Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/090460

Priority-Aware Fair Queueing for QoS Provisioning in the Internet

*Jui-Pin Yang**

Department of Information Technology and Communication, Shih-Chien University, Kaohsiung 84552, Taiwan, R.O.C.

Received: 29 Nov. 2014, Revised: 1 Mar. 2015, Accepted: 3 Mar. 2015
Published online: 1 Jul. 2015

Abstract: Fair bandwidth sharing and differentiated drop precedence are two key QoS provisioning in the Internet. However, they both are studied separately. In the paper, we propose a simple active queue management scheme that deals with fair bandwidth sharing and differentiated drop precedence at the same time namely priority-aware fair queueing (PAFQ). PAFQ uses a dynamic threshold to keep up with traffic variations and hence it can efficiently discriminate non-aggressive flows from aggressive flows. Furthermore, a swap policy conditionally exchanges the records of current queue lengths of both compared packets from the same flow by considering respective drop precedence. Accordingly, this policy provides differentiated drop precedence within a flow. Besides, a mark policy is used to selectively mark a packet with the maximum count of current queue length that contributes to achieve fair bandwidth sharing among competing flows. When a marked packet reaches at the head of FIFO buffer, it will be discarded directly. Simulation results validate that the PAFQ is able to provide excellent fair bandwidth sharing and differentiated drop precedence under a variety of traffic conditions. In addition, it keeps average queue lengths low.

Keywords: fair queueing, active queue management, bandwidth sharing, drop precedence

1 Introduction

The TCP is a well-known transmission protocol in the Internet because it is beneficial to improve traffic congestion. However, many real-time applications that base on UDP protocol such as VoIP (Voice over IP), VOD (Video on Demand) or network games have become the mainstream of bandwidth consumption. The main difference between the TCP and UDP protocols is that the TCP will cause a flow to reduce its sending rate once there are several lost packets. On the other hand, the UDP will cause a flow to send packets, as usual. When they both compete with network bandwidth in the meantime, those flows that base on the UDP always seize more bandwidth than that of the TCP. This situation not only results in unfair bandwidth sharing but also damages the effectiveness of congestion control schemes [1,2].

In accordance with the above issue many studies have been proposed. They could be coarsely classified into two categories, namely scheduling algorithms and queue management algorithms [3]. Generalized Processor Sharing (GPS) and Deficit Round Robin (DRR) belong to the scheduling algorithms, which are able to achieve perfect fairness on bandwidth sharing among competitive flows [4,5,6,7,8]. However, they should collocate with well-designed buffer management schemes such as Pushout (PO), Partial Sharing and Partial Partitioning (PSPP) and Threshold-based Selective Drop (TSD) so as to prevent fairness from being degraded [9,10,11]. The queue management algorithms such as Random Early Detection (RED), Core-Stateless Fair Queueing (CSFQ) and CHOKe (Choose and Keep for responsive flows, CHOose and Kill for unresponsive flows) come with the tide of fashion [12,13,14,15,16,17,18,19,20]. Their main idea is to determine arrival packets to be accepted or discarded according to specific queue status. Furthermore, they often accompany with a simple FIFO scheduling. Accordingly, the queue management algorithms have received more attentions than the scheduling algorithms.

Differentiated Services (Diffserv) architecture was proposed in order to satisfy the demands with respect to Quality of Service (QoS) [21]. This architecture is simple and scalable because it is unnecessary to maintain per-flow status in the network routers. Inversely, Integrated Services (Intserv) architecture has to collocate with Resource ReSerVation Protocol (RSVP), so it has to maintain per-flow status [22,23]. By marking packets

* Corresponding author e-mail: juipinyang@gmail.com

with proper parameters and concatenating each differentiated service domain, an end-to-end path is practicable in Diffserv. Furthermore, two related packet forwarding mechanisms that include expedited forwarding (EF) and assured forwarding (AF) have been standardized. The latter is composed of four classes and per class has three kinds of drop precedence corresponding to Green, Yellow and Red respectively. Most studies such as RIO (RED with In/Out), Adaptive RIO (A-RIO), RIO-C (RED with In/Out and Coupled queue) have been used to accomplish differentiated drop precedence [24,25]. In summary, most well-known schemes only cope with either fair bandwidth sharing or differentiated drop precedence alone. In other words, an active queue management scheme that can deal with two issues at the same time is attractive and practicable. Therefore, we propose the priority-aware fair queueing (PAFQ) scheme herein.

When a packet arrives at a congested router, the PAFQ compares the arriving packet with certain of resided packets according to dynamic threshold. If total queue length of unmarked packets is smaller than dynamic threshold, the arriving packet will be admitted to enter the buffer and no additional operation is needed. On the other hand, the arriving packet may need to swap the records of current queue length with that of some resided packet from the same flow according to priority (drop precedence) comparison. Next, a packet with the maximum current queue length will be marked. The main reason is that the aggressive flows often have relatively longer queue lengths and hence their resided packets should be marked more frequent. When a marked packet reaches at the head of the FIFO buffer, it will be discarded directly. In other words, only those unmarked packets are eligible to be transmitted. The PAFQ is simple to implement because it only maintains the status of competitive flows. Furthermore, it is able to provide excellent fairness and notable differentiated drop precedence while the average queue length is low. In summary, the PAFQ is very suitable for high-speed and high-performance routers in the future Internet.

The rest of the paper is organized as follows. In Section II, we review related work in association with fair bandwidth sharing and differentiated drop precedence in order. Section III presents the comprehensive details of the priority-aware fair queueing including PAFQ algorithm, packet scheduling and buffer management, weighted PAFQ, implementation complexity and performance matrices. Section IV shows the performance evaluation with respect to fair bandwidth sharing, differentiated drop precedence and average queue length respectively. In Section V, we summarize our conclusions and explain future work.

2 Related work

In this section, we review some related algorithms that have been proposed to enable fair bandwidth sharing between competitive flows or differentiated drop precedence for those packets with different priorities in sequence. The DRR assigns a virtual dedicated queue to each active flow and then resided packets in each virtual queue are served in a round-robin fashion dependent on available quantum size [6]. Besides, the DRR uses the PO buffer management scheme to manage buffer usage [9]. In the PO, an arriving packet will push out a resided packet from the longest queue when the buffer is full. Otherwise, no constraint is imposed on the arriving packet. In general, the PO achieves the best buffer utilization and improves packet loss performance, but it has to find the longest flow queue out and executes very frequent pushout operations especially for congested traffic. In order to improve complexity of PO, several dynamic threshold buffer management schemes were proposed such as PSPP and TSP [10,11]. Although the DRR is capable of perfect fair bandwidth sharing in its entirety, they both DRR and PO have to maintain per-flow state. Therefore, the combination is too difficult to implement in the high-speed routers.

The RED uses a single FIFO queue to accommodate arriving packets from each flow. The arriving packets may encounter different drop probabilities according to average queue size and some of parameters [12]. Since the RED discards packets early, it effectively prevents the TCP connections from global synchronization. However, the RED is unable to provide fairness among competitive flows because the aggressive flows will grab more bandwidth from that of the non-aggressive flows. This reason is that arriving packets for aggressive flows or non-aggressive flows all possess the same drop probability in the meantime. Based on the idea of the RED, several variants have been proposed in order to enhance the robustness of parameters and degree of fair bandwidth sharing [13,14,15].

