WF²Q-M: Worst-case fair weighted fair queueing with maximum rate control

Jeng Farn Lee [a,c,*], Meng Chang Chen [a], Yeali Sun [b]

[a] *Institute of Information Science, Academia Sinica Taiwan, Taipei 115, Taiwan, ROC*
[b] *Department of Information Management, National Taiwan University, Taiwan, ROC*
[c] *Department of Electrical Engineering, National Taiwan University, Taipei 106, Taiwan, ROC*

Abstract

While existing weighted fair scheduling schemes guarantee minimum bandwidths/resources for the classes/processes of a shared channel, the maximum rate control, which is critical to service providers, carriers, and network managers for resource management and business strategies in many applications, is generally enforced by employing traffic policing mechanisms. These approaches use either a concatenation of the rate controller and scheduler, or a policer in front of the scheduler. The concatenation method uses two sets of queues and a management apparatus that incurs overhead. The latter method may allow bursty traffic to pass through the controller, which violates the maximum rate constraint, or results in packet loss. In this paper, we present a new weighted fair scheduling scheme, WF²Q-M, which can simultaneously support maximum rate control and minimum service rate guarantees. WF²Q-M uses the virtual clock adjustment method to enforce maximum rate control and distribute the excess bandwidths of saturated sessions to other sessions without recalculating the virtual starting and finishing times of sessions. In terms of performance, we prove that WF²Q-M is theoretically bounded by a corresponding fluid reference model. A procedural scheduling implementation of WF²Q-M is proposed, and a proof of correctness is given. Finally, we present the results of extensive experiments to show that the performance of WF²Q-M is just as claimed.
© 2006 Elsevier B.V. All rights reserved.

Keywords: Scheduling; Rate control; Weighted fair; QoS

1. Introduction

Many critical Internet applications have strict performance requirements in terms of throughput, delay, delay jitter, and loss rate, or a combination of these items. Current best-effort service models cannot meet these requirements, as they handle all traffic equally and do not provide performance guarantees. A number of service disciplines have endeavored to provide per-connection or per-queue performance guarantees [1,2,5,7,9,16,24]. However, they do not provide the maximum rate constraint.

* Corresponding author. Tel.: +886 2 27883799x1614; fax: +886 2 26518660.
  E-mail address: kunimi@iis.sinica.edu.tw (J.F. Lee).

Maximum rate constraint (MRC) is an important management policy for service providers and many applications in which the maximum resource allocated to particular users are limited. Here are examples: (1) when resource is scarce, e.g., a wireless medium, or a shared link from an office, building, or campus to the Internet; (2) when there is a specific management policy, e.g., a restriction on throughput of certain traffics types, such as Web browsing, to a maximum rate; (3) to support a business model, e.g., rate constraints to support a pricing model; and (4) in multimedia streaming, especially when there are no feedback mechanisms from receivers; (5) support formulti-services and multi-user resource limited communication contexts (such as wireless communication [15]) so that QoS can be provided.

MRC is used in edge routers to shape outgoing and ingoing traffic into the desired traffic patterns and to provide QoS according to the service level agreements (SLAs). Additionally, it can be used by local carrier in their Internet Data Center (IDC). An IDC accommodates hundreds of servers attached to the carrier's backbone network via a number of high-speed LAN. Each server is owned by one customer, who chooses the connection speed. For instance, a big server may need a 100 Mbps connection, while a small one only needs 64 Kbps. But now as they are connected via high-speed LANs, it is critical for the carrier to regulate their traffic so that the servers cannot use more bandwidth than what they subscribe. Another situation is also popular that users in a building/campus sharing a link to the Internet require the MRC mechanism to protect normal usage from excessive and/or abusive usage of greedy users. Moreover, MRC can be used in both Diffserv and Inserv networks, depending on the definition of session to be per service class or per flow [13].

Therefore, many services need disciplines which are able to simultaneously provide minimum performance guarantees and enforce maximum service rate constraints. In this paper, we propose a new service discipline called WF^{2}Q-M (worst-case fair weighted fair queueing with maximum rate control), which provides the property of WF^{2}Q as well as maximum rate constraint enforcement. Compared with traditional approaches, WF^{2}Q-M is efficient in terms of buffer space, management complexity, and computation cost. A preliminary version of this work was presented in [12].

This paper is organized as follows. In Section 2, we review weighted fair scheduling schemes and

approaches for maximum rate constraint. In Section 3, we describe the proposed WF^{2}Q-M and its corresponding GPS-M model. In Section 4, we present the system properties of WF^{2}Q-M, and we show the performance of WF^{2}Q-M through simulations in Section 5. Conclusions are drawn in Section 6.

