PAPER *Special Issue on Internet Technology III*

# Cell-based Schedulers with Dual-rate Grouping*

**Dong WEI**[†], ***Student Member,*** **Jie YANG**[†], ***Nonmember,***
**Nirwan ANSARI**[†], ***Regular Member, and*** **Symeon PAPAVASSILIOU**[†], ***Nonmember***

**SUMMARY** The use of fluid Generalized Processor Sharing (GPS) algorithm for integrated service networks has received much attention since early 1990's because of its desirable properties in terms of delay bound and service fairness. Many Packet Fair Queuing (PFQ) algorithms have been developed to approximate GPS. However, owing to the implementation complexity, it is difficult to support a large number of sessions with diverse service rates while maintaining the GPS properties. The grouping architecture has been proposed to dramatically reduce the implementation complexity. However, the grouping architecture can only support a fixed number of service rates, thus causing the problems of granularity, bandwidth fairness, utilization, and immunity of flows. In this paper, we propose a new implementation approach called dual-rate grouping, which can significantly alleviate the above problems. Compared with the grouping architecture, the proposed approach possesses better performance in terms of approximating per session-based PFQ algorithms without increasing the implementation complexity.

***key words:*** *scheduling, high-speed, service rate, granularity*

## 1. Introduction

High-speed, service-integrated packet switches are required to support a large number of sessions with diverse service rate requirements. Statistical multiplexing is employed to improve the throughput of a switch. When multiplexed at the same output of a scheduler, different sessions interact with each other, and therefore scheduling algorithms are used to control the interactions among them.

Based on an idealized fluid model, A.K. Parekh [2] proposed the Generalized Processor Sharing (GPS) algorithm, which possesses three desirable properties: 1) it can guarantee the latency bound to any leaky-bucket-constrained session; 2) it can ensure fair allocation of bandwidth among all backlogged sessions; 3) it has a certain capability of immunity, i.e., it can isolate well-behaving sessions from disadvantageous effects of other misbehaving sessions. However, GPS is an idealized model and cannot be implemented in real world. Some service disciplines generally called Packet Fair Queuing (PFQ) algorithms, which differ in tradeoffs between implementation complexity and performance in terms of latency bound and service fairness, have been proposed to approximate GPS. In reality, due to the complexity, it is difficult to implement these disciplines in a scheduler to support a large number of sessions with diverse service rate requirements while maintaining all desirable GPS properties.

Implementation complexity of PFQ algorithms is determined by the following factors [1]: 1) the calculation of the system virtual time, which indicates the amount of normalized service that should be received by each session; 2) sorting the service order of all sessions; 3) the management of another priority queue to regulate packets (only if those algorithms with the "smallest eligible virtual finish time first," such as WF$^2$Q [8] or WF$^2$Q+ [6], are adopted as the service discipline). PGPS [2] and Weighted Fair Queuing [3] use the virtual system time defined by the fluid GPS model. Both need to track all backlogged sessions, and hence the worst case complexity is $O(N)$, where $N$ is the number of sessions. Some other PFQ algorithms, with virtual system time complexity $O(1)$ [4] and $O(\log N)$ [5][6], have been developed. The sorting complexity of most algorithms is $O(\log N)$. S. Suri, et al. [7], proposed to use the van Emde Boas data structure, which has the complexity of $O(\log\log N)$. H. Zhang, et al., [6][8] proposed a selection policy by selecting packets among all eligible sessions. This selection policy can improve the worst-case delay for clearing the backlog of a session's queue, but it requires extra management of another priority queue.

A novel grouping architecture, which can dramatically reduce the overall complexity, has been proposed in [1]. All sessions with the same service rate stay in the same group when they are active. However, this architecture has a restriction that only a fixed number of service rates can be supported.

The above restriction leads to the problem of service rate granularity. Such a problem may degrade the fairness of bandwidth allocation among different sessions. We observe that if bandwidth is not allocated fairly, even though the scheduling disciplines have integrated traffic regulation capability [6][8], their immunity capability to protect any session from other sessions' negative impact will still be degraded. Based on the fact that the service rate is essentially the amount of service received in a time interval, in order to achieve fair bandwidth allocation in the time interval, we propose to provide different service rates to a session alternately. With this approach, the number of service rate groups in the scheduler is not increased, and thus the implementation complexity remains the same as that of the

Manuscript received June 13, 2002.
Manuscript revised August 12, 2002.

[†]The authors are with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, 07102, USA.

[*]This work has been supported in part by the New Jersey Commission on Higher Education via the NJI-TOWER project, and the New Jersey Commission on Science and Technology via the NJCWT.

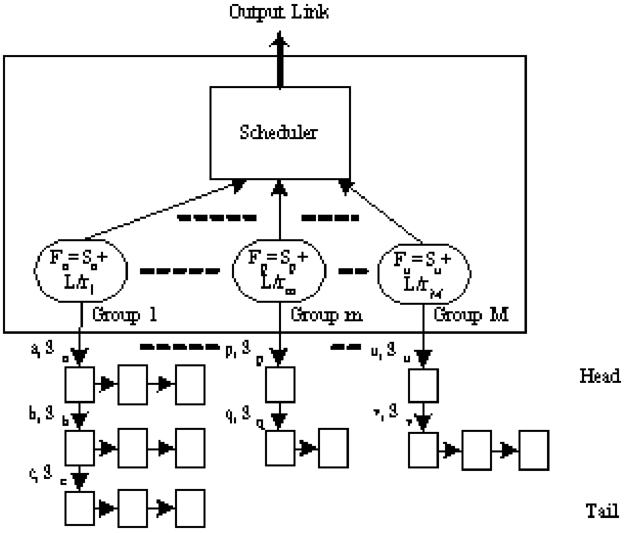


**Fig. 1** A cell-based scheduler with the grouping architecture.

grouping architecture. We implement such an approach by a scheme called dual-rate grouping to enable one session to switch between two different service rate groups. In such a scheme, not only the rate granularity but also the fairness of bandwidth allocation and the immunity capability can be improved. We also show that the dual-rate grouping is superior to the original grouping architecture in terms of approximating per session-based PFQ algorithms.