In the CSFQ [16], the edge routers have to maintain per-flow state and insert the state (flow arrive rate) into corresponding packet headers. When a core router receives a packet, it has to estimate fair share rate and then decide to accept or discard the arriving packet according to a simple probabilistic model. The CSFQ is able to provide reasonable fairness; moreover, it pushes high complexity toward the edge routers that greatly ease the sophisticated implementation in the core routers. Consequently, the CSFQ is feasible to be deployed in the network environments that consist of high-speed core routers and medium-speed edge routers. RFQ scheme consists of packet coloring and buffer management, and also aims at providing fair bandwidth sharing [17]. In the first place, it should classify flow arriving rate into a set of layers, with a globally consistent color per layer. The edge routers have to insert the color into corresponding packet headers. When a packet arrives at a core router, the arriving will be discarded only if its color level over a color threshold. The color threshold dynamically changes in accordance with traffic variations. As compared with the CSFQ, the RFQ provides approximate fairness but it only carries with simple color rather than explicit flow arrival rate. Furthermore, it wards off exponential averaging estimation while a packet is generating. Both CSFQ and RFQ schemes divide the routers into edge or core and they have to maintain per-flow state in the edge. Next, a simple active queue management scheme is unneeded to maintain per-flow state in the edge routers and core routers, that is, CHOKe [18]. It randomly compares the arriving packet with a resided packet. If both packets originate from the same flow, they are discarded simultaneously. Otherwise, the arriving packet will go through the same procedures of the RED. It is obvious that the aggressive flows have more resided packets than that of non-aggressive flows, so their resided packets are more likely to be draw for comparisons. The CHOKe will cause the resided packets of aggressive flows a higher probability to be discarded, so it partly obtains better fairness than that of RED. In summary, the above mentioned schemes can only provide fair bandwidth sharing but they are all incapable of dealing with packets with differentiated drop precedence.

RIO is viewed as an extended version of the RED that possesses the capability of differentiated drop precedence [24]. It uses two sets of parameters to differentiate the drop precedence with respect to In and Out packets. In the first place, the RIO calculates average queue size including In and Out packets in order to decide the treatment with respect to Out packets. For In packets, the RIO merely calculates the average queue size including In packets. Obviously, the In packets have lower drop precedence as compared with the Out packets when the traffic is congested. To correspond with requirements differentiated services [21,22,23], the RIO must be revised to handle three kinds of drop precedence. In contrast to the RIO, WRED supports preferential treatment on higher priority packets by combining the capabilities of the RED with IP Precedence, and then selectively discards lower priority packets along with the increment of queue lengths [25]. The WRED needs to calculate average queue size of a physical buffer and adequately set two minimum thresholds.

3 Priority-Aware Fair Queueing

In this section, we orderly explain the priority-aware fair queueing (PAFQ) that are composed of five subsections including PAFQ algorithm, packet scheduling and buffer management, a weighted version of PAFQ, implementation complexity and performance matrices.

3.1 PAFQ Algorithm

The PAFQ algorithm can provide fair bandwidth sharing among competitive flows and differentiated drop precedence within a flow. In addition, it keeps low average queue lengths. The PAFQ algorithm is one of the active queue management schemes, so it is simple to implement, too. A flow chart of the algorithm is given in Fig. 1. When a new packet arrives at a router, the PAFQ algorithm evaluates whether residual buffer size is able to accommodate the new arriving packet. If the buffer size is insufficient, then the arriving packet will be discarded immediately without any additional operation. Otherwise, the algorithm calculates the sum of current queue length of unmarked packets plus packet size itself, which is denoted as Q_{unmark} . For all packets, they must be classified as mark or unmark type. The main difference between the unmarked packet and marked packet is that the former is eligible to be transmitted when it stays at the head of FIFO buffer. Inversely, the latter will be discarded immediately at that time. If $Q_{unmark} < Th_{new}$, the arriving packet will be admitted to enter the buffer and identified as unmark type. Furthermore, the PAFQ algorithm needs to maintain type, drop precedence level (Green, Yellow or Red) and current queue length of the arriving packet. The Th_{new} is a dynamic threshold that adjusts to traffic variations. Consequently, the PAFQ algorithm is able to determine the adequate number of comparisons with respect to the unmarked packets that efficiently discriminate the non-aggressive flows from aggressive flows.

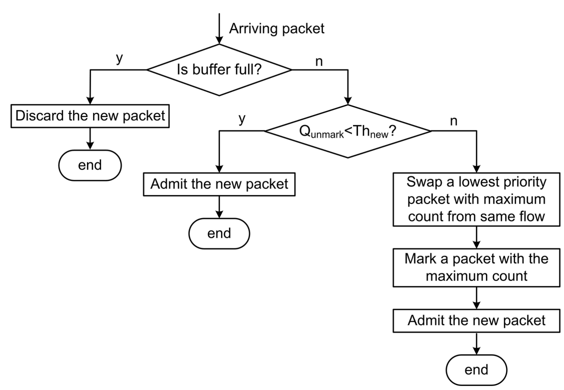

Fig. 1: PAFQ algorithm

We use Equations (1) and (2) to estimate the Th_{new} . In the first place, the PAFQ algorithm sums up the difference between maximum and minimum current queue lengths of unmarked packets once a packet arrives within a fixed time interval. Let T_c denote the static duration of a time interval (ms). Furthermore, $Max(Q_i)$ and $Min(Q_i)$ denotes the maximum and minimum current queue lengths of unmarked packets respectively when ith packet arrives. Also, m denotes the amount of arriving packets that are admitted to enter the buffer during a specific time interval. Finally, we obtain $W_{current}$ that represents the mean of maximum difference in queue lengths of competitive flows. In general, a larger $W_{current}$ means that superior discrimination of queue lengths between aggressive flows and non-aggressive flows. In this paper, the definition of a flow is that it possesses the same pair of source and destination IP addresses.

$$W_{current} = \frac{\sum_{i=1}^{m}[Max(Q_i) - Min(Q_i)]}{m} \quad (1)$$

In order to elude meaningless adjustment in Th_{new} , we define a H_p parameter which is used to judge whether the Th_{new} should be altered or not. When a packet arrives and current queue length of unmarked packets equals Th_{old} , then a hit happened. Next, we add hit number up in turn until at the end of time interval and then calculate the hitting probability that is equivalent to total hit numbers divided by the amount of arriving packets. If the hitting probability is larger than the H_p , we use Equation (2) to estimate the Th_{new} further. On the other hand, it is unnecessary to change the Th_{new} , namely $Th_{new} = Th_{old}$. If $min_{th} \leq W_{current} \leq max_{th}$, the Th_{new} keeps the same as the Th_{old} because the expected flow discrimination has been reached already. The $min_{th}$and max_{th} are two control thresholds that represent minimum and maximum limits of $W_{current}$ respectively. If $W_{current} < min_{th}$ or $W_{current} > max_{th}$, we let the ratio of Th_{new} and Th_{old} that is proportional to $W_{target}/W_{current}$ where $W_{target} = (min_{th} + max_{th})/2$. In a word, the method that we use to estimate the Th_{new} is beneficial to speed up convergence and increase stability.