2. Related works

2.1. GPS, WFQ, and WF^{2}Q

In this section, we review GPS (generalized processor sharing) [16], and the popular packet approximation algorithms, WFQ (weighted fair queueing) [16] and WF^{2}Q (worst-case fair weighted fair queueing) [1]. GPS is a fluid system in which the traffic is infinitely divisible and all traffic streams can receive service simultaneously. Every session i of through traffic is assigned a positive real number ϕ_i indicating its weight in sharing the channel capacity. Let $W_{i,\text{GPS}}(t_1, t_2)$ be the amount of work received by session i in the time interval $[t_1, t_2]$; then a GPS server guarantees

$$\frac{W_{i,\text{GPS}}(t_1, t_2)}{W_{j,\text{GPS}}(t_1, t_2)} \geqslant \frac{\phi_i}{\phi_j} \quad i, j = 1, 2, \ldots, N \tag{1}$$

for any session i that is continuously backlogged throughout the interval $[t_1, t_2]$. A session is *backlogged* if it has packets waiting for transmission in its queue or if its packet is in service. Each backlogged session i receives the guaranteed service rate r_i:

$$r_i = \frac{\phi_i}{\sum \phi_j} C \tag{2}$$

in which C is the link capacity and $\sum \phi_j$ is the sum of the weights of all sessions.

Unlike the idealized fluid model, realistic packet systems serve only one session at a time, and the transmission unit is a packet. WFQ and WF^{2}Q are two popular packet approximation service disciplines of GPS. Let a_i^k be the arrival time of the kth packet of session i, and let d_i^k be its departure time under GPS. Both WFQ and WF^{2}Q select the packet with the smallest d_i^k as the next packet to be transmitted. The disciplines differ in that WF^{2}Q only considers the sets of packets that have started receiving service in the corresponding GPS system, whereas WFQ does not. Parekh and Gallager [16] showed that WFQ has the following properties:

$$d_{i,\text{WFQ}}^k - d_{i,\text{GPS}}^k \leqslant \frac{L_{\max}}{C}, \quad (3)$$

$$W_{i,\text{GPS}}(0, t) - W_{i,\text{WFQ}}(0, t) \leqslant L_{\max},$$

where $d_{i,\text{WFQ}}^k$ and $d_{i,\text{GPS}}^k$ are the departure times under WFQ and GPS, respectively; $W_{i,\text{WFQ}}(0, t)$ and $W_{i,\text{GPS}}(0, t)$ are the total amounts of service received by session i at time t from WFQ and GPS, respectively; $L_{\max}$ is the maximum packet length. These two properties show that WFQ achieves a service performance close to that of GPS. However, WFQ has a problem that it may serve far ahead from GPS system. In [1], Bennett and Zhang showed that WF^{2}Q has a property that provides a tight bound for this problem, as shown in the following formula:

$$W_{i,\text{WF}^2\text{Q}}(0, t) - W_{i,\text{GPS}}(0, t) \leqslant \left(1 - \frac{r_i}{C}\right) L_{i,\max}. \quad (4)$$

Both WFQ and WF^{2}Q are implemented based on the system virtual clock $V(t)$, which is the normalized amount of service that a backlogged session should receive at time t in the corresponding GPS system. $V(t)$ evolves as follows:

$$V(0) = 0,$$

$$V(t + \tau) = V(t) + \frac{\tau}{\sum_{i \in B(t)} \phi_i}, \quad (5)$$

where $B(t)$ is the set of backlogged sessions.

Each packet p_i^k (i.e., the kth packet in session i) is assigned a virtual starting S_i^k and finishing time F_i^k, defined as

$$S_i^k = \max\{F_i^{k-1}, V(a_i^k)\},$$

$$F_i^k = S_i^k + \frac{L_i^k}{\phi_i * C}. \quad (6)$$

As the virtual clock function is monotonically increasing, its inverse function exists and is denoted by $V^{-1}(t)$. Thus, $V^{-1}(S_i^k)$ and $V^{-1}(F_i^k)$ are the real

clock times when packet p_i^k starts and finishes service in the corresponding GPS system. While WFQ selects the packet with the earliest virtual finishing time, WF^{2}Q only considers packets whose virtual starting times are earlier than $V(t)$, and selects from among them packet p_i^k with the smallest F_i^k.

WF^{2}Q is an accurate approximation algorithm for GPS. However, it can only guarantee the minimum service rate for a session; it cannot constrain the maximum service rate. In this paper, we propose a new service discipline called WF^{2}Q-M, which is similar to WF^{2}Q in many ways, in that it has bounded-delay and fairness properties with respect to the corresponding fluid model. Besides guaranteeing the minimum rate, WF^{2}Q-M also provides the maximum service rate constraint.