The rest of the paper is organized as follows. In Sect. 2, we review PFQ algorithms and the grouping architecture. Some important issues, which arise in the grouping architecture, are presented. In Sect. 3, we describe the proposed dual-rate grouping scheme and its implementation. Experimental results presented in Sect. 4 show that performance is significantly improved with our proposed approach. Finally, concluding remarks are included in Sect. 5.

## 2. PFQs and the grouping architecture

PFQ algorithms have a global variable - virtual system time $V(\cdot)$, which is defined differently for different PFQ algorithms. They also maintain a virtual start time and a virtual finish time for each session. When the ***k-th*** packet of session ***i*** arrives, the virtual start time $S_i(\cdot)$ and virtual finish time $F_i(\cdot)$ of this packet are given as follows:

$$S_i(t) = \begin{cases} \max\{V(t), F_i(t-)\} & \textit{session} \quad i \\ & \textit{becomes} \\ & \textit{active} \\ F_i(t-) & p_i^{k-1} \\ & \textit{finished} \\ & \textit{service} \end{cases} \tag{1}$$

$$F_i(t) = S_i(t) + \frac{L_i^k}{r_i} \tag{2}$$

where $L_i^k$ is the packet size of the ***k-th*** packet of session ***i***, and $r_i$ is the required service rate of session ***i***.

The virtual system time is updated when a packet starts to receive service [4] or new sessions become active. All PFQ algorithms have similar sorted-queue architecture; they differ in two aspects: 1) the virtual system time function, i.e., how to update the virtual system time; 2) packet selection policy.

WF$^2$Q [8] and WF$^2$Q+ [6] algorithms are the optimal PFQ algorithms in terms of accuracy in approximating GPS [2]. Both of them guarantee the smallest latency bound and Worst-case Fairness Index (WFI) among all PFQ algorithms. WF$^2$Q+ is superior to WF$^2$Q in terms of implementation complexity. Therefore, WF$^2$Q+ algorithm is adopted to conduct the performance analysis and simulations in this paper.

A grouping architecture for cell-based schedulers, as shown in Fig. 1, was proposed to efficiently implement PFQ algorithms in high-speed cell-based switches [1]. By employing the Locally Bounded Timestamp (LBT) property [1], the priority relationship among sessions in the same service group can be maintained without sorting. All sessions with the same service rate requirements are placed in the same group. The operation of the system can be summarized as follows:

- When a new session *p* is set up, it is assigned to a service rate group according to its rate requirement. The service rate of the group must be no less than the requirement of session *p*. At this moment, the first packet of session *p* is placed at the tail of its service group.
- The scheduler selects the packet with the smallest virtual finish time to transmit among all sessions in the heads of service rate groups.
- After a session receives service, if it is still backlogged, it is placed at the tail of the service rate group; if the session is temporarily idle or finished, it is taken out of the service rate group. The next session in the same group is placed at the head.
- When a session becomes active again, it can be treated as a new session and placed at the tail of the corresponding group.

In each group, each backlogged session is shifted one by one to the head of the group, and thus the session with the smallest virtual start time in each group is always at the head. Scheduling is performed only among sessions at the head of each group. Therefore, with this grouping architecture, the complexity of scheduling and updating of the virtual system time is reduced. For example, if WF$^2$Q+ is employed, the implementation complexity is reduced from $O(\log N)$ to $O(\log M)$, where *N* is the number of sessions and *M* is the number of service rate groups. In other words, the complexity of scheduling and updating of the virtual system time is decoupled from the number of sessions. Another key advantage of the grouping architecture is that it is able to perform per session-based traffic regulation (by the virtual start time) and traffic scheduling (by the virtual finish time) in an integrated manner, hence reducing the overall

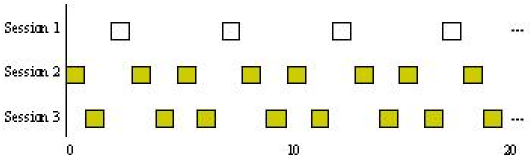


**Fig. 2** The service order of the scheduler with the grouping architecture.

worst-case complexity.

In the grouping architecture, a restriction is introduced: the scheduler can only support a fixed number of service rates at any time. In the following, consider an architecture that supports *M* service rate groups, and assume, without loss of generality, that $r_{g_1} < r_{g_2} < \cdots < r_{g_m} < \cdots < r_{g_M}$, where $r_{g_m}$ is the service rate of the *m-th* group. The following issues may arise in the grouping architecture:

**Granularity issue:** if a new session $i$, with service rate $r_i$, such that $r_{g_{m-1}} < r_i \leq r_{g_m}$, is required to be set up, the scheduler either declines the request, or may over-provision it with rate $r_{g_m}$.

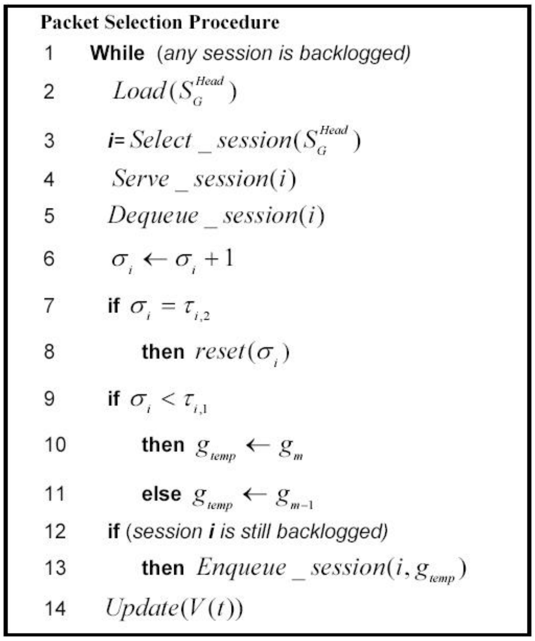
**Bandwidth fairness issue:** in the grouping architecture, a group service rate corresponds to a weight in the GPS model. At any time, the bandwidth is allocated according to the weights of all backlogged sessions. Therefore, this may cause the issue of unfair bandwidth allocation by providing session $i$ with service rate $r_{g_m}$.

