performance, but this behavior completely ignores keeping the PLD model.

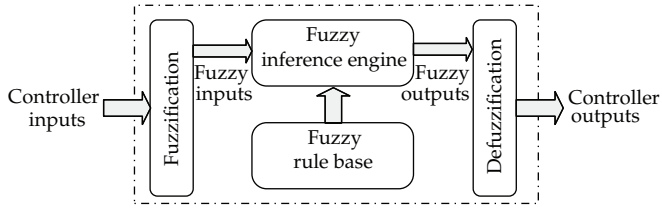


Figure 3: The architecture of the fuzzy controller.

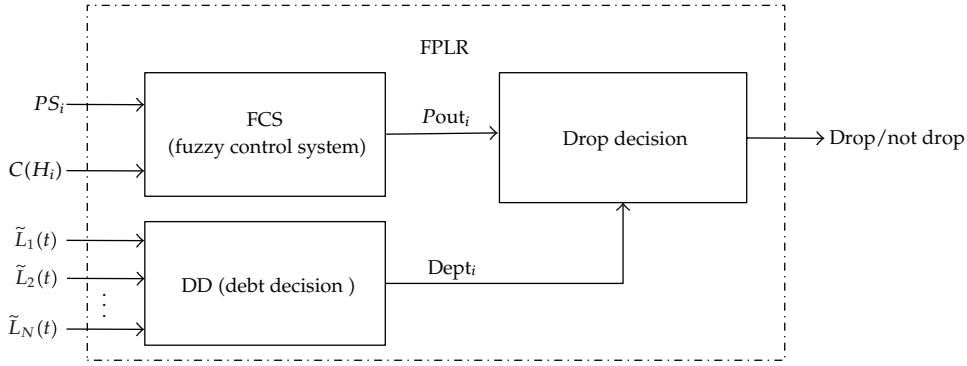


Figure 4: The structure of FPLR.

Therefore, designing a dropper to simultaneously keep the PLD model and raise the overall performance in a wireless network with a multirate link is a challenge.

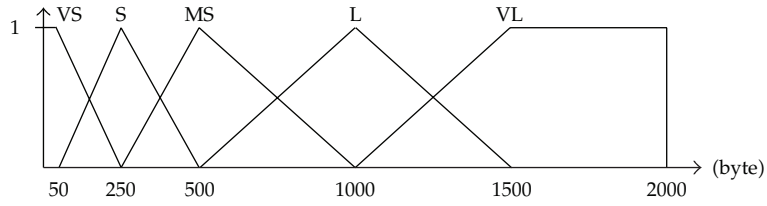
2.4. The Fuzzy Controller

The fuzzy controller that is a nonlinear mapping system developed by Mamdani [12, 13] consists of four main components: fuzzification, fuzzy inference engine, rule base, and defuzzification, as shown in Figure 3.

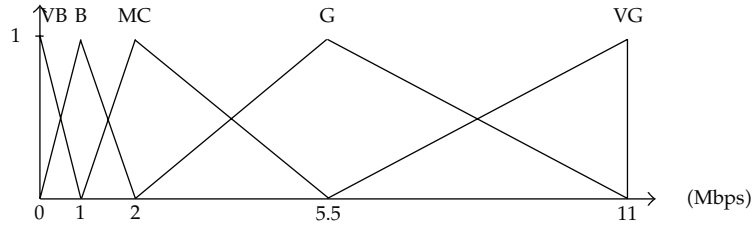
The fuzzification (or fuzzifier) calculates suitable sets of degree of membership, called "fuzzy sets," for crisp inputs. The fuzzy inference engine performs inference procedure for given inputs according to the fuzzy rules to derive proper actions. The rule base consists of a set of linguistic rules, as well as a set of membership functions for linguistic values. The defuzzification (or defuzzifier) converts the fuzzy output to crisp values.

3. Fuzzy Proportional Loss Rate Dropper (FPLR)

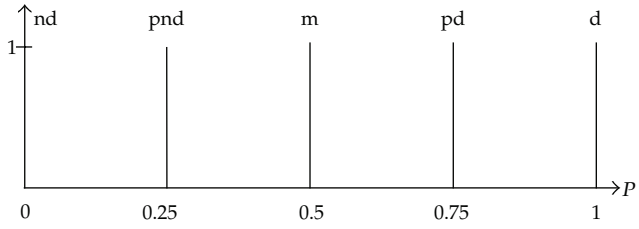
FPLR uses three processes to decide how to drop packets, as shown in Figure 4. A fuzzy controller system on the first process takes advantage of fuzzy theory to PLD. The debt decision module in the second process is to compensate for the loss rate debts. The third process is to determine an appropriate packet to drop.