2.2. Rate-controlled service disciplines

Several non-work-conserving disciplines have been proposed, including Jitter earliest-due-date (Jitter-EDD) [21], stop-and-go [8], hierarchical round robin (HRR) [15], and rate-controlled static priority (RCSP) [22]. They provide delay-jitter bounds, end-to-end delay bounds or rate control based on either a time-framing strategy or a sorted priority queue mechanism. In [23], Zhang and Ferrari showed that the general rate-controlled service discipline could represent all of the disciplines. As shown in Fig. 1, the general rate-controlled server has two stages: the rate controller and scheduler. The rate controller is responsible for shaping input traffic into desired traffic patterns, assigning an eligible time for each packet, and moving packets to the scheduler when they are eligible. The scheduler multiplexes eligible packets from all connections and determines the service sequence of the packets.

An important operation issue with the rate-controller is deciding when to move packets to the
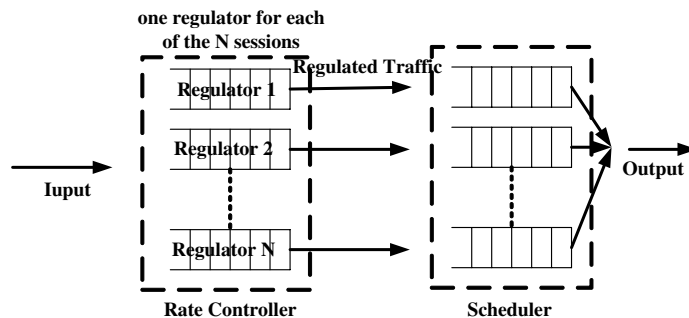

Fig. 1. Two-stage rate-control service discipline.

scheduler. The simplest method is to set a timer for the head packet of each queue; however, this introduces overhead, which is not acceptable for high-speed routers. Hardware implementation may help to reduce the overhead, but, this approach limits the number of timers and, therefore, the number of classes. Most solutions in the literature are based on time-framing, event-driven strategies, or both. There is a tradeoff between system accuracy and time granularity in the framing strategy. A smaller frame period results in more accurate bandwidth allocation and a higher operation cost, while a larger frame period leads to the opposite results. The event-driven strategy is based on the occurrence of driving events, e.g., packet enqueues or dequeues. As the timing of event occurrences is not predictable, high uncertainty is intrinsic in this approach. Our proposed service discipline alleviates the problem by eliminating the eligible time calculation and combining the rate controller and scheduler, thus greatly reducing the overhead.

Another popular rate-controlled model is the policer-based rate-control service model, as shown in Fig. 2, in which a token bucket or a leaky bucket [20] is used as the policer. When an incoming packet obtains enough tokens, it proceeds directly to the scheduler; otherwise, it is dropped. Note that if a policer is implemented with the packet buffer, we can consider it as a special case of the two-stage rate-control service discipline. While a token bucket policer can maintain the session's average rate, burst traffic with a rate exceeding the designated maximum rate may be transmitted. If the policer uses a leaky bucket, the maximum rate constraint can be strictly enforced, however bursty packets may be dropped, which resulting in less than expected throughput. Although they do not have the complexity problems as that in a two-stage rate-control service, their drawbacks prevent them from providing effective maximum rate control.
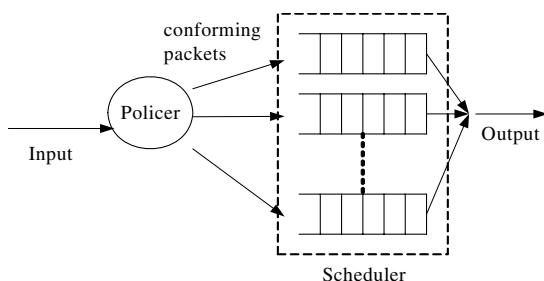
Here, we present simulation results obtained using ns-2 [14] to show that the policer-based rate-control service discipline is not suitable for maximum rate control. We compare the packet loss rates of our proposed method (WF^2Q-M), token bucket, leaky bucket, and a concatenation of token bucket and leaky bucket with different simulation settings. The scheduler of the Policer-based rate controller was WF^2Q. The data source generated UDP traffic based on the exponential ON/OFF model, where the average times of the ON and OFF periods were 312 ms and 325 ms, respectively. The data rate in an ON period was 4.25 Mbps, and the packet size was 1500 bytes. The token generation rate for both token bucket and leaky bucket was set to be the maximum rate constraint, 4 Mbps. The bucket depth of token bucket was set to be the same as the buffer size of WF^2Q-M, ranging from 0.05 Mbits to 1 Mbits.