$$Th_{new} = \begin{cases} Th_{old} & min_{th} \leq W_{current} \leq max_{th} \\ \frac{W_{target}}{W_{current}} \cdot Th_{old} & \text{else} \end{cases} \quad (2)$$

If $Q_{unmark} \geq Th_{new}$, the PAFQ algorithm finds an unmarked packet in the buffer that possesses the highest drop precedence with minimum count of current queue length from the same flow. If the drop precedence level of the arriving packet is smaller than that of the candidate, it needs to compare both counts beyond. If the count of the arriving packet is larger, they both swap the count values. Next, the arriving packet is admitted to enter the buffer and labeled as unmark type. Otherwise, no exchange occurs. The swap policy gives better preference to the lower drop precedence packets within a flow, so that it attains to differentiated drop precedence. Here, Green, Yellow and Red drop precedence are mapping to the high, medium and low priority packets respectively. Finally, a mark policy is used to label an unmarked packet with the maximum count as mark type. By marking that packet, the PAFQ is able to provide fair bandwidth sharing.

3.2 Packet Scheduling and Buffer Management

The PAFQ uses a FIFO packet scheduling algorithm to transit the resided packets. When a marked packet reaches at the head of the buffer, it will be discarded immediately. The same procedures repeat until there is an unmarked packet. The PAFQ uses drop tail buffer management to manage the buffer usage. In another word, those marked packets also occupy the buffer space. If the PAFQ equips with insufficient buffer size, it may danger to the performance of fair bandwidth sharing and differentiated drop precedence. However, the PAFQ always keeps average queue lengths of unmarked packets low even if the buffer size is large. The reason is that the dynamic threshold precisely and adequately changes according to traffic variations. On the other hand, the PO buffer management scheme is applicable to the PAFQ that contributes to cope with small buffer size. The way is to replace the mark policy by the pushout policy simply. When the queue length of unmarked packet is equal or larger than the dynamic threshold, the PO has to execute a pushout operation.

It means that PO needs countless pushout operations when the traffic is extremely congested. In order to keep consistent simplicity as the PAFQ, we choose the simple FIFO scheduling and drop tail buffer management scheme herein.

3.3 Weighted PAFQ

The PAFQ algorithm could be easily extended to support flows with proportional fair share rate. Let α_j denote the weight of flow j. When a packet of flow j arrives, its new count equals original count divided by α_j . Furthermore, the new count is used to estimate the $W_{current}$. For instance, flow j will obtain approximately twice fair share rate when α_j is set at 2. Besides, we can enhance the weighted PAFQ to guarantee certain throughput of higher drop precedence such as Yellow and Red packets. Let $S_{Yellow,j}$ and $S_{Red,j}$ denote the maximum allowable number of swap for the Yellow and Red packets of flow j respectively. When a Green packet of flow j is arriving, a Red packet of flow j that possesses the lowest count from the same flow will be swapped. If the swapped number of a specific Red packet has exceeded the $S_{Red,j}$, then a Yellow packet of flow j is the next candidate subject to $S_{Yellow,j}$. Similarly, a Yellow packet is able to swap a Red packet from the same flow j subject to $S_{Red,j}$. In contrast to original PAFQ, both $S_{Yellow,j}$ and $S_{Red,j}$ equal Th_{new} . It means that packets with lower drop precedence can swap those packets with higher drop precedence in the same flow without any limitation. As a result, it may cause those packets with higher drop precedence bandwidth starvation.

3.4 Implementation Complexity

At each router, both the time and space complexity of the PAFQ algorithm are constant with respect to the competitive flows, and thus it is suitable for high-speed routers. When a packet arrives at a router which needs to (1) compare the arriving packet with unmarked packets in the buffer and then obtain the packet count from the same flow, (2) find current maximum and minimum count from unmarked packets, (3) estimate the $W_{current}$ and then Th_{new} (4) swap the counts of both packets, and (5) mark a packet with maximum count. All above operations are simple to implement nowadays. In contrast to the proposed PAFQ, both CSFQ and RFQ need to execute sophisticated packet classification algorithms at each edge router. In a word, the PAFQ is relatively easy to implement and simplify network configurations.

3.5 Performance Metrics

In order to investigate the performance of the PAFQ, we define two performance matrices as the measurement benchmarks of fair bandwidth sharing and differentiated drop precedence. They both are composed of normalized bandwidth ratio and packet loss probability (PLP) respectively. The former is used to validate the degree of fair bandwidth sharing among different flows and the latter is used to demonstrate loss performance of per drop precedence within a flow. Next, we introduce the definitions of normalized bandwidth ratio and packet loss probability respectively. The first definition is associated with the max-min fairness **[25]** which is presented in Equation (3). In this equation, the N denotes the number of flows and the f denotes the max-min fair rate. In addition, the r_i denotes the mean arrival rate of flow i.

$$\sum_{i=1}^{N} min\{r_i, f\} = C \quad (3)$$

By deriving the f from equation (3), we use equation (4) to define the normalized bandwidth ratio where NBR_i denotes the normalized bandwidth ratio of flow i and M_i denotes the mean departure rate of flow i. In the ideal case, the normalized bandwidth ratio for all flows is equal to 1.

$$MBR_i = M_i / min\{r_i, f\} \quad (4)$$

Next, the definition of PLP is shown in Equation (5). $PLP_{i,j}$ represents the PLP of flow i with drop precedence j. Besides, we also study the behavior of average queue lengths so as to validate that the PAFQ is able to keep low average queue lengths regardless of buffer size.

$$PLP_{i,j} = D_{i,j} / A_{i,j} \quad (5)$$

$D_{i,j}$:Total discarded packets of flow i with drop precedence j

$A_{i,j}$:Total arriving packets of flow i with drop precedence j

4 Simulation Results

We consider a single congested link topology where the capacity of each link is of 10 Mbps. In addition, the buffer size is of 1024 KB and there are 10 competitive flows. The initial settings of other parameters are as follows; $min_{th} = 5$, $max_{th} = 7$, $H_p = 0.05$, $T_c = 80$ ms and $Th_{new} = 16$. Each flow generates infinite packets from a specific ON-OFF traffic model. For a generated packet, the probability of being Green, Yellow and Red drop precedence is set at 0.2, 0.3 and 0.5 respectively. To simplify computer simulations, all packet size is set at 1 KB and total simulation time is of 100 seconds. We run 100 times in each circumstance so as to get reliable simulation results. Finally, the policy that we select an unmarked packet as mark type is from tail to head. Unless otherwise specified, we use the previous statements for the following simulations all the time. The given schemes only consider either fairness or differentiated drop precedence, so that there is no proper candidate as compared with the PAFQ. Accordingly, we profoundly and comprehensively study the performance of fair bandwidth sharing and differentiated drop precedence with respect to the PAFQ. Also, we validate that PAFQ keeps low average queue lengths regardless of buffer size. Next, they will be studied in 4.1, 4.2 and 4.3 subsections in sequence.