(a) The membership functions for the input PS



(b) The membership functions for the input $C(H)$



(c) The membership functions for the output of fuzzy inference engine

Figure 5: The membership functions.

3.1. Fuzzification

In this paper, most of the membership functions for the fuzzy sets are chosen to be triangular, because triangular membership functions are the most economic. All membership functions are defined on common interval $[0, 1]$. With fuzzy logic, we assign grade values to our two variables, which are packet sizes and channel states. Thus the fuzzy set A consists of two fuzzy variables, PS, $C(H)$, where PS is the fuzzy variable term for the packet size and $C(H)$ is the channel capacity encountered when the scheduler transmits the HOL packet. The membership functions of PS and $C(H)$ are shown in Figures 5(a) and 5(b), respectively, and they are the input of the FCS. The fuzzification component translates the inputs into corresponding fuzzy values.

In Figure 5(a), VS, S, MS, L, and VL denote very small, small, medium small, large, and very large, respectively. In Figure 5(b), VB, B, MC, G, and VG indicate very bad channel, bad channel, medium channel, good channel, and very good channel, respectively.

Let the fuzzified output of the fuzzy inference engine be P, which uses singleton fuzzy sets representing the drop probability of the packet, as shown in Figure 5(c). In Figure 5(c), nd, pnd, m, pd, and d denote not drop, probably not drop, maybe drop, probably drop, and drop, respectively.

Table 1: The rule base.

		Packet sizes				
		VS	S	MS	L	VL
Channel capacity	VB	d	d	d	d	d
	B	pd	pd	pd	m	m
	MC	pd	pd	m	pnd	pnd
	G	m	m	pnd	pnd	pnd
	VG	nd	nd	nd	nd	nd

3.2. Rule Establishment

Fuzzy rules can effectively treat the nonlinearity, and the whole fuzzy rules are shown in Table 1. Each rule performs the mapping process from the fuzzy input to fuzzy out. These rules are intuitive. For example, when channel state is very bad and no matter a packet is big or small, an incoming packet must be dropped.

3.3. Defuzzification

The center of gravity (CoG) technique, which computes the weighted average of the center of gravity of each membership function, is used to compute the defuzzified output, P_{out}, of the FCS. That is $P_{\text{out}i} = \sum_{f=1}^{n} \mu_{i,f}(P_{i,f}) P_{i,f} / \sum_{f=1}^{n} \mu_{i,f}(P_{i,f})$, where $P_{i,f}$ is the center of the membership function recommended by the consequence of rule f of class i, and the height is cut by the minimum vale after MAX-MIN inference engine [14]. $u_{i,f}(P_{i,f})$ is the degree of membership of input variables, packet size and channel states, of class i. The n is number of rules.

3.4. Debt Decision

The debt decision is adopted because, in this work, maintaining the loss rate differentiation is considered to be more important than achieving high throughput. A debt value debt is used to record the "drop debt" carried by class i. debt_i having the positive value represents that the number of loss happening is less than the expectation in class i. That is, some other classes instead of class i drop the packets. Similarly, the value of debt_i being negative and zero means that class i has more and equal number of losses than its expectation, respectively. debt is specified by

$$\begin{aligned} &\text{If } \widetilde{L}_J(t) > \left(\frac{\sum_{i=1}^{N} \widetilde{L}_i(t)}{N} \right) \text{ then } \text{debt}_J = -0.25, \\ &\text{If } \widetilde{L}_J(t) < \left(\frac{\sum_{i=1}^{N} \widetilde{L}_i(t)}{N} \right) \text{ then } \text{debt}_J = 0.25, \end{aligned} \tag{3.1}$$

where $\widetilde{L}_J(t)$ is the normalized average loss rate of class J at time t and N is the number of service classes.

We would like to use the decision function of FPLR to decide that the packet should be dropped and need not be dropped. We use a threshold θ to compare with P_{out}. Thus the condition of dropping the packet of the candidate class J is as follows:

$$\left(P_{\text{out}_J} + \text{debt}_J\right) > \theta. \tag{3.2}$$

3.5. Algorithm

Algorithm 1 presents the pseudocode of the FPLR. When the buffer has empty space, the dropper simply inserts the packet into a proper queue. When buffer overflow occurs, the dropper selects an appropriate packet to drop. Let $L_i(t)$ and $A_i(t)$ be the number of dropped packets and the number of arrived packets of class i at time t, respectively.