The measured packet loss parameters include the packet loss rate and *over maximum rate percentage*, which is the percentage of output rate exceeding the designated maximum rate measured as the receiver side. We consider both to be illegitimate under maximum rate constraint. The simulation results show that WF^2Q-M can enforce the maximum rate such that its *over maximum rate percentage* is 0. In Fig. 3, the simulation results show that the packet loss rate of WF^2Q-M is almost identical to that of the token bucket scheme due to the reason that the buffer size of WF^2Q-M and the bucket depth of the token bucket are set as the same, but the token bucket scheme incurs a high *over maximum rate percentage*. The total pack loss of the token bucket scheme (i.e., the sum of the packet loss rate and over
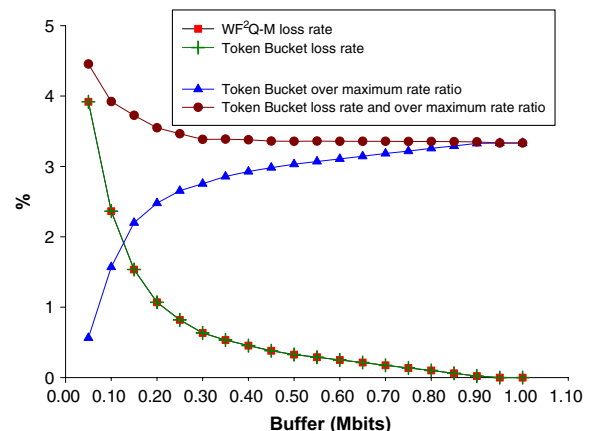

Fig. 2. Policer-based rate-control service discipline.

Fig. 3. Simulation results for the policer-based rate controller.

maximum rate ratio) is much higher than that of WF²Q-M, which prevents it from being a good maximum rate constraint server. The results for the case with a leaky bucket, and a concatenation of a token bucket and leaky bucket are not included in Fig. 3 since their over maximum rate ratio are up to 33%, which are much higher than those obtained from the others. The packet loss rate obtained with leaky bucket is the same result obtained with a concatenation of token bucket and leaky bucket since the leaky bucket dominates the packet drop process. To sum up, neither the token bucket nor the leaky bucket is found to be suitable for supporting the maximum rate constraint service.

Moreover, there are some works about integrated shaping-scheduling architectures [3,6,10, 17,19], but they all used a concatenation of the rate controller and scheduler, which was challenged in this work. In [25], Zhu et al. proposed a deadline-curve based EDF scheduler (DC-EDF). While both WF²Q-M and DC-EDF do not use shaper, their targets are quite different. WF²Q-M aims to strictly enforce the maximum rate constraint and efficiently distributed the excess bandwidth; however, DC-EDF allows the potential earliness of a packet and guarantees end-to-end delay bounds.

3. WF²Q-M

We propose a new service discipline called WF²Q-M (worst-case fair weighted fair queueing with maximum rate control) to enforce the maximum rate constraint. WF²Q-M is more efficient than conventional two-stage rate-control service disciplines in that it uses only one set of queues and produces more accurate output than the policer-based rate-control service disciplines do since WF²Q-M strictly enforces the maximum rate constraint with low packet loss rate. Like other weighted fair service disciplines, WF²Q-M users can define a set of sessions and specify a positive real number for each session as its relative link sharing weight on the shared link. In addition, WF²Q-M users can assign maximum rates (also called *peak rates*) for sessions, called *maximum rate constrained* (*MRC*) sessions, as the upper bounds of transmission rates. MRC sessions that transmit data at their peak rates are *saturated*. Otherwise, the sessions are *non-saturated*. For non-MRC sessions and non-saturated MRC sessions, WF²Q-M allocates bandwidth to sessions proportional to their associated weights, just as WF²Q does. Table 1 lists the notations used in this paper and their definitions. For notations associated with

Table 1
Notations

a_i^k	Arrival time of the kth packet of session i
$B(t)$	The set of backlogged sessions at time t
$B_p(t)$	The set of saturated sessions at time t
$\overline{B_p(t)}$	The set of non-saturated sessions at time t
C	Link capacity
$d_{i,\text{SD}}^k$	Departure time of the kth packet of session i under scheduling discipline SD
e_i^k	Eligible time of the kth packet of session i under the maximum rate constraint
F_i^k	Virtual finishing time of the kth packet of session i
L_i^k	Size of the kth packet of session i
$\widehat{L}_j$	The packet length transmitted during time interval (l_j, l_{j+1})
MRC	Maximum rate constrained
$Norm(t)$	Normalization factor at time t
P_i	The maximum rate of session i
p_i^k	The kth packet of session i
$Q_{i,\text{SD}}(t)$	The queue size of session i at time t under scheduling discipline SD
r_i	The guaranteed rate of session i
$r\_GPS_i(t)$	The allocated bandwidth of session i at time t under the GPS policy
$r_i(t)$	The allocated bandwidth of session i at time t under the GPS-M policy
S_i^k	Virtual starting time of the kth packet of session i
SI_i	The saturation index of session i
$speedup(t)$	The real clock to virtual clock mapping ratio
SE_i	Session i
$W_{i,\text{SD}}(0,\tau)$	The work/service received by session i from time 0 to time τ under scheduling discipline SD
ϕ_i	The assigned weight of session i
$\phi_{B(t)}$	The sum of the assigned weights of backlogged sessions at time t
$\phi_{B_p(t)}$	The sum of the assigned weights of saturated sessions at time t