For example, assume that a scheduler with the grouping architecture can provide service rate 1, $2^{-1}$, ..... $2^{-M+1}$, with the normalized output link capacity equal to 1. The scheduler provides service to three sessions from time 0. The required service rates are 0.25, 0.375 and 0.375, respectively. Owing to the service rate granularity, to meet the service requirements, sessions 2 and 3 are served in the group with service rate 0.5. Assume that all three sessions are continuously backlogged from time 0. The service order of packets is shown in Fig. 2 (assuming WF$^2$Q+ is the service discipline). Clearly, the service order is periodic with period of 5 when all three sessions are continuously backlogged in the interval (0,t). Thus, the bandwidth each session receives, during (0,t), is 0.2, 0.4 and 0.4, respectively. Therefore, session 1 receives less service than requested while sessions 2 and 3 receive more than requested.

**Immunity issue:** with heavy load, even though all sessions are well shaped and regulated, some sessions could be adversely affected by other sessions. For instance, in the previous example, session 1 is affected by sessions 2 and 3, and therefore its required service rate cannot be guaranteed during (0,t). Although WF$^2$Q and WF$^2$Q+ have integrated regulation function, the regulation performance may be degraded in the grouping architecture due to granularity.

## 3. The Proposed Dual-rate Grouping Scheme

In order to alleviate problems induced by the service rate granularity while maintaining the simplicity of the grouping architecture, we propose a scheme to improve the service rate granularity without compromising the implementation complexity. The main principle of our proposed scheme is to provide different service rates to a session alternately while ensuring more accurate throughput in this time interval. Based on this idea, the following new dual-rate grouping scheme is developed.



**Packet Selection Procedure**

```
While (any session is backlogged)
    $Load(S_G^{Head})$
    i= $Select\_session(S_G^{Head})$
    $Serve\_session(i)$
    $Dequeue\_session(i)$
    $\sigma_i \leftarrow \sigma_i + 1$
    if $\sigma_i = \tau_{i,2}$
        then $reset(\sigma_i)$
    if $\sigma_i < \tau_{i,1}$
        then $g_{temp} \leftarrow g_m$
        else $g_{temp} \leftarrow g_{m-1}$
    if (session i is still backlogged)
        then $Enqueue\_session(i, g_{temp})$
    $Update(V(t))$
```

**Fig. 3** Scheduling algorithm in dual-rate grouping.

Consider session $i$ with the required service rate $r_i$, where $r_{g_{m-1}} < r_i \leq r_{g_m}$, and assume that, in per session-based PFQ algorithm, totally $K$ packets in session i receive service in time interval $(t_1, t_2)$, and all packets have the same size $L$. Then, the service that session i receives in time interval $(t_1, t_2)$ is $KL$. By pumping some packets into service group $g_m$ and the rest to $g_{m-1}$, we try to achieve the same amount of service in time interval $(t_1, t_2)$. If we denote by $\alpha_i$ the portion of packets allocated to service group $g_m$ and $1 - \alpha_i$ portion to service group $g_{m-1}$, the following equation must hold:

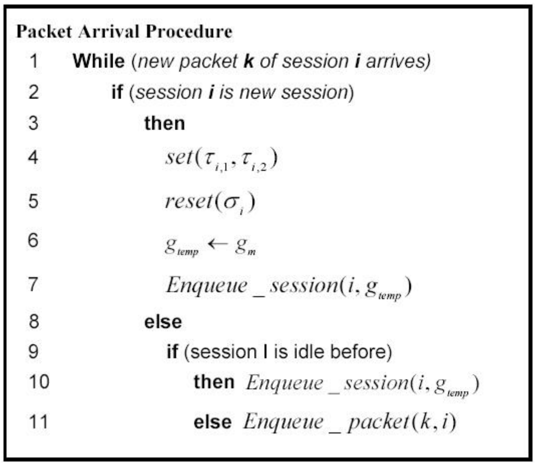
$$\begin{aligned} r_i &= \frac{KL}{t_2 - t_1} = \frac{KL}{\alpha_i KL r_{g_m}^{-1} + (1-\alpha_i) KL r_{g_{m-1}}^{-1}} \\ &= \frac{1}{\alpha_i r_{g_m}^{-1} + (1-\alpha_i) r_{g_{m-1}}^{-1}} \end{aligned}$$

Thus, $\alpha_i$ can be computed as follows:

$$\alpha_i = \frac{1 - r_{g_{m-1}} r_i^{-1}}{1 - r_{g_{m-1}} r_{g_m}^{-1}} \tag{3}$$

Therefore, with Eq. (3), the scheduler can place packets into different service rate groups accordingly to achieve a desirable bandwidth.

For session i, three components are introduced to implement the dual-rate grouping scheme: 1) a counter, $\sigma_i$, is used to record how many packets are transmitted in a periodic manner; 2) a marker, $\tau_{i,1}$, is used to indicate into which service group the session should be inserted; 3) another marker, $\tau_{i,2}$, is employed to indicate when the counter



```
Packet Arrival Procedure
1   While (new packet k of session i arrives)
2       if (session i is new session)
3           then
4               $set(\tau_{i,1}, \tau_{i,2})$
5               $reset(\sigma_i)$
6               $g_{temp} \leftarrow g_m$
7               $Enqueue\_session(i, g_{temp})$
8           else
9               if (session I is idle before)
10                  then $Enqueue\_session(i, g_{temp})$
11                  else $Enqueue\_packet(k, i)$
```

**Fig. 4** Operation on packet arrival in dual-rate grouping.

should be reset. Thus,

$$\alpha_i = \frac{\tau_{i,1}}{\tau_{i,2}} \tag{4}$$

Figure 3 shows the pseudo-code of the scheduling algorithm, while Fig. 4 presents the operation on new packet arrivals using the dual-rate grouping. Steps 6-11 in Fig. 3 and steps 4-6 in Fig. 4 are additional operations compared with the original grouping architecture. Note that these extra operations do not increase the complexity of the PFQ algorithm and memory access in implementation. $V(t)$ can be updated as $V(t+\tau) = \max\{V(t)+\tau, \min_{i\in B(t+\tau)} S_i(t+\tau)\}$, if WF$^2$Q+ is adopted.