When an arriving packet encounters a full buffer, the dropper selects a proper packet, which may be this arriving packet or a packet in the buffer to be dropped for keeping the proportional loss rate among classes. At determining which packet to be dropped, the dropper first calculates the normalized average loss rate $\widetilde{L}_i(t) = L_i(t)/A_i(t)\sigma_i$ for each class i. Then the class with the smallest normalized average loss rate, that is, $J = \arg\min_{i\in B(t)} \widetilde{L}_i(t)$, is regarded as the candidate class.

The dropper considers three important factors, including channel state, packet size, and debt degree, to determine whether the packet of the class J should be dropped.

Note the dropper drops the HOL packet, rather than the tail packet or arriving packet, because using this method can reduce the queuing delay of queued packets.

If the candidate class J does not satisfy the above condition, the dropper will choose the candidate class K, which has the next smallest normalized average loss rate. Judging whether the HOL packet of candidate class K will be dropped is similar to (3.2), that is, $(P_{\text{out}_K} + \text{debt}_K) > \theta$.

Some noticeable points about the FPLR are explained as follows.

(1) FPLR drops the HOL packet, rather than the tail packet, of the selected class, because using this method can reduce the queuing delay of queued packets. Also the HOL packet will actually experience the current channel capacity, but the tail packet may not be transmitted at this capacity because of the time-varying bandwidth.

(2) For FPLR, because the packets destined to a low-capacity channel are easily dropped and the packets destined to a high-capacity channel are easily kept in the buffer, most packets are transmitted in a good channel, generating high performance.

(3) The fuzzy logic is an effective tool for efficient buffer management. It can easily handle several nonlinear factors and does not need detailed mathematic descriptions for the system.

4. Simulation and Discussion

The simulations compare the performance of FPLR and WPLR in terms of *loss rate ratio*, *loss improvement*, *throughput improvement*, and *delay improvement*. The simulations are conducted to investigate the effects of packet arrival rate, number of mobile hosts, and state transition rate

L_i: the number of lost packets of class i
A_i: the number of arrived packets of class i
σ_i: the loss differentiation parameter of class i
$C(H_i)$: the channel capacity encountered when the scheduler transmits the HOL packet of class i
$B(t)$: the set of backlogged classes at present t
PS_i: the size of the HOL packet in class i
$P_{i,f}$: the center of the membership function recommended by the consequence of rule f of class i
$\mu_{i,f}(P_{i,f})$: the degree of membership of input variables, packet size and channel states, of class i
$P_{\text{out}i}$: the defuzzified output of FCS of class i
θ: drop threshold
debt_i: lost debt of class i

FPLR Dropper

```
{
  a class-i packet arrives at time t, A_i(t) + +;
  if (buffer overflow){
    calculate L̃_i(t) = L_i(t)/A_i(t)σ_i, i = 1,2,...,N;
    calculate P_out_i = Σ_{f=1}^{n} μ_{i,f}(P_{i,f})P_{i,f} / Σ_{f=1}^{n} μ_{i,f}(P_{i,f}), i = 1,2,...,N; f = 1,2,3,4,5;
    J=arg min_{i∈B(t)} L̃_i(t);
    debt_i = 0;
    if (L̃_J(t) > (Σ_{i=1}^{N} L̃_i(t)/N)) then debt_J = −0.25, i = 1,2,...,N;
    else if (L̃_J(t) < (Σ_{i=1}^{N} L̃_i(t)/N)) then debt_J = 0.25, i = 1,2,...,N;
    if ((P_out J +debt_J)> θ) {
      drop the HOL packet from class J;
      L_J(t) + +;}
    else{
      do {
        findK = arg next min_{i∈B(t)} L̃_i;
        if ((P_out K +debt_K)> θ) {
          drop the HOL packet from class K;
          L_K(t) + +;}
      } while(one packet is dropped or all classes have been visited)
      if (no packet is dropped){
        drop the HOL packet from class J;
        L_J(t) + +;}
    }
   accept the incoming packet;
}
```

Algorithm 1: The FPLR algorithm.

upon WPLR and FPLR. Also, the behaviors of these two droppers under different timescales are explored. First, the average loss rate of each class i, $\overline{L}_i$, is obtained by using (2.4), and the loss rate ratio of class i over class j is defined as $R_{ij}=\overline{L}_i/\overline{L}_j$. The loss improvement of class i is defined as $(\overline{L}_i^P - \overline{L}_i^F)/\overline{L}_i^P$ where $\overline{L}_i^P$ and $\overline{L}_i^F$ are the loss rates of class i made by WPLR and FPLR, respectively. Similarly, the throughput improvement and delay improvement of class i are defined as $(T_i^F - T_i^P)/T_i^P$ and $(\overline{D}_i^P - \overline{D}_i^F)/\overline{D}_i^P$, respectively, where T_i denotes the throughput of class i and $\overline{D}_i$ denotes the average queuing delay of class i.