packets, such as length, time, etc., the superscripts denote packet numbers, and the subscripts denote session numbers. The scheduling discipline employed is also indicated in the second position of the subscript. When without ambiguity, the WF^{2}Q-M subscript may be omitted for simplicity; e.g., $d^k_{i,\mathrm{WF^2Q\text{-}M}}$ may be represented as d^k_i.

3.1. GPS-M model

The fluid model GPS-M (generalized processor sharing with maximum rate control) introduced in this section is used as the reference model of WF^{2}Q-M. GPS-M is an extension of GPS in that every session i is assigned a weight. Sessions that may be constrained by their assigned maximum rates P_i are called *maximum rate constrained* (MRC) sessions. If a MRC session receives a rate in the corresponding GPS that is higher than its assigned maximum rate, then we call the session *saturated*, and the set of saturated sessions is denoted as $B_p(t)$:

$$SE_i \in \begin{cases} B_p(t) & \text{if } \left(\dfrac{\phi_i}{\sum_{j\in B(t)} \phi_j}\right) * C > P_i, \\ \overline{B_p(t)} & \text{otherwise.} \end{cases} \quad (7)$$

We also denote the set of backlogged sessions not in $B_p(t)$ as $\overline{B_p(t)}$. In GPS-M, sessions in $B_p(t)$ receive their assigned maximum rates, and sessions in $\overline{B_p(t)}$ share the remaining bandwidth in proportion to their weights as in GPS. The allocated bandwidth of the session is defined as follows:

$$r_i(t) = \begin{cases} P_i, & \text{if } SE_i \in B_p(t), \\ \phi_i * Norm(t) & \text{otherwise,} \end{cases}$$
$$\text{where } Norm(t) = \frac{\left(C - \sum_{k\in B_p(t)} P_k\right)}{\phi_{B(t)} - \phi_{B_p(t)}}. \quad (8)$$

$\phi_{B(t)}$ is the sum of the assigned weights of backlogged sessions at time t, and $\phi_{B_p(t)}$ is the sum of the assigned weights of sessions in $B_p(t)$ at time t. Note that although $Norm(t)$ is infinite when $\phi_{B(t)}$ is equal to $\phi_{B_p(t)}$, we do not use $Norm(t)$ in this situation since no session is in $\overline{B_p(t)}$. Additionally, $\sum_{k\in B_p(t)} P_i$ will not exceed C when $\sum_{i\in B(t)} \frac{\phi_i}{\sum_{j\in B(t)} \phi_j} * C = C$, and $r_i(t)$ is equal to P_i only if $\left(\frac{\phi_i}{\sum_{j\in B(t)} \phi_j}\right) * C > P_i$. Therefore, the sum of the

maximum rates is always less than C if all the MRC sessions are saturated in the system.

GPS-M is the same as GPS in many ways; the server can serve all backlogged sessions simultaneously, and the service is infinitely divisible. The difference between GPS and GPS-M is that GPS serves session i with rate $(\phi_i/\phi_{B(t)}) \times C$, while GPS-M serves it with $r_i(t)$. In other words, GPS-M with no sessions in $B_p(t)$ is equivalent to GPS. When all the backlogged sessions are saturated and the sum of the maximum rates of the sessions in $B_p(t)$ is less than the link capacity, GPS-M becomes non-work conserving.

Formula (8) for determining the bandwidth allocation is a declarative definition. To calculate the allocated bandwidth, we need a procedural algorithm to distribute the excess bandwidths of saturated sessions (i.e., $C*\phi_i/\phi_{B(t)} - P_i$) among non-saturated sessions. The algorithm that finds $B_p(t)$ and its proof will be presented in Sections 3–5.