The process of *Enqueue_session(i,$g_{temp}$)* (in Fig. 4) places session $i$ at the tail of the service group $g_{temp}$, and *Dequeue_session(i)*takes session $i$ out of the head of its current service group.

In order to maintain the LBT property, the following cases need to be considered in updating the virtual start time of each session:

- Session ***i*** is backlogged, and there is no need to change to another service rate group. Then, the virtual start time $S_i(t) = F_i(t-)$, where $F_i(t-)$ is the virtual finish time of the previous cell in the same session.
- Session ***i*** is active again from the idle status, and there is no need to change to another service rate group. Then, the virtual start time $S_i(t) = \max\{S^{Tail}_{g_{temp}}(t), V(t)\}$, where $S^{Tail}_{g_{temp}}(t)$ is the virtual start time of the session at the tail of the same service rate group.
- Session ***i*** is backlogged and needs to change to another service rate group. Then, the virtual start time $S_i(t) = \max\{S^{Tail}_{g_{temp}}(t), \min\{S^{Head}_{g_{temp}}(t) + \frac{L}{r_{g_{temp}}}, F_i(t-)\}\}$ where $g_{temp}$ is the service rate group in which the session is inserted and $S^{Head}_{g_{temp}}(t)$ is the virtual start time of the session at the head of this service rate group.
- Session ***i*** becomes active again from the idle status and needs to change to another service rate group. Then, the virtual start time $S_i(t) =$

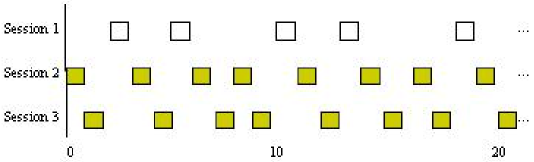


**Fig. 5** Ideal Service Order.

$$\max\{S^{Tail}_{g_{temp}}(t), \min\{S^{Head}_{g_{temp}}(t) + \frac{L}{r_{g_{temp}}}, V(t)\}\}$$

## 4. Experimental results and discussion

In this section, we provide some numerical results to evaluate the performance of our proposed scheme and compare it with the corresponding performance of the grouping architecture. To achieve this, two sets of experiments are performed. In Experiment 1, we demonstrate that, using the proposed approach, fairer bandwidth allocation can be achieved and the performance in terms of latency approximates per session-based PFQ better than the grouping architecture. In Experiment 2, we demonstrate that, if some sessions are not well shaped, the integrated regulation function of WF$^2$Q+ algorithm can be degraded, and the proposed approach can alleviate the negative impacts of ill behaving sessions.

Consider the same example described in Sect. 2, then the ideal service order is shown in Fig. 5.

We have implemented WF$^2$Q+ with per session-based queuing, WF$^2$Q+ with the grouping architecture, and our proposed scheme by OPNET. Throughout our evaluation process, we assume that the output link capacity is $R = 8000$ cells/second. Three sessions are in service, and the size of the token buffer is 1024 cells. Since WF$^2$Q+ has the best performance in terms of approximating GPS, WF$^2$Q+ scheme with per session-based queuing is used as the reference, i.e., the ideal case.

Note that, in general, $r_{g_{m+1}}/r_{g_m}$ can be any value larger than 1. This ratio is selected equal to 2, in our experiments, for simplicity in representation and implementation consideration.

### 4.1 Experiment 1

In the following, let $r_1$=2000 cells/second, which is 0.25 of the total output capacity, and $r_2 = r_3$=3000 cells/second, which is 0.375 of the total output capacity. Moreover, assume that each session is shaped by a leaky-bucket, and the session's token rate is the same as its required service rate.

Based on Eqs. (3) and (4), we obtain $\alpha_1 = 1$, $\tau_{1,1} = \tau_{1,2} = 1$;$\alpha_2 = \alpha_3 = 2/3$, $\tau_{2,1} = \tau_{3,1} = 2$, and $\tau_{2,2} = \tau_{3,2} = 3$. With these parameters and the dual-rate grouping architecture, when all sessions are continuously backlogged, the service order of all sessions is identical to that with per-session based WF$^2$Q+ scheme as shown in Fig. 5.

When all sessions are backlogged (for example in time interval (5s, 10s)), the normalized bandwidths allocated to