4.1. Simulation Models

The model we simulated is depicted in Figure 2, and the scheduler is using the First-Come First-Served (FCFS) scheduler. In all simulations, three service classes ($N = 3$) are assumed, and the corresponding loss rate differentiation parameters are set as $\sigma_1 = 1$, $\sigma_2 = 2$, and $\sigma_3 = 4$. Packet arrival follows a Poisson process, and its mean overall arrival rate is $\lambda = 120$ packets/sec. The distribution of packet sizes for all classes is such that 40% of packets are 40 bytes, 50% packets are 550 bytes, and 10% packets are 1500 bytes. Thus, the mean packet size is 441 bytes. The threshold θ is set as 0.5. The total buffer size is 10,000 bytes, and the number of hosts is five, that is, $M = 5$. The wireless channel is simulated by a multistate Markov process, which has five states with the values of capacities varying among 0 (purely bad), 1 Mbps, 2 Mbps, 5.5 Mbps, and 11 Mbps (purely good). The transition rate matrix of channel capacity is set as

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} -a & ap_1 & ap_1^2 & ap_1^3 & ap_1^4 \\ ap_2 & -a & ap_2 & ap_2^2 & ap_2^3 \\ ap_3^2 & ap_3 & -a & ap_3 & ap_3^2 \\ ap_4^3 & ap_4^2 & ap_4 & -a & ap_4 \\ ap_5^4 & ap_5^3 & ap_5^2 & ap_5 & -a \end{bmatrix} \end{array}, \tag{4.1}$$

where a is the state transition rate to other states and p_i is the probability of state i being translated to its neighbor states when the transition occurs. The default value of a is 30, and the values of p_1, p_2, p_3, p_4, and p_5 can be calculated by letting the sum of each row equal to 0. In each simulation, at least 500,000 packets for each class are generated for the sake of stability.

4.2. Packet Arrival Rate

To observe the influence of different packet arrival rates on achieved loss rate ratios, loss improvement, delay improvements and throughput improvement, the arrival rate (λ) is varied from 100 to 150 packets/sec. The target loss rate ratios are 4 ($\sigma_3/\sigma_1 = 4/1$) for class 3 over class 1 and 2 ($\sigma_2/\sigma_1 = 2/1$) for class 2 over class 1. Figure 6(a) shows that WPLR and FPLR can achieve the accurate loss rate ratios. For the FPLR dropper, because the class with a larger debt has a higher probability to be dropped, the target loss rate proportions can be still maintained.

Observed from Figure 6(b), the loss improvements attained by FPLR increase as arrival rate increases, and three classes have the same loss improvement. At a low packet arrival rate, the packet losses seldom happen, causing that the fuzzy mechanism has minor influence. As the arrival rate increases, since the opportunity of using the fuzzy mechanism becomes large, more small packets transmitted via low-capacity channels are dropped, and more large packets transmitted via high-capacity channels are kept. Thus the loss improvements made by FPLR increase.