3.2. Virtual clock adjustment

Here we propose a mechanism called *virtual clock adjustment* that distributes the excess bandwidth from sessions in $B_p(t)$ among the sessions in $\overline{B_p(t)}$ in proportion to their assigned weights without recalculating the virtual starting and finishing times of packets of sessions in $\overline{B_p(t)}$. The following example shows how virtual clock adjustment works. Assume that there are four sessions sharing the same link. For simplicity, assume that all packets are of size 1 and that the link speed is 1. Let the weights of the four sessions be 50%, 25%, 12.5%, and 12.5%. Assume that session 1 is inactive and that each of the other sessions receives one packet at the beginning of every second. In Fig. 4, a rectangle represents a packet with virtual starting and finishing times in WF^{2}Q. The service order $(p^1_2, p^1_3, p^2_2, p^1_4, p^3_2, p^2_3, p^4_2, p^2_4, \ldots)$ is shown in Fig. 5. The transmission rates of sessions 2, 3, and 4 are 0.5, 0.25, and 0.25, respectively. Now let session 2 become an MRC session with a maximum rate of 0.4. Excess bandwidth in the amount of 0.1 is distributed among sessions 3 and 4 evenly. The resulting bandwidth of sessions 2, 3, and 4 is 0.4, 0.3 and 0.3 respectively.

For a virtual time based scheduler, when the virtual times of packets are modified, the scheduling sequence is changed; thus, the allocated bandwidths of sessions are changed. The simplest approach to time adjustment is to recalculate the virtual starting
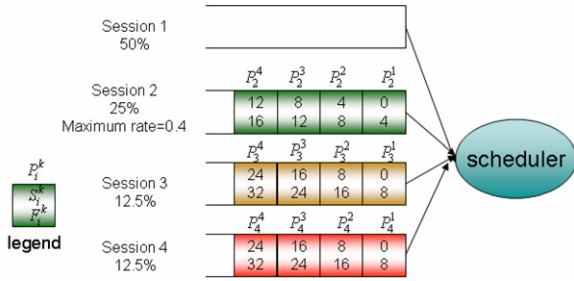
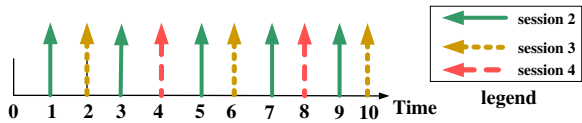Fig. 4. Virtual starting and virtual finishing times of packets in WF^{2}Q.

Fig. 5. WF^{2}Q service order corresponding to Fig. 4.

and finishing times of every packet in the system when the backlog changes. This method is infeasible, however, due to the high overhead incurred. The proposed virtual clock adjustment method aims to reduce the high computational complexity by adjusting the ticking rate of the virtual clock, and the virtual times of packets of saturated sessions are adjusted so that saturated sessions are constrained by their peak rates. In other words, the virtual times of sessions in $\overline{B_p}(t)$ are relatively ahead to gain higher transmission rate. The virtual clock $V(t)$ is

$$
V(0) = 0,
$$
$$
V(t + \tau) = V(t) + \frac{\tau}{speedup(t)}. \tag{9}
$$

Assume that the backlog of the server does not change during the time period $(t, t + \tau)$.

The real clock to virtual clock mapping ratio *speedup(t)* is $\phi_{B(t)} * \frac{\frac{\phi_{B(t)} - \phi_{B_p(t)}}{\phi_{B(t)}} * C}{C - \sum_{k \in B_p(t)} P_k}$, where $\phi_{B(t)}$ is the original ratio, the numerator is the original total bandwidth shared by $\overline{B_p}(t)$ sessions, and the denominator is the total bandwidth shared by $\overline{B_p}(t)$ sessions after excess bandwidth is received from $B_p(t)$ sessions. For instance, the virtual finish time of p_3^1 is 8, which maps to a real clock time of 4 in WF^{2}Q, while in WF^{2}Q-M, the real clock is 10/3 (i.e., $8 * (0.5 * (0.5/0.6))$). To summarize, the real clock to virtual clock mapping ratio of WF^{2}Q-M is defined as

$$
speedup(t) = \begin{cases} \frac{C}{Norm(t)} & \text{if } \phi_{B(t)} \neq \phi_{B_p(t)}, \\ \phi_{B(t)} & \text{otherwise.} \end{cases} \tag{10}
$$

Note that when there are only saturated sessions such that $\phi_{B(t)} = \phi_{B_p(t)}$, the *speedup(t)* of WF^{2}Q-M is the same as that of WF^{2}Q.