4.1 Fair Bandwidth Sharing

Fig. 2 shows the normalized bandwidth ratio (NBR) versus per flow under different buffer sizes. Furthermore, all flows are indexed from 1 to 10 in order. The respective mean arriving rate of per flow is set at 1, 2, 3, 4, 5, 6, 7, 8, 9 and10 Mbps, so that the max-min fair rate equals 1 Mbps. In Fig. 3, the red dash line represents the optimal fairness because the NBRs of all flows equal 1.0. When buffer size is set at 128 KB, the NBRs of flow 1 to flow 3 are smaller than 1.0, flow 1 especially. On the other hand, the NBRs of flow 4 to flow 10 are larger than 1.0. This leads to the worst fairness because buffer size is severely insufficient. The PAFQ uses drop tail buffer management scheme to manage buffer usage. Accordingly, it greatly limits the PAFQs efficiency to fairness because the buffer is full with the highest probability. When buffer size is set at 256 KB, flow 1 to flow 3 all have larger NBRs. For instance, the NBR of flow 1 increases from 0.48 to 0.74 and the NBR of flow 2 increases from 0.80 to 0.98. On the contrary, the NBRs of flow 4 to flow 10 averagely decrease from 1.11 to 1.03. When the buffer size is set at 512 KB, the PAFQ roughly gets rid of the effect of a full buffer. The NBR of flow 1 equals 0.88 and the others equal 1.0 respectively. When the buffer size is set at 1024 KB or 2048 KB, it doesnt improve the NBR of each flow. It means that the effect of buffer size is negligible with respect to the PAFQ. The PAFQ uses dynamic threshold Th_{new} to choose the adequate number of comparisons that accomplishes the traffic discrimination, so it prevents the PAFQ from optimal fairness. However, the PAFQ shows excellent fairness without excessive complexity.

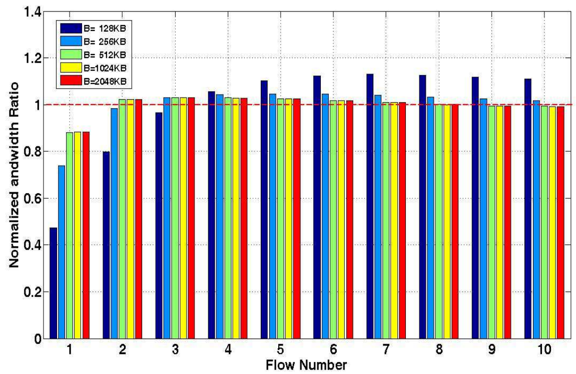

Fig. 2: Normalized bandwidth ratio versus per flow under different buffer sizes

Fig. 3 shows the normalized bandwidth ratio versus per flow where their mean arriving rate is only composed of 0.5 Mbps or 10 Mbps. In the [(0.5Mbps, 9), (10Mbps, 1)] case, it means that flow1 to flow 9 are of 0.5 Mbps and flow 10 is of 10 Mbps. The similar representations are applied to the following traffic conditions. In Fig. 4, we found that the NBR of each flow is very close to 1.0 in all kinds of traffic conditions. In other words, the PAFQ provides approximately optimal fairness. The main reason is that the difference of mean arriving rate between non-aggressive flows (0.5 Mbps) and aggressive flows (10 Mbps) is explicit, so that the PAFQ is more effective to differentiate and then protect the arriving packets of the non-aggressive flows. When the number of 10 Mbps flows increases, it causes the NBRs of 0.5 Mbps flows little decrement. The unmarked packets of 0.5 Mbps flows have a higher probability to be modified as mark type because of a larger dynamic threshold Th_{new} .

Fig. 4 shows the normalized bandwidth ratio versus per flow where their mean arriving rate is only composed of 2 Mbps or 6 Mbps. In Fig. 4, it is quite obvious that the NBR of each flow is very close to 1.0 in all kind of traffic conditions, too. The difference of mean arriving rate between 2 Mbps flows and 6 Mbps flows is smaller than that of Fig. 3. However, the 2 Mbps flows have higher arriving rate over the max-min fair rate (1 Mbps) and the 6 Mbps flows have less aggressive than that of 10 Mbps flows. Consequently, the PAFQ also provides

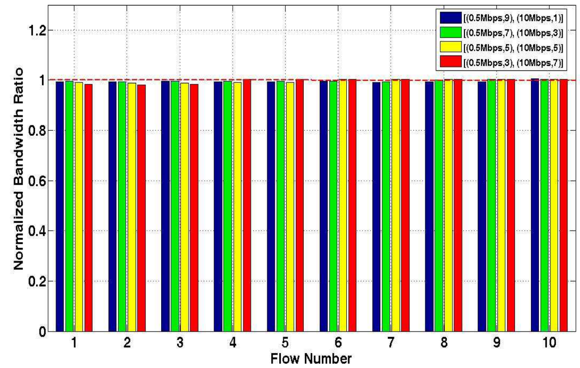

Fig. 3: Normalized bandwidth ratio versus per flow where their mean arriving rate is only composed of 0.5 Mbps or 10 Mbps

approximately optimal fairness. From Fig. 2 to Fig. 4, we conclude that the PAFQ is able to provide robust and excellent fair bandwidth sharing under various traffic loads.

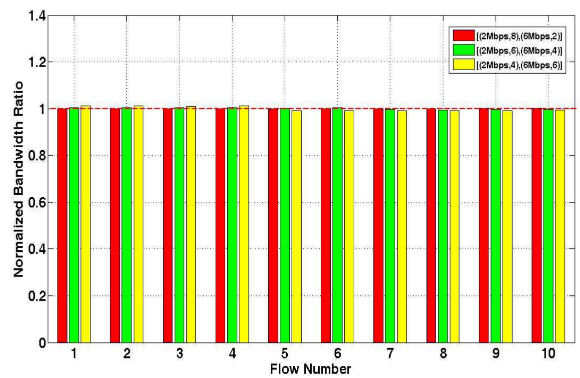

Fig. 4: Normalized bandwidth ratio versus per flow where their mean arriving rate is only composed of 2 Mbps or 6 Mbps

Fig. 5 shows the normalized bandwidth ratio versus per flow under different buffer sizes. Furthermore, there are thirty flows and then indexed from 1 to 30. The mean arriving rate of each flow is set at 1/3, 2/3, 1, 4/3, 5/3, 2, .. and 10 Mbps respectively, hence the max-min fair rate equals 1/3 Mbps. When the buffer size is set at 256 KB, the NBRs of flow 1 to flow 9 are smaller than 1.0. On the other hand, the NBRs of flow 10 to flow 30 are larger than 1.0. Evidently the PAFQ performs worse fairness than that of Fig. 3, because the buffer size is relatively insufficient herein. The total mean arriving rate is of 55 Mbps in Fig. 2, but that is of 155 Mbps in Fig. 5. When buffer size increases, the PAFQ has better fairness correspondingly. When the buffer size is set 4096 KB, the PAFQ provides the best fairness because the buffer size is sufficient. The NBRs of all flows are around 1.0 except for the flow 1 whose NBR is equivalent to 0.85. As we mentioned in subsection 3.2, the PO buffer management scheme can be used to improve performance degradation because of small buffer size. However, this issue is out of scope of this study. The total mean arriving rate 155 Mbps is about 3 times of 55 Mbps, so we suggest that the ideal buffer size could be set at the product of 512 KB (the suitable buffer size in Fig. 2) and 8 (2^3), that is, 4096 KB. In summary, the PAFQ still provides excellent fairness even if the number of flows increases and each flow has violent variation on traffic loads.