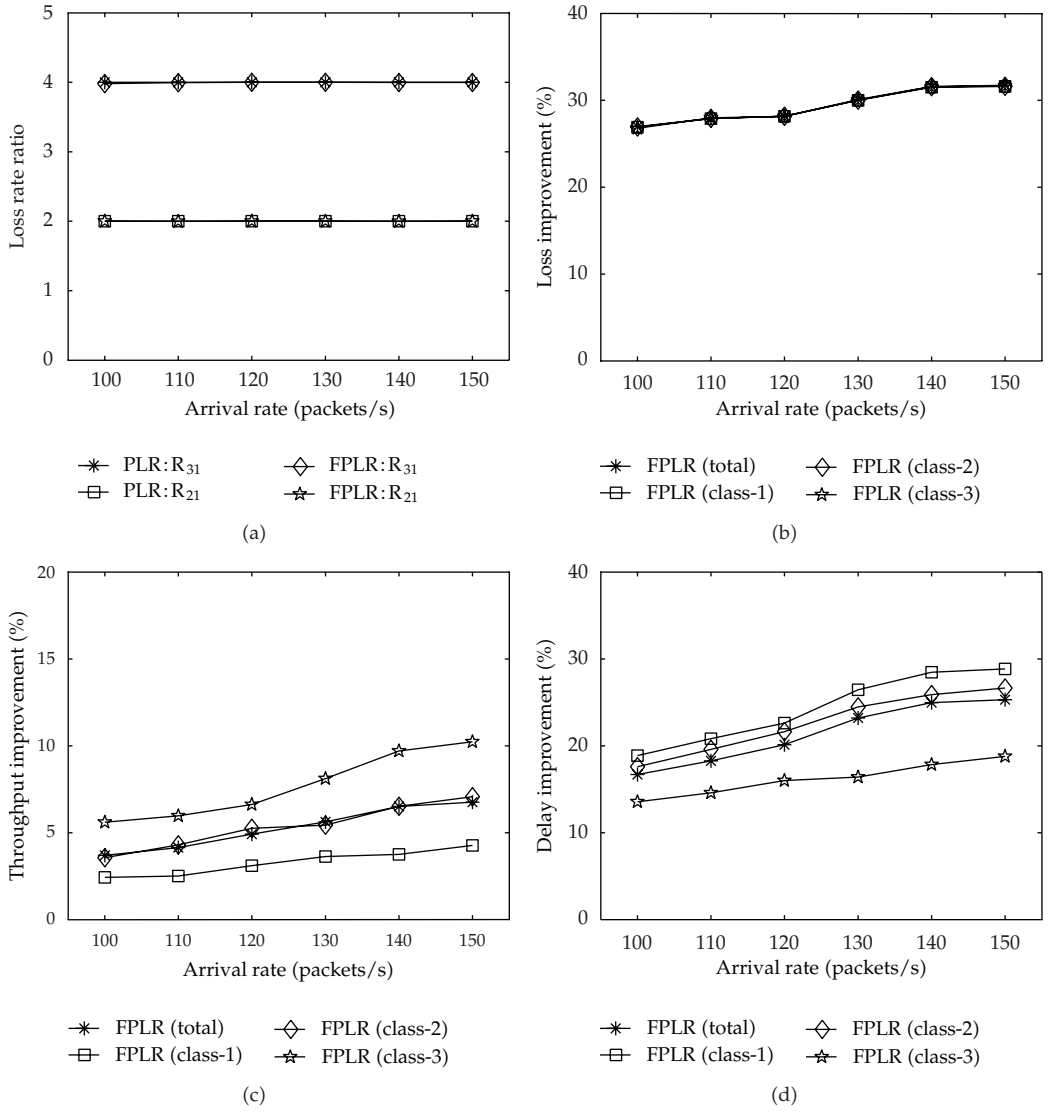


Figure 6: The effect of packet arrival rate.

From Figure 6(c), two phenomena are observed. First, the throughputs improved by FPLR increase as arrival rate increases. Second, the throughput improvements of all classes are in the order class 3 > class 2 > class 1. The first phenomena is because the more loss improvement, the more throughput improvement. The second phenomenon is not trivial and explained in the following.

Let λ_i be the packet arrival rate of class i. Thus for WPLR, the throughput of class i, T_i^P, can be expressed as $T_i^P = \lambda_i(1 - \overline{L}_i^P)$. Also let the loss improvement of class i, made by the dropper FPLR, be denoted as f_i^F, which is equal to $(\overline{L}_i^P - \overline{L}_i^F)/\overline{L}_i^P$. Therefore, the throughput

of class i can be easily obtained as $T_i^F = \lambda_i(1 - \overline{L}_i^F) = \lambda_i(1 - (1 - f_i^F)\overline{L}_i^P)$. Hence, the throughput improvement of class i is

$$\frac{T_i^F}{T_i^P} - 1 = \frac{1 - (1 - f_i^F)\overline{L}_i^P}{1 - \overline{L}_i^P} - 1. \tag{4.2}$$

From this equation, we can prove the more loss improvement, the more throughput improvement.

For different classes, observed from Figure 6(b), the loss improvements, f_i^F, of all classes are the same. Also Figure 6(a) shows that $\overline{L}_3^P > \overline{L}_2^P > \overline{L}_1^P$. Under these conditions, from (4.2), the throughput improvements of all classes are in the order class 3 > class 2 > class 1. For example, all $\lambda_i = 120$ packets/sec, loss rates of classes 1, 2, and 3 for WPLR are 20%, 40%, and 80%, respectively, that is, their throughputs are 100, 80, and 40. When the loss improvement is 50% for all classes, the loss rates of classes 1, 2, and 3 for our proposed dropper reduce to 10%, 20%, and 40%, respectively, that is, their throughputs are 110, 100, and 80. In this example, the throughput improvements for classes 1, 2, and 3 are 10%, 25%, and 100%, respectively.

Figure 6(d) shows the delay improvements achieved by FPLR. The delay improvements are caused by two reasons. First, because the packet destined to a poor channel is easier to be dropped than that destined to a good channel, nondropped packets usually encounter a good channel, causing that they have short transmission time and thus short queuing delay. Second, FPLR dropping the HOL packet, rather than dropping the tail packet, reduces the queuing delay of the queued packets. Also the delay improvements enhance as the arrival rate increases because the opportunity of using the fuzzy mechanism becomes large. Finally, the delay improvements of all classes are in the order class 1 > class 2 > class 3 since the more throughput improvement, the less delay improvement.