3.3. Maximum rate control models

In this section, we will present a fluid and a packet maximum rate control model and show their correspondence. To enforce maximum rate control for sessions in $B_p(t)$, the eligible time for each packet will be introduced, and only those packets whose eligible times have been exceeded will be considered for receiving service. The eligible time of a packet is defined as $e_i^k = \max(a_i^k, e_i^{k-1} + L_i^{k-1}/P_i)$. With the eligible time constraint, it is obvious that the sessions cannot transmit packets at rates that are higher than their maximum rates. Incorporating the eligible time, we can define the packet starting time $S_{i,\text{GPS-M}}^k$ and finishing time $F_{i,\text{GPS-M}}^k$ of the packets of sessions in $B_p(t)$ in GPS-M as follows:

$$
e_i^k = \max\left(a_i^k, e_i^{k-1} + \frac{L_i^{k-1}}{P_i}\right),
$$
$$
S_{i,\text{GPS-M}}^k = \max\{e_i^k, F_{i,\text{GPS-M}}^{k-1}\}, \tag{11}
$$
$$
F_{i,\text{GPS-M}}^k = S_{i,\text{GPS-M}}^k + \frac{L_i^k}{P_i}.
$$

To obtain the packet virtual starting time S_i^k and virtual finishing time F_i^k of the $B_p(t)$ packets in WF^{2}Q-M, the ratio function in formula (10) is applied to map the real clock times to virtual clock times of saturated sessions. The virtual starting and finishing times of a $B_p(t)$ packet, if *speedup(t)* maintains the same value during the transmission of L_i^k in GPS-M, are specified as follows:

$$
e_i^k = \max(a_i^k, e_i^{k-1} + \frac{L_i^{k-1}}{P_i}),
$$
$$
S_i^k = \max\{V(e_i^k), F_i^{k-1}\}, \tag{12}
$$
$$
F_i^k = S_i^k + \frac{L_i^k}{P_i} * \frac{1}{speedup(t)}.
$$

For $\overline{B_p}(t)$ sessions, the packet starting and finishing times in GPS-M, if $r_i(t)$ maintains the same value during the transmission of L_i^k in GPS-M, are defined as follows:

$$
S_{i,\text{GPS-M}}^k = \max\{a_i^k, F_{i,\text{GPS-M}}^{k-1}\},
$$
$$
F_{i,\text{GPS-M}}^k = S_{i,\text{GPS-M}}^k + \frac{L_i^k}{r_i(t)}. \tag{13}
$$

The virtual starting and finishing times of $\overline{B_p}(t)$ packets in WF^{2}Q-M are defined as follows:

$$S_i^k = \max\{V(a_i^k), F_i^{k-1}\},$$
$$F_i^k = S_i^k + \frac{L_i^k}{\phi_i * C}. \tag{14}$$

Formula (14) is the same as that in WF^{2}Q. Note that F_i^k remains unchanged when the system backlog changes.

We call a session continuously backlogged in WF^{2}Q-M if the session is serviced continuously in the corresponding GPS-M. The following theorem shows the correspondence (i.e., $S_i^k = V(S_{i,\text{GPS-M}}^k)$ and $F_i^k = V(F_{i,\text{GPS-M}}^k)$) between GPS-M and WF^{2}Q-M systems.

Since there may be backlogged session changes while a packet is being serviced, $speedup(t)$ and $r_i(t)$ cannot remain constant. Assume there are backlog changes at times $l_1, l_2, \ldots, l_q$, and that $\widehat{L}_j$ represents the portion of a packet transmitted during (l_j, l_{j+1}). When session SE_i maintains saturation, the virtual finishing times in formula (12) can be further revised as follows:

$$F_i^k = S_i^k + \sum_{j=1}^{q} \frac{\widehat{L}_j}{P_i} * \frac{1}{speedup(l_j)}. \tag{15}$$

The finishing time in formula (13) can be revised as follows:

$$F_{i,\text{GPS-M}}^k = S_{i,\text{GPS-M}}^k + \sum_{j=1}^{q} \frac{\widehat{L}_j}{r_i(l_j)}. \tag{16}$$

Note that if during transmission of a packet, the session status changes from saturated to non-saturated, or vice versa, and then the virtual finishing time can be obtained by using different formulas according to its status at the time.

Lemma 1. *For a continuously backlogged session SE_i, $F_i^k - S_i^k = V(F_{i,GPS\text{-}M}^k) - V(S_{i,GPS\text{-}M}^k)$.*

Proof. We will show that the lemma is true for the following cases.

Case 1: $SE_i \in \overline{B_p}(t)$.

From formulas (14) and (16), the transmission time of a packet measured in real time is $F_{i,\text{GPS-M}}^k - S_{i,\text{GPS-M}}^k = \sum_{j=1}^{q} \frac{\widehat{L}_j}{r_i(l_j)}$, and it is $F_i^k - S_i^k = \frac{L_i^k}{\phi_i * C}$ in virtual time. For two times t_1 and t_2 that $t_1 < t_2$, assume backlogged sessions change at time $l_1, l_2, \ldots, l_q$, then $V(t_2) - V(t_1) = \sum_{j=1}^{q-1} \frac{l_{j+1} - l_j}{speedup(l_j)}$. Therefore, $V(F_{i,\text{GPS-M}}^k) - V(S_{i,\text{GPS-M}}^k) = \sum_{j=1}^{q-1} \frac{l_{j+1} - l_j}{speedup(l_j)} =$

$\sum_{j=1}^{q} \frac{\widehat{L}_j}{r_i(l_j)} * \frac{1}{speedup(l_j)}$. After replacing $r_i(l_j)$ and $speedup(l_j)$ into the above equation, we can reduce the above to

$$V(F_{i,\text{GPS-M}}^k) - V(S_{i,\text{GPS-M}}^k) = \sum_{j=1}^{q} \frac{\widehat{L}_j}{\phi_i * C} = \frac{L_i^k}{\phi_i * C}.$$

Case 2: $SE_i \in B_p(t)$.