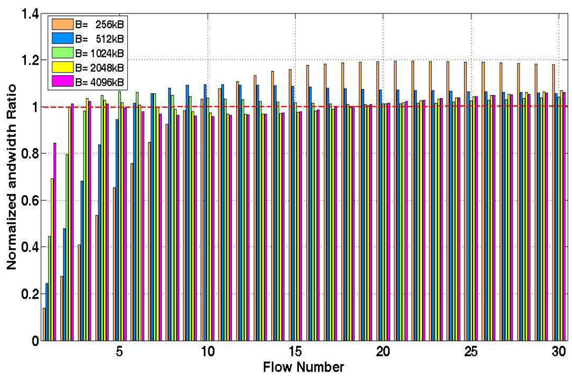

Fig. 5: Normalized bandwidth ratio versus per flow under thirty flows and different buffer sizes

Fig. 6 shows the normalized bandwidth ratio versus per flow where the burstiness of each ON-OFF flow becomes 2, 3 or 4 times simultaneously. Furthermore, the other simulation configurations are identical with that of Fig. 2. When the burstiness increases, the NBRs of flow 6 to flow 10 increase. On the other hand, the NBRs of flow 1 to flow 5 decreases. When the burstiness changes 4 times especially, the flow 1 gets the smallest NBR at 0.57. On the other hand, flow 10 gets the largest NBR at 1.20. The reason is that the higher burstiness not only leads to a smaller dynamic threshold but also enlarges current queue lengths of the non-aggressive flows. In general, different degrees of burstiness mostly cause different levels of fairness degradation for active queue management schemes. Although the PAFQ only provides acceptable fairness under such large traffic burstiness, it reduces the number of packet comparisons that also contributes to low average queue lengths. We can change the min_{th} and max_{th} to improve fairness, if necessary.

Fig. 7 shows the normalized bandwidth ratio versus per flow under different values of (min_{th},max_{th}).

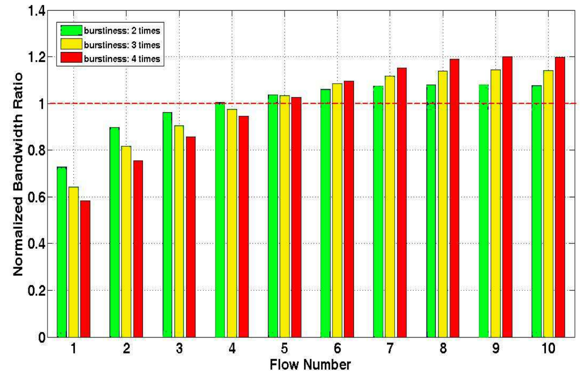

Fig. 6: Normalized bandwidth ratio versus per flow where the burstiness of each flow changes in the meantime

Furthermore, the other simulation configurations are identical with that of Fig. 2. When the (min_{th},max_{th}) pair are set at (1, 3), the NBR of the most aggressive flow 10 equals 1.15 but the NBR of the least non-aggressive flow 1 equals 0.66. It means that traffic discrimination among flows is insufficient. When the (min_{th},max_{th}) pair are set at (3, 5), the PAFQs fairness is apparently improved. Most importantly, the PAFQ provides the best fairness when the (min_{th},max_{th}) pair are set at (5,7). When the (min_{th},max_{th}) are set at (9, 11), flow 10 gets the smallest NBR, namely 0.93. Otherwise, flow 2 gets the largest NBR, namely 1.09. Flow 10 is the most aggressive flow, so its queue length is usually longer than that of the others. This causes excessive amount of resided packets of flow 10 to be marked and then discarded. As for flow 2, it roughly has the second smallest queue length while its mean arriving rate is larger than max-min fair rate (1 Mbps). As a result, those packets of flow 2 possess the highest chance to occupy the buffer. According to the same principle, flow 3 obtains the second largest NBR, that is, 1.07.

Fig. 8 shows normalized bandwidth ratio versus per flow under different gaps of (min_{th},max_{th}). Furthermore, the other simulation configurations are identical with that of Fig. 7. In Fig. 8, the fairness is robust and remarkable except the (min_{th},max_{th}) pair are set at (1, 11). The main reason is that $W_{current}$ quickly reaches the lower limit, namely 1. Accordingly, the Th_{new} is a small and stable value because the upper limit is 11. In another word, the PAFQ is unable to reach sufficient traffic discrimination by identifying queue lengths. As usual, the most aggressive flows obtain larger NBRs as compared with that of the non-aggressive flows.

Figure 9 shows normalized bandwidth ratio versus per flow where packets have different ratios of drop precedence. Furthermore, the other simulation configurations are identical with that of Fig. 2. In Fig. 9, the ratio of drop precedence affects the PAFQs fairness.

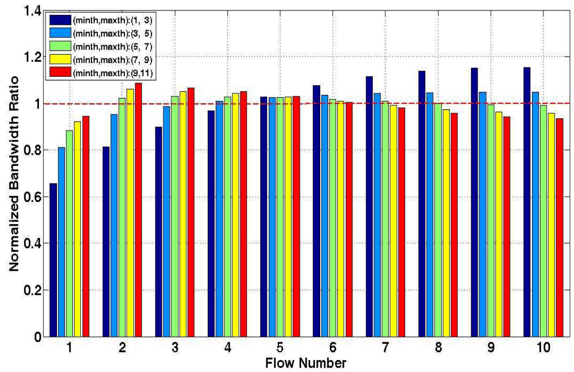

Fig. 7: Normalized bandwidth ratio versus per flow under different values of min_{th} and max_{th}

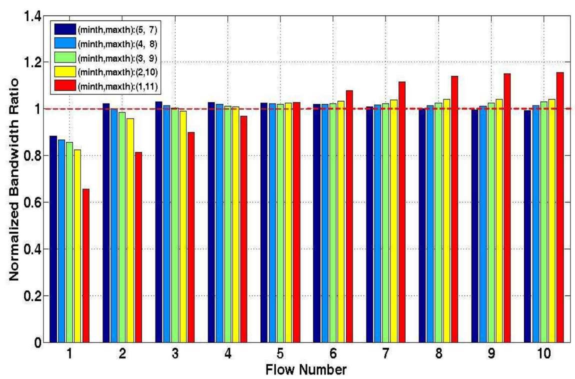

Fig. 8: Normalized bandwidth ratio versus per flow under different gaps between min_{th} and max_{th}

When the ratio changes from (1, 2, 7) to (8, 1, 1), for instance, the NBR of flow 1 decreases. The number of low drop precedence (Green packets) increases and the number of higher drop precedence (Yellow and Red packets) decreases, so the packets with low drop precedence have less opportunity to swap with that of higher drop precedence. Therefore, flow 1 has smaller NBR. In a word, the PAFQ also provides excellent fairness no matter which ratios of drop precedence exist. From Fig. 2 to Fig. 9, we conclude that the PAFQ is able to provide very robust and approximately optimal fairness under various traffic conditions.