4.3. Timescale

In this simulation, the loss rate ratios between successive classes are measured over five time intervals—100, 500, 1000, 5000, and 10000 p-units, where a p-unit is the average packet transmission time. During each time interval, the loss rate ratios of class 2 over class 1 and class 3 over class 2 are measured and averaged.

Figure 7 shows five percentiles, 5%, 25%, 50% (median), 75%, and 95%, of the average loss rate ratio. FPLR has the broader ranges on short timescales than WPLR and similar ranges on long timescales. The reason is that although using the fuzzy mechanism will achieve the loss rate ratio temporarily away from target loss rate proportion in the short term, they will let the opportunity of using the debt decision mechanism keep PLD on the long term.

4.4. Number of Mobile Hosts

The number of mobile hosts is varied from 5 to 25 to observe the influence on achieved loss rate ratios, loss improvement, delay improvement, and throughput improvement. Packets of each class are evenly distributed to each mobile host. Since below all simulation results for different classes have the similar treads as Figure 6. Therefore we will only show the total

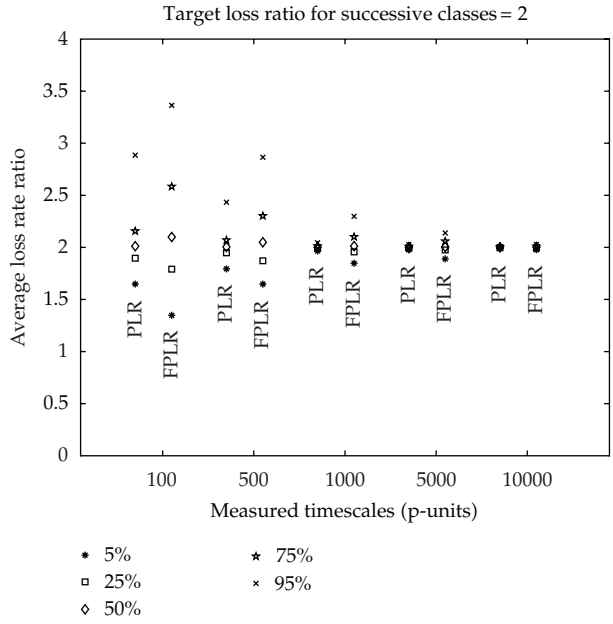


Figure 7: Five percentiles of average loss rate ratio on various measured timescales.

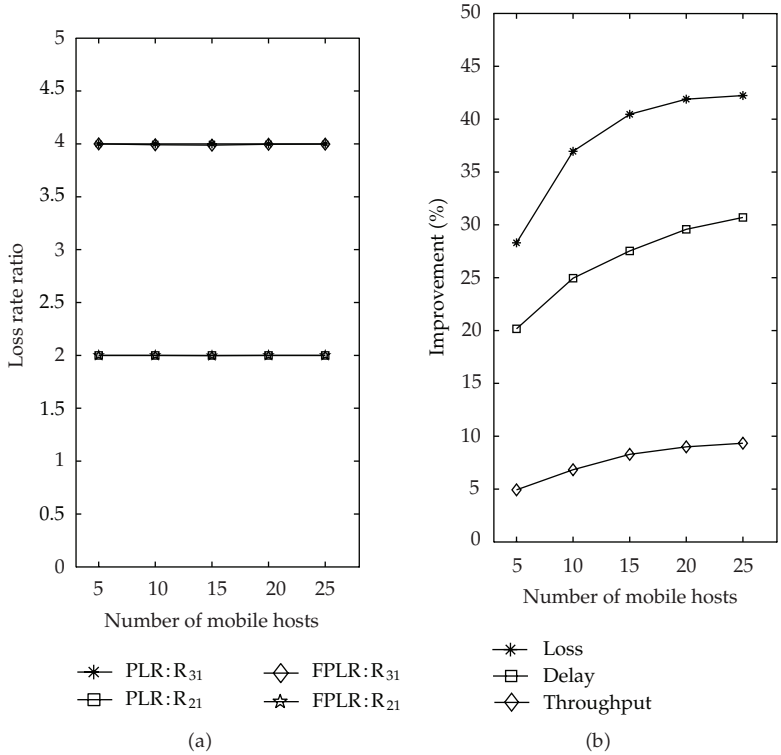


Figure 8: The effect of the number of mobile hosts.