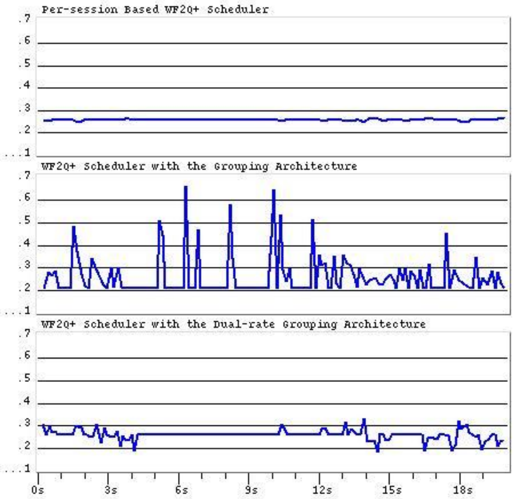


**Fig. 6** Bandwidth allocation of session 1 in Experiment 1.

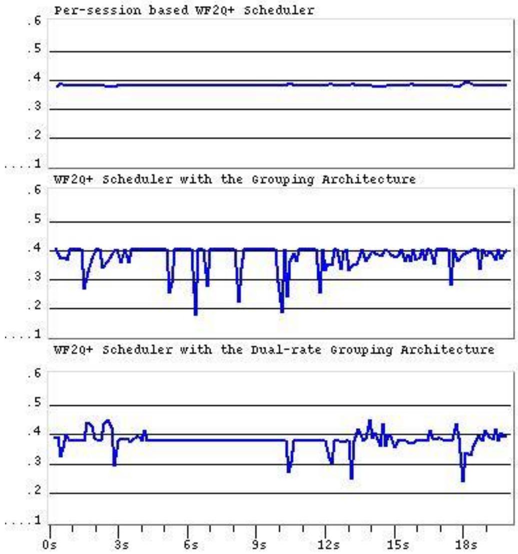


**Fig. 7** Bandwidth allocation of session 2 in Experiment 1.

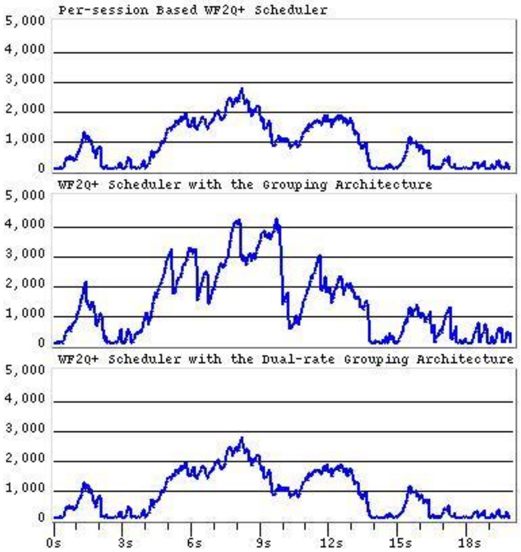


**Fig. 8** Delay of session 1 in Experiment 1.

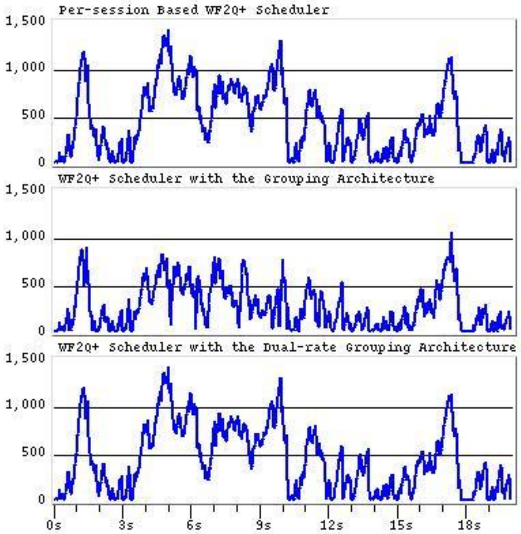


**Fig. 9** Delay of session 2 in Experiment 1.

each session should be 0.25, 0.375, and 0.375, respectively, as shown in Figs. 6 and 7 for session 1 and 2 (since traffic characteristics of sessions 2 and 3 are identical, we only show session 2 here), because the normalized weights of the three sessions are 0.25, 0.375 and 0.375, respectively. With the grouping architecture, as shown in Figs. 6 and 7, the received bandwidths should be 0.20 and 0.40, respectively, because the weights of the sessions are 0.25, 0.50 and 0.50.

Note that the arrival rates of sessions 1 and 2 are 2000 cells/second and 3000 cells/second, respectively, and the allocated bandwidths are 1600 cells/second and 3200 cells/second, respectively; in other words, session 2 is over-provisioned, while session 1 is under-provisioned. Thus session 2 is emptied frequently, and session 1 can receive more bandwidth when the queue of session 2 is emptied. Therefore, the allocated bandwidth to each session oscillates. With dual-rate grouping, bandwidths allocated to each session approximate the ideal case better than the grouping architecture since session 2 and 3 are getting two weights of 0.25 and 0.50 alternately (because they are placed into two different service groups alternately), and the average weight is 0.375. The oscillations as well as its amplitude have been reduced. When both sessions are backlogged (for example in time interval (5s, 10s)), the allocated bandwidths of sessions 1 and 2 coincide with the ideal case of per-session based WF$^2$Q+.

As shown in Figs. 8 and 9, the dual-rate grouping also approximates per session-based WF$^2$Q+ better than the grouping architecture in terms of delay. Especially when all sessions are backlogged (in the interval (5s, 10s)), the curve of delay of session 1 and 2 are identical with that of per-