4.2 Differentiated Drop Precedence

Fig. 10 shows the packet loss probability of different drop precedence versus per flow. Furthermore, all simulation configurations are the same as that of Fig. 2 except we

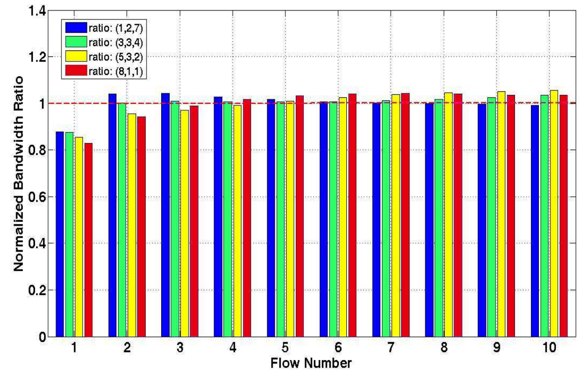

Fig. 9: Normalized bandwidth ratio versus per flow under different ratios of drop precedence

only consider the buffer size are of 128 KB, 512 KB and 2048 KB respectively. In Fig. 10, the first Green, Yellow and Red bars in each flow represent Green, Yellow and Red drop precedence respectively when buffer size is set at 128 KB. Similarly, the second Green, Yellow and Red bars in each flow represent the Green, Yellow and Red drop precedence respectively when buffer size is set at 512 KB. We use the same rule to connect illustrative legend and resultant bars. In the first place, we explain the PLPs of flow 1 with different drop precedence whose mean arriving rate is equal to 1 Mbps. When the buffer size is set at 128 KB, it leads to drop-tail behavior for all flows especially for flow 1. Consequently, the PLPs of Green, Yellow and Red drop precedence are near.
When the buffer size is set at 512 KB or 2048 KB, flow 1 shows differentiated drop precedence by eliminating the effect of buffer size. Accordingly, the PLP of Green packets is smaller than that of Yellow packets and the PLP of Yellow packets is relatively smaller than that of Red packets. The reason is that the PAFQ fully uses the swap policy to reduce packet loss of lower drop precedence such as Green and Yellow packets.

Next, let us look into the PLPs of flow 2. When the buffer size is set at 128 KB, the PLPs of Green and Yellow packets are lower than that of Red packets as compared with flow 1 because flow 2 has more Red packets to be swapped. When the buffer size is set at 512 KB or 2048 KB, the PLP of Green packets is relatively smaller than that of Yellow packets and the PLP of Yellow packets is relatively smaller than that of Red packets. In another word, it achieves better differentiated drop precedence. The reason is that Green packets have more chance to swap with Yellow and Red packets and Yellow packets has more chance to swap with Red packets, too. Finally, let us look into the PLPs of flow 10. When the buffer size is set at 128 KB, the difference of PLPs of Green and Yellow packets are more obvious than that of flow 1 and flow 2. Flow 10 has the maximum amount of Yellow and Red packets, so the swap policy is functional even if the buffer size is insufficient. When the buffer size is set at 512 KB or 2048 KB, the PLPs of Green, Yellow and Red packets equal 0.56, 0.96 and 1.0 respectively. The mean arriving rate of Green, Yellow and Red packets is composed of 2Mbps, 3 Mbps and 5 Mbps, and the max-min fair rate is of 1 Mbps. In the ideal situation, the PLPs of Green, Yellow and Red packets should be equal to 0.5, 1.0 and 1.0 respectively. In a word, the PAFQ is able to provide notable differentiated drop precedence indeed.

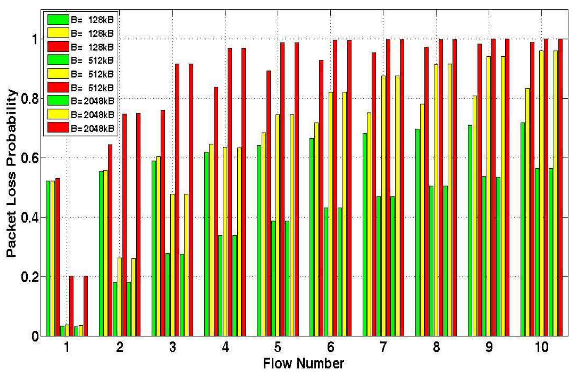

Fig. 10: Packet loss probability of different drop precedence versus per flow under ten flows with different buffer sizes

Fig. 11 shows the packet loss probability of different drop precedence versus per flow and the simulation configurations are the same as that of Fig. 3. In the beginning, we focus on the PLPs of 0.5 Mbps flows with different drop precedence. In all traffic conditions, the 0.5 Mbps flows have very low PLPs with respect to Green, Yellow and Red packets. When the number of 10 Mbps flows increases, their aggression causes the 0.5 Mbps flows higher PLP of Red packets because of swap policy. As for 10 Mbps flows, they always have more packets resided in the buffer. Consequently, the swap policy performs more efficient that enhances the performance of differentiated drop precedence. Finally, each 10 Mbps flow will obtain less max-min fair share rate along with the increment of 10 Mbps flows. Accordingly, their PLPs of Green, Yellow and Red packets roughly increase.

Fig. 12 shows the packet loss probability of different drop precedence versus per flow and the simulation configurations are the same as that of Fig. 4. In all traffic conditions, the PAFQ provides notable differentiated drop precedence. When the number of 6 Mbps flows increases, the Th_{new} increases accordingly. For 2 Mbps flows, both PLPs of Green and Yellow packets decrease because they are more Red packets to be swapped, hence their PLPs of Red packets increase. For 6 Mbps flows, it is different

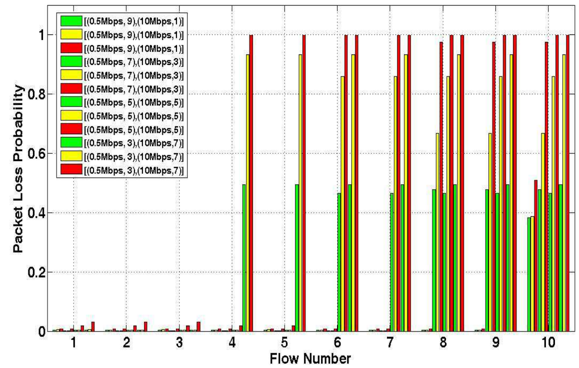

Fig. 11: Packet loss probability of different drop precedence versus per flow where their mean arriving rate equals either 0.5 Mbps or 10 Mbps

from the 2 Mbps flows. We find that only PLPs of Green packets decrease because the mean arriving rate of Green packets (6 Mbps*0.2=1.2 Mbps) has exceeded the max-min fair share rate (1 Mbps).