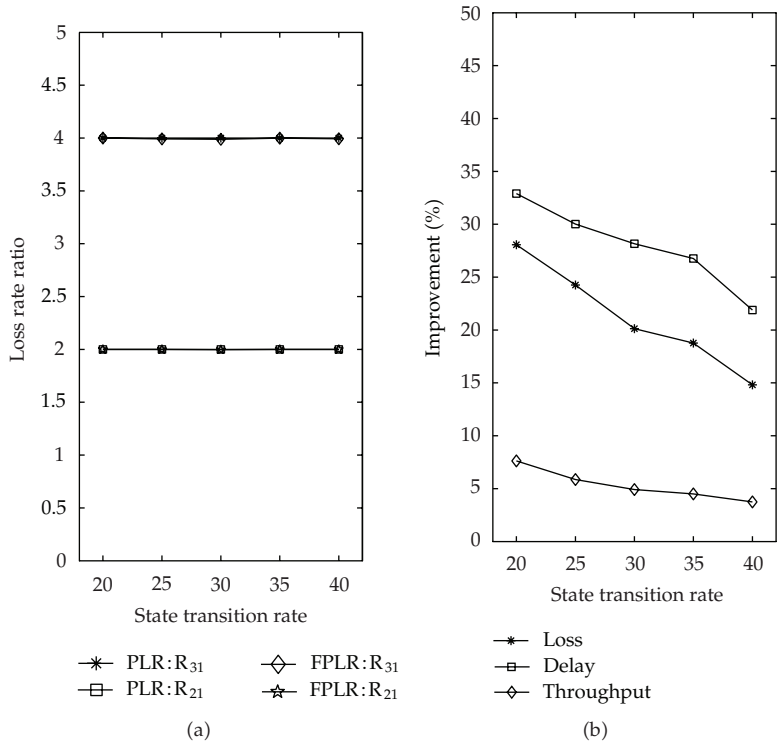


Figure 9: The effect of the state transition rate.

loss improvement, total throughput improvement, and total delay improvement. Figure 8(a) shows that the number of mobile hosts does not affect the loss rate ratios achieved by two droppers. Figure 8(b) reveals that the loss improvement, throughput improvement, and delay improvement made by FPLR increase as the number of mobile hosts increases. When there are only a few mobile hosts, for FPLR, few HOL packets destined to poor channels can be selected as a substitution, so that the improvements are not high. As the number of mobile hosts increases, since more small packets transmitted via poor channels can be chosen, the improvements made by FPLR raise.

4.5. State Transition Rate

Given that a channel varies among the five states, 0, 1 Mbps, 2 Mbps, 5.5 Mbps, and 11 Mbps, the transition rate, a, is varied from 20 to 40 to observe its effect upon loss rate ratios and all improvements. Figure 9(a) shows that the state transition rate does not affect the loss rate ratios. From Figure 9(b), as state transition rate changes faster, all improvements of FPLR become smaller. When the state transition rate is small, the bad channel will lasts long, causing that the HOL blocking also last long. Many packets will be accumulated in the buffer, and packet losses usually occur, leading FPLR to have high opportunities to adopt the fuzzy mechanism. When the state transition rate becomes larger, the packets caused by the HOL blocking can be absorbed in the buffer, implying a smaller loss rate. Thus, the improvements made by FPLR also become smaller.

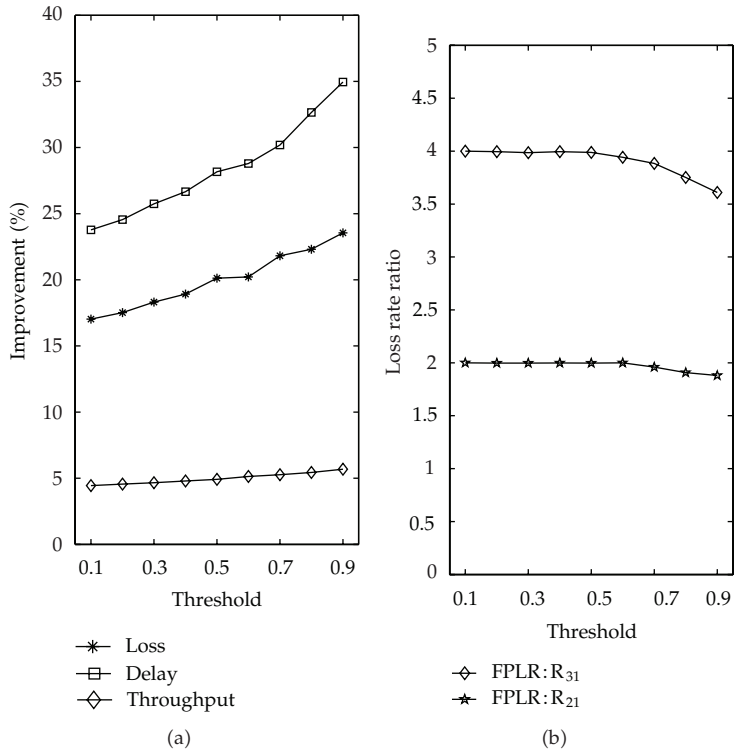


Figure 10: The effect of threshold θ.