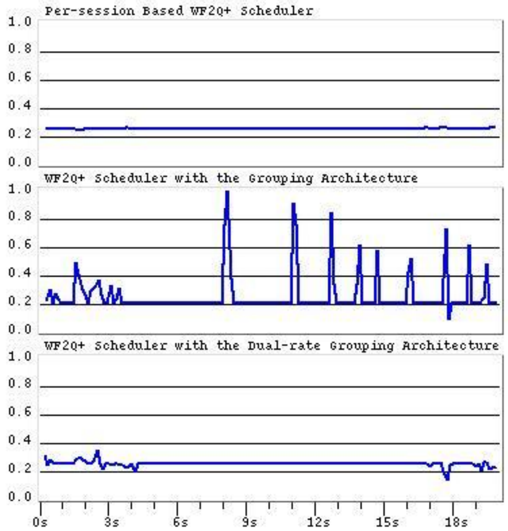


**Fig. 10** Bandwidth allocation of session 1 in Experiment 2.

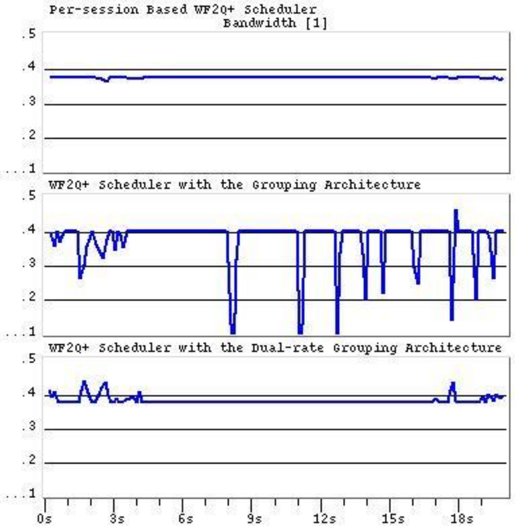


**Fig. 11** Bandwidth allocation of session 2 in Experiment 2.

session based WF$^2$Q+. This is attributed to the fact that when all sessions are backlogged, the dual-rate grouping scheme can allocate more accurate bandwidth to all sessions than the grouping architecture. Note that, in Fig. 9, the delay with the grouping architecture is smaller than that of per-session based scheduler, because session 2 receives more bandwidth than it should receive when it is backlogged. In both Figs. 8 and 9, the unit of delay is time slot.

## 4.2 Experiment 2

In reality, per session-based leaky-bucket shaping may not be implemented in high-speed schedulers due to implementation complexity. In this experiment, we assume that session 2 is ill-behaving with arrival rate 4000 cells/second, although its required service rate is only 3000 cells/second.

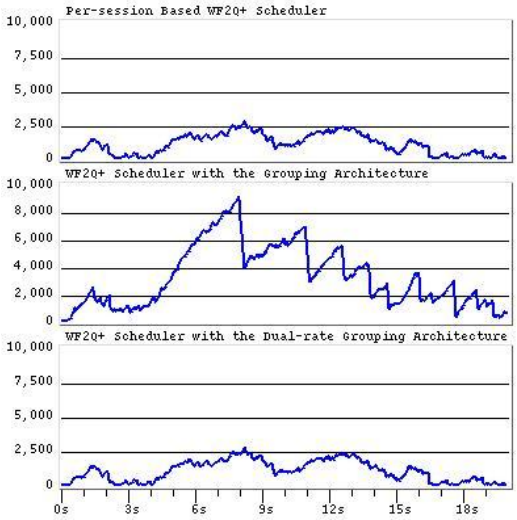


**Fig. 12** Delay of session 1 in Experiment 2.

Traffic characteristics of sessions 1 and 3 remain the same as in Experiment 1.

As shown in Figs. 10 and 11, with per-session based WF2Q+ scheme, the allocated bandwidths for sessions 1and 2 are 0.25 and 0.375, respectively, implying that this scheme possesses the capability to protect well-behaving sessions from the adverse impact of ill-behaving sessions. With the grouping architecture, the allocated bandwidth of well-behaving session 1 is adversely affected by ill-behaving session 2. As shown in Fig. 12, the delay of session 1 is also affected (session 3, not shown here, is also similarly affected). The allocated bandwidth and delay of session 1, with our dual-rate grouping scheme, approximate that of the per-session based scheme very well. With the grouping architecture, as shown in Fig. 11, the ill-behaving session 2 receives more bandwidth (0.4) than it should (0.375), when it is backlogged in time interval (4s, 8s). With the dual-rate grouping scheme, it receives 0.375, which is session 2's required bandwidth. Therefore, with our proposed scheme, the delay bounds of sessions can be guaranteed to be the same as in per-session based WF$^2$Q+ scheme, except for those ill-behaving sessions. Per-session based WF$^2$Q+ scheme possesses the ability to protect a well-behaving session from being affected by ill-behaving sessions and the ability to regulate those ill-behaving sessions. This immunity capability can be degraded when the grouping architecture is employed. We demonstrate, in this experiment, that our scheme can alleviate the adverse affects of those ill-behaving sessions, and hence significantly improve the immunity capacity.

## 4.3 Discussions

Although our proposed scheme can approximate the per-session based scheme better, the service order may not be exactly the same as that of the per-session based scheme in some cases. Thus, in this subsection, we discuss some issues associated with our proposed dual-rate grouping scheme.

### 4.3.1 Approximation of $\alpha_i$

In the previous experiments, we selected $\tau_{i,1}$ and $\tau_{i,2}$ to represent $\alpha_i$ exactly. However, due to the limitation of the number of bits to represent $\tau_{i,1}$, $\tau_{i,2}$ and counter $\sigma_i$, sometimes we need to approximate $\alpha_i$ with $\tau_{i,1}$ and $\tau_{i,2}$. Therefore, for implementation purpose, we propose to select $\tau_{i,1}$ and $\tau_{i,2}$, such that

$$\frac{\tau_{i,1}-1}{\tau_{i,2}} < \alpha_i \leq \frac{\tau_{i,1}}{\tau_{i,2}} \tag{5}$$

Let

$$\hat{\alpha}_i = \frac{\tau_{i,1}}{\tau_{i,2}} \tag{6}$$

Thus $\hat{\alpha}_i$ is selected to approximate $\alpha_i$.

### 4.3.2 Admission control issue

In [1], to guarantee that the service rate of each session is satisfied, the following inequality must hold:

$$\sum_{i=1}^{N} r_i \leq R \tag{7}$$

In our proposed scheme, each session may receive more bandwidth due to the approximation of $\alpha_i$. Let

$$\hat{r}_i = \frac{1}{\hat{\alpha}_i r_{g_m}^{-1} + (1-\hat{\alpha}_i) r_{g_{m-1}}^{-1}} \tag{8}$$

By Eqs. (5) and (6), $\alpha_i \leq \hat{\alpha}_i$, then with Eq. (8), we can obtain $r_i \leq \hat{r}_i$.