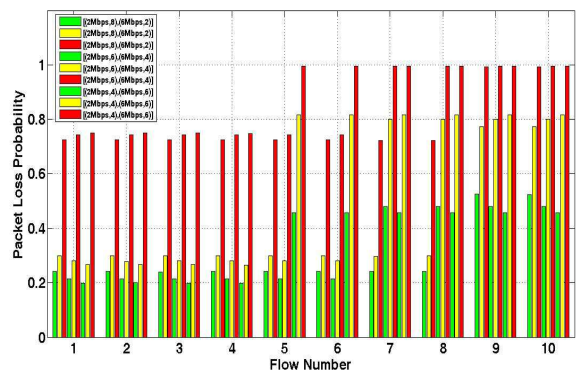

Fig. 12: Packet loss probability of different drop precedence versus per flow where their mean arriving rate only equals either 2 Mbps or 6 Mbps

Fig. 13 shows the packet loss probability of different drop precedence versus per flow and the simulation configurations are the same as that of Fig. 6. When the burstiness increases, the PLPs of Green, Yellow and Red packets of flow 1 to flow 5 increase at the same time because of lower NBRs. On the other hand, flow 6 to flow 10 they all obtain a larger NBR that decrease their PLPs of Green, Yellow and Red packets accordingly.

Fig. 14 shows the packet loss probability of different drop precedence versus per flow and the simulation

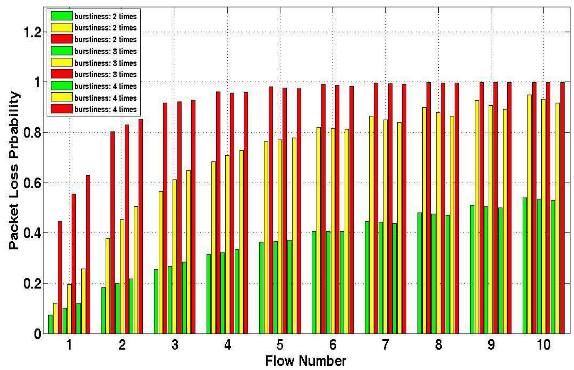

Fig. 13: Packet loss probability of different drop precedence versus per flow where the burstiness of each flow changes in the meantime

configurations are the same as that of Fig. 9. When the ratio of Green packets increases and the ratios of Yellow and Red packets decrease, the PLPs of Green, Yellow and Red packets all increase with respect to each flow.
The reason is that most of Yellow and Red packets are swapped by the Green packets. Consequently, they both PLPs increase. Next, we also find that PLPs of Green packets for each flow increase. The reason is that the ratio of Green packets increases, hence they have higher probability to be marked and then discarded.From Fig. 10 to Fig. 14, we conclude that PAFQ is able to provide notable differentiated drop precedence under a variety of traffic conditions.

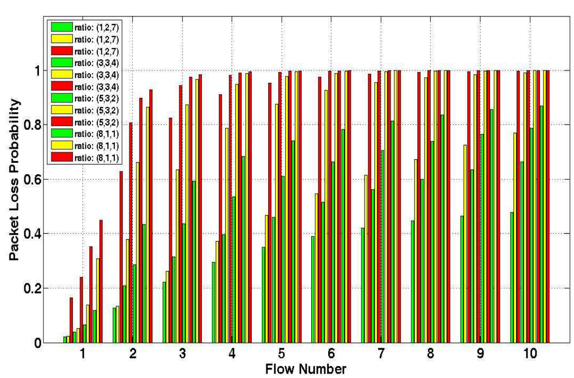

Fig. 14: Packet loss probability of different drop precedence versus per flow under various ratios of drop precedence

4.3 Average Queue Length

Fig. 15 shows average queue length versus time with respect to Fig. 2. Here, we calculate average queue length that only takes current queue length of unmarked resided packets into account once a packet is arriving. In other words, we ignore all marked resided packets because they will be discarded eventually. In Fig. 15, the maximum average queue length is near 90 KB, even if the buffer size is of 2048 KB. It means that large buffer size has very limited effect on average queue lengths. The PAFQ uses Th_{new} to decide whether a packet should be marked or not, hence it completely dominates the growth of average queue length. When the buffer size is set at 128 KB, it has the smallest average queue length as compared with that of other buffer sizes. The buffer size is insufficient, so that the Th_{new} is unable to increase up the adequate value. When buffer size is equal or larger than 512 KB, the average queue length is approximately around 81 KB. In other words, the Th_{new} reaches a stable and proper value according to the setting of min_{th} and max_{th}.

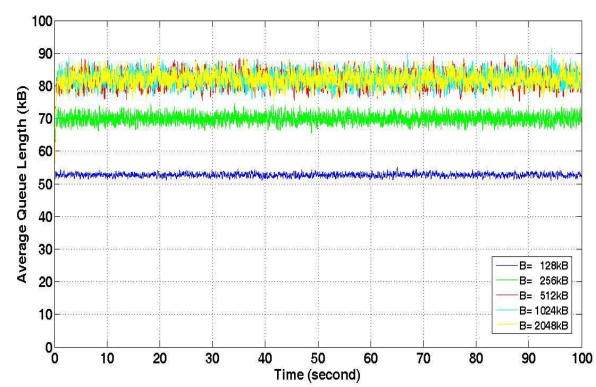

Fig. 15: Average queue length versus time under ten flows with different buffer sizes

Fig. 16 shows average queue length versus time with respect to Fig. 11. When the number of 10 Mbps flows increases from 1 to 3, 5 and 7, it results in a larger Th_{new} in sequence. Accordingly, it increases average queue lengths. The variations of average queue lengths are smaller than that in Fig. 16 because the difference of traffic intensity between 0.5 Mbps and 10 Mbps flows is quite obvious. Therefore, the $W_{current}$ is stable that leads to a stable Th_{new} , too. In a word, the PAFQ can dynamically adjust the Th_{new} to cope with traffic conditions.

Fig. 17 shows average queue length versus time with respect to Fig. 13. When the burstiness increases from 2 to 4 times, the average queue length decreases. The main reason is that $W_{current}$ reaches the interval between min_{th}

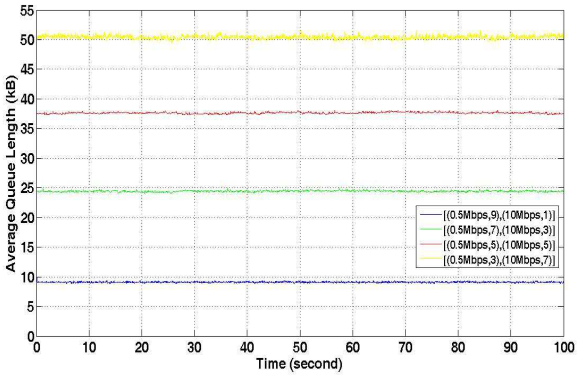

Fig. 16: Average queue length versus time where their mean arriving rate equals either 0.5 Mbps or 10 Mbps

and max_{th} more quickly, so a smaller Th_{new} is produced. From the simulation results, a larger burstiness is beneficial for the PAFQ to decrease average queue length, but it may degrade the fairness and packet loss performance. To overcome performance degradation, we can choose a larger pair of min_{th} and max_{th} . However, it causes a higher average queue length and additional packet comparisons. In a word, the PAFQ is adjustable that depends on required performance targets. From Fig. 15 to Fig. 17, we conclude that PAFQ is able to keep low average queue lengths under different traffic conditions.