4.6. Threshold θ

Herein we investigate the effects of the parameter θ for FPLR. Figure 10(a) reveals that loss improvement, throughput improvement, and delay improvement made by FPLR increase as threshold θ increases. As the threshold θ increases, since the opportunity of considering packet size and channel states becomes large, smaller packets transmitted via low-capacity channels are dropped, and more large packets transmitted via high-capacity channels are kept. Thus the improvements made by FPLR increase. However, Figure 10(b) shows when threshold θ exceeds 0.5, the target loss rate ratios slide down as threshold θ increases. The reason is that in this case the loss rate proportions have minor influence, so the target loss rate proportions cannot be maintained.

5. Conclusions

This paper proposed a high-performance algorithm, FPLR, which can be used to provide proportional loss differentiation in a wireless network with multi-state channel. FPLR uses three processes to decide how to drop packets. A fuzzy controller system on the first process takes advantage of fuzzy theory to PLD. The debt decision module in the second process is to compensate for the loss rate debts. The third process is to determine an appropriate packet to drop. FPLR not only considers the normalized average loss rate, but also considers the channel state and packet sizes, in order to improve the performance of dropping.

From the simulation results, FPLR is examined to deal with location-dependent channel capacity well and does provide more accurate proportional loss rate differentiation than WPLR. Compared with WPLR, FPLR indeed provides better throughput and lower queuing delay and loss in the wireless network with a multistate channel.

Our future works include developing a wireless proportional scheduler to provide both proportional loss differentiation and delay differentiation.

References

[1] R. Braden, D. Clark, and S. Shenker, "Integrated services in the internet architecture: an overview," IETF RFC 1633, June 1994.

[2] S Blake, D. Black, M. Carlson, Z. Wang, and W. Weiss, "An architecture for differentiated services," IETF RFC 2475, December 1998.

[3] C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional differentiated services: delay differentiation and packet scheduling," *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 4, pp. 109–120, 1999.

[4] C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional differentiated services: delay differentiation and packet scheduling," *IEEE/ACM Transactions on Networking*, vol. 10, no. 1, pp. 12–26, 2002.

[5] C. Dovrolis and P. Ramanathan, "Proportional differentiated services, part II: loss rate differentiation and packet dropping," in *Proceedings of the IEEE/IFIP International Workshop on Quality of Service (IWQoS '00)*, pp. 52–61, June 2000.

[6] U. Bobin, A. Jonsson, and O. Schelen, "On creating proportional loss-rate differentiation: predictability and performance," *Lecture Notes in Computer Science*, vol. 2092, pp. 372–379, 2001.

[7] J. Zeng and N. Ansari, "An enhanced dropping scheme for proportional differentiated services," in *Proceedings of the International Conference on Communications (ICC '03)*, pp. 11–15, 2003.

[8] J. Aweya, M. Ouellette, and D. Y. Montuno, "Weighted proportional loss rate differentiation of TCP traffic," *International Journal of Network Management*, vol. 14, no. 4, pp. 257–272, 2004.

[9] M. Zhang, J. Wu, C. Lin, and K. Xu, "WSAP: provide loss rate differentiation with active queue management," in *Proceedings of the International Conference on Communication Technology (ICCT '03)*, pp. 385–391, July 2003.

[10] X. Zhou, D. Ippoliti, and T. Boult, "HPPD: a hop-count probabilistic packet dropper," in *Proceedings of the IEEE International Conference on Communications (ICC '06)*, pp. 116–121, July 2006.

[11] Y. Cao and V. O. K. Li, "Scheduling algorithms in broad-band wireless networks," *Proceedings of the IEEE*, vol. 89, no. 1, pp. 76–87, 2001.

[12] E. H. Mamdani, "Application of fuzzy algorithms for simple dynamic plant," *Proceedings of the Institution of Electrical Engineers*, vol. 121, no. 12, pp. 1585–1588, 1974.

[13] E. H. Mamdani and S. Assilian, "An experiment in liquistic synthesis with a fuzzy logic controller," *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1–13, 1975.

[14] L. Zadeh, "Soft computing and fuzzy lgic," *IEEE Software*, vol. 11, no. 6, pp. 48–56, 1994.