Thus, when we employ the following policy instead of Eq. (7) in the admission control scheme, each session's required service rate can be guaranteed.

$$\sum_{i=1}^{N} \hat{r}_i \leq R \tag{9}$$

### 4.3.3 Delay and delay bound

Compared with the per-session based scheme, the delay bound of each session will not be increased because the actual service rate is no less than that it requires, when Eq. (9) is employed in the admission control scheme. To compare with the per-session based WF$^2$Q+ scheme, a single packet could have an extra delay in the following cases: for instance, in the above example, 1) when sessions 2 and 3 receive service in the group with the service rate of 4000 cells/second and session 1 receives service in the group with the service rate of 2000 cells/second, session 1 receives less bandwidth than it requires, because session 2 and 3 receive more bandwidth than required, and thus some packets in session 1 will have an extra delay; 2) when sessions 1, 2 and 3 all receive service in the group with the service rate of 2000 cells/second, sessions 2 and 3 receive less bandwidth than they require, and thus some packets in sessions 2 and 3 will have more delay.

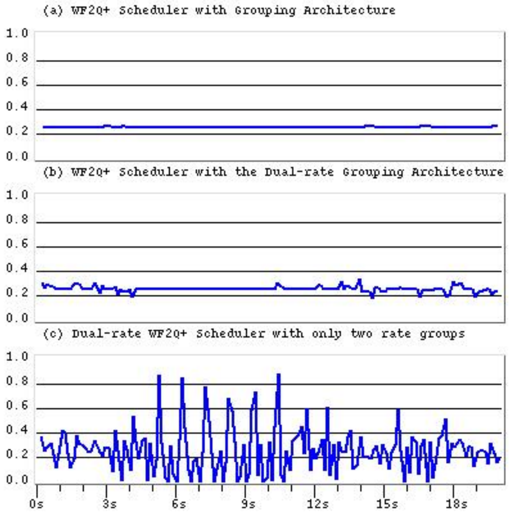


**Fig. 13** Bandwidth allocation of session 1 in Experiment 3.

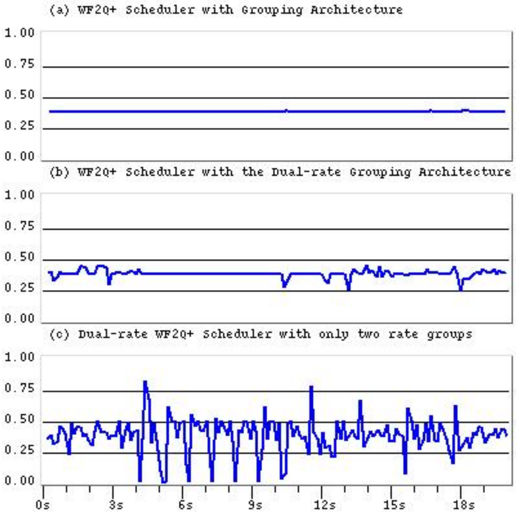


**Fig. 14** Bandwidth allocation of Session 2 in Experiment 3.

### 4.3.4 The number of groups of service rates

It should be noted that the more the number of service groups, i.e., the larger *M*, the better performance in terms of approximating per-session based scheduler. The reason is that, with more service groups, more service rates can be supported, and thus finer service rate granularity can be achieved, and over-provisioned and under-provisioned bandwidth can be reduced. Furthermore, we note that it is possible to use only two service groups, one with the maximum service rate and the other with the minimum service rate, to achieve any required service rate by pumping cells in these groups alternately. This is an extreme case of the dual-rate grouping architecture, with *M*=2. We compare the performance of such a scheduler and a scheduler with more service rate groups in the following experiment.

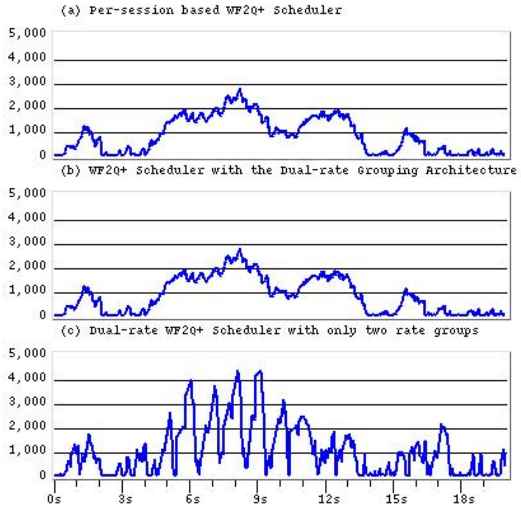


**Fig. 15** Delay of session 1 in Experiment 3.

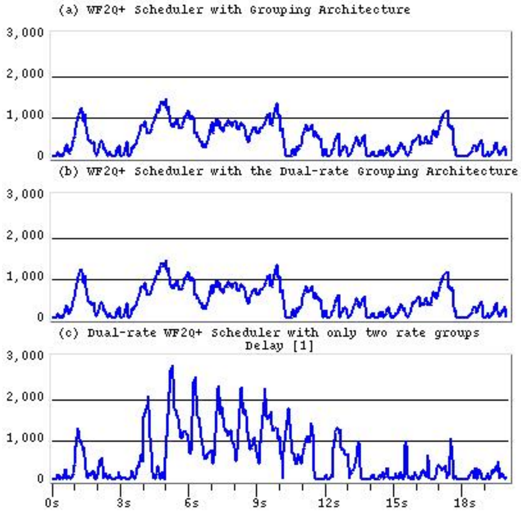


**Fig. 16** Delay of session 2 in Experiment 3.

Experiment 3: $R$=8000 cells/s, $r_1$=2000 cells/s, $r_2 = r_3$=3000 cells/s. Scheduler 1: per session based WF2Q+ scheduler; Scheduler 2: scheduler with the dual-rate grouping architecture, minimum service rate is 7.8125 cells/s, maximum service rate is 8000 cells/s. There are 11 service rate groups. Session 1 always stays in group 9 (the service rate is 2000 cells/s), and session 2 and 3 are switched between group 9 and 10 (the service rate of group 10 is 4000 cells/s); Scheduler 3: scheduler with the dual-rate grouping architecture, however, there are only two service rate groups, i.e., the minimum rate 7.8125 cells/s and maximum rate 8000 cells/s.