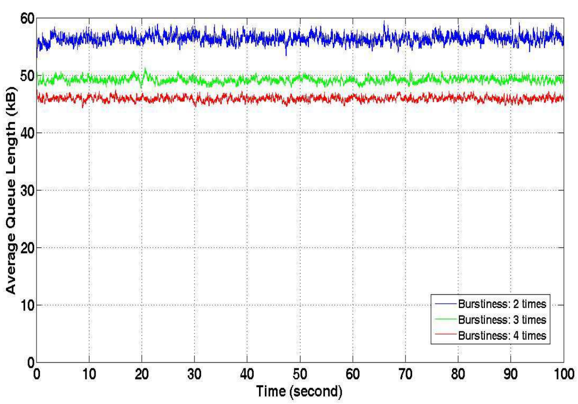

Fig. 17: Average queue length versus time where the burstiness of each flow changes in the meantime

5 Conclusions

In this paper, we propose a simple and efficient active queue management scheme which satisfies the QoS requirements inclusive of fair bandwidth sharing and differentiated drop precedence at the same time namely PAFQ. When network conditions are changing, the PAFQ automatically and properly adjusts the thresholds based on traffic variations. Accordingly, it achieves sufficient traffic discrimination. Next, the PAFQ uses a swap policy to conditionally exchange the count of lower drop precedence with that of higher drop precedence within the same flow. Therefore, it provides differentiated drop precedence within a flow. Besides, the PAFQ uses a mark policy to selectively mark a packet with the maximum count that provides fair bandwidth sharing among competing flows. Simulation results validate that the PAFQ is able to provide excellent fairness and notable differentiated drop precedence under a variety of traffic conditions. In addition, the PAFQ keeps low and stable average queue lengths all the time. In a word, the PAFQ is suitable to be deployed in the routers with high-speed and high-performance requirements. In the future, we would like to apply queueing theory to mathematically analyze the fairness and packet loss probability of different drop precedence in association with the PAFQ. Furthermore, we will consider more complicated network environments that deeply analyze the performance of the PAFQ. Finally, we will study a weighted PAFQ version in order to achieve proportional bandwidth sharing and also enhance differentiated drop precedence.

Acknowledgement

The author acknowledges the financial support from the Shih-Chien University and National Science Council in Taiwan, Republic of China under the grant numbers USC-102-05-05016 and NSC-102-2221-E-158-001 respectively.

References

[1] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski and L. Zhang, Recommendations on queue management and congestion avoidance in the Internet, RFC **2309** (1998).
[2] A. Mankin, K. Ramakrishnan, Gateway congestion control survey, RFC **1254** (1991).
[3] M. Irland, Buffer management in a packet switch, IEEE Transactions on Communications, **26** (3) 328-337 (1978).
[4] F. Kamoun, L. Kleinrock, Analysis of shared finite storage in a computer node environment under general traffic conditions, IEEE Transactions on Communications, **28** (7) 992-1003 (1980).

[5] A. K. Choudhury, E. L. Hahne, Dynamic queue length thresholds for shared-memory packet switches, IEEE/ACM Transactions on Networking, **6** (2) 130-140 (1998).
[6] E. L. Hahne, A. K. Choudhury, Dynamic queue length thresholds for multiple loss priorities, IEEE/ACM Transactions on Networking, **10** (3) 368-380 (2002).
[7] R. Fan, A. Ishii, B. Mark, G. Ramamurthy, Q. Ren, An optimal buffer management scheme with dynamic thresholds, Proceedings of the GLOBECOM, 631-637 (1999).
[8] Y. S. Chu, J. P. Yang, C. S. Wu, M. C. Liang, Partial sharing and partial partitioning buffer management scheme for shared buffer packet switches, IEICE Transactions on Communications, **E85-B** (1) 79-88 (2002).
[9] J. P. Yang, Performance analysis of threshold-based selective drop mechanism for high performance packet switches, Performance Evaluation, **57** (2) 89-103 (2004).
[10] S. X. Wei, E. J. Coyle, M.-T. T. Hsiao, An optimal buffer management policy for high-performance packet switching, Proceedings of the GLOBECOM, 924-928 (1991).
[11] I. Cidon, L. Georgiadis, R. Guerin, Optimal buffer sharing, IEEE Journal on Selected Areas in Communications, **13** (7) 1229-1239 (1995).
[12] S. Sharma, Y. Viniotis, Optimal buffer management policies for shared-buffer ATM switches, IEEE/ACM Transactions on Networking, **7** (4) 575-587 (1999).
[13] H. Kroner, G. Hebuterne, P. Boyer, A. Gravey, Priority management in ATM switching nodes, IEEE Journal on Selected Areas in Communications, **9** (3) 418-427 (1991).
[14] A. K. Choudhur, E. L. Hahne, Space priority management in a shared memory ATM switch, Proceedings of the GLOBECOM, 1375-1383 (1993).
[15] C. G. Kang, H. H. Tan, Queueing analysis of explicit policy assignment push-out buffer sharing schemes for ATM networks, Proceedings of the INFOCOM, 500-509 (1994).
[16] R. Roy, S. S. Panwar, Efficient buffer sharing in shared memory ATM systems with space priority traffic, IEEE Communications Letters, **6,** (4) 162-164 (2002).
[17] V. Zaborovsky, O. Zayats, V. Mulukha, Priority queueing with finite buffer size and randomized push-out mechanism, Proceedings of the ICN, 316-320 (2010).
[18] Y. S. Lin, C. B. Shung, Quasi-pushout cell discarding, IEEE Communications Letters, **1**, (5) 146-148 (1997).
[19] M. Shreedhar and G. Varghese, Efficient fair queuing using deficit round-robin, IEEE/ACM Transactions on Networking, **4** (3) 375-385 (1996).
[20] S. Ramabhadran, J. Pasquale, The stratified round robin scheduler: design, analysis and implementation, IEEE/ACM Transactions on Networking, **14** (6), 1362-1373 (2006).
[21] S. C. Tsao, Y. D. Lin, Pre-order deficit round robin: a new scheduling algorithm for packet-switched networks, Computer Networks, **35** (2-3) 287-305 (2001).
[22] S. Floyd, V. Jacobson, Random early detection gateways for congestion avoidance, IEEE/ACM Transactions on Networking, **1** (4), 397-413 (1993).
[23] C. Zhiruo, W. Zheng, E. Zegura, Rainbow fair queueing: fair bandwidth sharing without each-flow state, Proceedings of the INFOCOM, 922-931 (2000).
[24] I. Stoica, S. Shenker, H. Zhang, Core-stateless fair queueing: A scalable architecture to approximate fair bandwidth allocations in high-speed networks, IEEE/ACM Transactions on Networking, **11** (1) 33-46 (2003).
[25] Kai Mao , A Logarithmic Slow-Start Algorithm of TCP Vegas in IP Networks, Applied Mathematics and Information Sciences, **7** (2) 599-605 (2013).

Jui-Pin Yang was born in Kaohsiung, Taiwan, R.O.C., in 1972. He received the Ph.D. degree in department of electrical engineering at National Chung Cheng University in 2003. From 2004 to 2008, he served as an engineer and project leader in Industrial Technology Research Institute (ITRI). He is currently an associate professor with the department of information technology and communication, Shih Chien University, Taiwan. From 2010 to 2014, he was elected as a special outstanding researcher from National Science Council, Taiwan. His research interests include computer network, information system design and development and cloud technology.