As shown in Figs. 13 and 14, the bandwidth allocated to session 1 and 2 oscillate a lot by using scheduler 3. The reason is that when sessions are put into the maximum rate group, their bandwidths are all over-provisioned much more than they require, and when they are put into the minimum rate group, their bandwidths are under-provisioned much less than they require. As a consequence, the maximum delay of each session becomes larger, as shown in Figs. 15 and 16. Therefore, more service groups can provide better performance even with our dual-rate grouping architecture.

## 5. Conclusions

In this paper, we have proposed a new dual-rate grouping strategy in order to alleviate the problems of granularity associated with the original grouping architecture. The performance evaluation study has demonstrated that the proposed scheme can approximate the PFQs better than the original grouping architecture in terms of bandwidth allocation, immunity capability, and delay. One of the most important advantages of our proposed scheme is that its implementation complexity remains the same as the grouping architecture.

### References

[1] D.C. Stephens, J.C.R. Bennett and H. Zhang, "Implementing scheduling algorithms in high-speed networks," IEEE Journal on Selected Areas in Communications, vol.17, no.6, pp. 1145 -1158, June 1999.

[2] A.K. Parekh, "A generalized processor sharing approach to flow control in integrated services networks," Ph.D thesis, MIT, 1992.

[3] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queuing algorithm," Internetworking: Research and Experience, vol.1, no.1, pp.3–26, 1990.

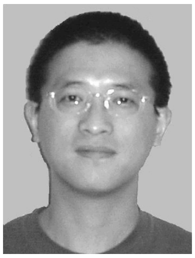
[4] S.J. Golestani, "A self-clocked fair queuing scheme for broadband applications," Proceedings of IEEE INFOCOM'94, pp.636–646, April 1994.

[5] D. Stiliadis and V. Varma, "Design and analysis of frame-based fair queuing: a new traffic scheduling algorithm for packet-switched networks," Proceedings of the ACM SIGMETRICS Conference on Measurement & Modeling of Computer Systems, pp. 104 - 115, 1996.

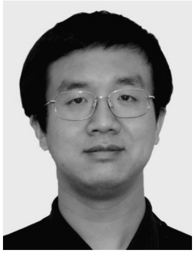
[6] J.C.R. Bennett and H. Zhang, "Hierarchical packet fair queuing algorithms," Proceedings of ACM SIGCOMM'96, pp.143–156, August 1996.

[7] S. Suri, G. Varghese and G. Chandranmenon, "Leap forward virtual clock," Proceedings of INFOCOM'97, vol.2, pp. 557 -565, April 1997.

[8] J.C.R. Bennett and H. Zhang, "WF$^2$Q: worst-case fair weighted fair queuing," Proceedings of IEEE INFOCOM'96, pp.120–128, March 1996.



**Dong Wei** received the B.S. and M.S. degrees in Electrical Engineering from Tsinghua University and New Jersey Institute of Technology, respectively. During 1995-1998, he worked as a system engineer and project manager in Siemens Ltd., China. He is currently a Ph.D candidate in the Department of Electrical and Computer Engineering, New Jersey Institute of Technology. His current research interests are high-speed networks, QoS and Internet security.



**Jie Yang** received the BS degree in information engineering and the MS degree in communication and information systems from Xidian University, China, in 1996 and 1999, respectively. He is currently a PhD candidate in electrical engineering in the Department of Electrical and Computer Engineering, New Jersey Institute of Technology. His current research interests are high-speed switch/router architectures, admission control, resource allocation and traffic engineering, and Internet security.

**Nirwan Ansari** received the B.S.E.E. (summa cum laude), M.S.E.E., and Ph.D. from the New Jersey Institute of Technology, University of Michigan, and Purdue University in 1982, 1983, and 1988, respectively. He joined the Department of Electrical and Computer Engineering at NJIT in 1988, and has been Professor since 1997. He is a technical editor of the IEEE Communications Magazine, was instrumental, while serving as its Chapter Chair, in rejuvenating the North Jersey Chapter of the IEEE Communications Society which received the 1996 Chapter of the Year Award, currently serves as the Chair of the IEEE North Jersey Section, and also serves in the IEEE Region 1 Board of Directors and various IEEE committees. He was the 1998 recipient of the NJIT Excellence Teaching Award in Graduate Instruction, and a 1999 IEEE Region 1 Award. His current research focuses on various aspects of high-speed networks including QoS routing, congestion control, traffic scheduling, video traffic modeling and delivery, and resource allocation. He authored with E.S.H. Hou Computational Intelligence for Optimization (1997, and translated into Chinese in 2000), and edited with B. Yuhas Neural Networks in Telecommunications (1994), both published by Kluwer Academic Publishers.

**Symeon Papavassiliou** received the Diploma in Electrical Engineering from the National Technical University of Athens, Greece, in 1990 and the M.Sc. and Ph.D. degrees in Electrical Engineering from Polytechnic University, Brooklyn, New York in 1992 and 1995 respectively. From 1995 to 1996 Dr. Papavassiliou was a Technical Staff Member at AT&T Bell Laboratories in Holmdel, New Jersey, and from 1996 to August 1999 he was a Senior Technical Staff Member at AT&T Laboratories in Middletown, New Jersey. From June 1996 till August 1999 he was also an Adjunct Professor at the Electrical Engineering Department of Polytechnic University, Brooklyn, NY. Since August 1999 he has been an Assistant Professor at the Electrical and Computer Engineering Department of New Jersey Institute of Technology, Newark, New Jersey. Dr. Papavassiliou was awarded the Best Paper Award in INFOCOM'94 and the "AT&T Division Recognition and Achievement Award" in 1997. His main research interests lie in the areas of computer and communication networks with emphasis on wireless communications and high-speed networks, network design and management, TCP/IP and internetworking, computer network modeling and performance evaluation and optimization of stochastic systems.