From formulas (11) and (15), the transmission time of a packet measured in real time is $F_{i,\text{GPS-M}}^k - S_{i,\text{GPS-M}}^k = \frac{L_i^k}{P_i}$, and it is $F_i^k - S_i^k = \sum_{j=1}^{q} \frac{\widehat{L}_j}{P_i} * \frac{1}{speedup(l_j)}$ in virtual time. Thus, $V(F_{i,\text{GPS-M}}^k) - V(S_{i,\text{GPS-M}}^k) = \sum_{j=1}^{q} \frac{l_{j+1} - l}{speedup(l_j)} = \sum_{j=1}^{q} \frac{\widehat{L}_j}{P_i} * \frac{1}{speedup(l_j)}$.

If the status of a session flip-flops between $B_p(t)$ and $\overline{B_p}(t)$ during the transmission of a packet, then Case 1 or Case 2 can be applied according to the status of the session, and this lemma still holds. □

Theorem 1. *For a continuously backlogged session SE_i, $S_i^k = V(S_{i,GPS\text{-}M}^k)$ and $F_i^k = V(F_{i,GPS\text{-}M}^k)$ for all backlogged packets p_i^k.*

Proof. We will prove this theorem by induction.

Step $N = 1$. From the definitions of formulas (12) and (14), $S_i^1 = V(S_{i,\text{GPS-M}}^1)$, and from Lemma 1, $F_i^1 - S_i^1 = V(F_{i,\text{GPS-M}}^1) - V(S_{i,\text{GPS-M}}^1)$. Together, these formulas imply that $F_i^1 = V(F_{i,\text{GPS-M}}^1)$.

Step $N = m$. Assume when $N = m$; this theorem is true, i.e., $S_i^m = V(S_{i,\text{GPS-M}}^m)$ and $F_i^m = V(F_{i,\text{GPS-M}}^m)$.

Step $N = m + 1$. As SE_i is continuously backlogged, $S_{i,\text{GPS-M}}^{m+1} = F_{i,\text{GPS-M}}^m$ and $S_i^{m+1} = F_i^m$. From Step $N = m$, we know that $F_i^m = V(F_{i,\text{GPS-M}}^m)$ and $S_i^{m+1} = V(F_{i,\text{GPS-M}}^m)$, which is equivalent to $V(S_{i,\text{GPS-M}}^{m+1})$. From Lemma 1, $F_i^{m+1} - S_i^{m+1} = V(F_{i,\text{GPS-M}}^{m+1}) - V(S_{i,\text{GPS-M}}^{m+1})$, after removing S_i^{m+1} and $V(S_{i,\text{GPS-M}}^{m+1})$, we have $F_i^{m+1} = V(F_{i,\text{GPS-M}}^{m+1})$. □

For the above example, the virtual starting and finishing times of packets in WF^{2}Q-M are shown in Fig. 6. Following formula (12), we find that $F_2^1 = \frac{1}{0.4} / (\frac{0.25}{0.6} * 1) = 6$ in WF^{2}Q-M, while F_2^1 is 4 in WF^{2}Q. The service order in WF^{2}Q-M is $(p_2^1, p_3^1, p_4^1, p_2^2, p_3^2, p_4^2, p_2^3, p_3^3, p_4^3, p_4^3, \ldots)$, as shown in Fig. 7. As shown in Figs. 6 and 7, the service rate of the three sessions is 4:3:3, which conforms to the bandwidth allocation polices of GPS-M.

3.4. Packet eligibility

While the introduction of eligible times of packets allows WF^{2}Q-M to enforce the maximum rate
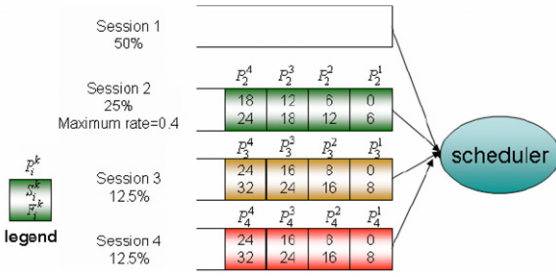
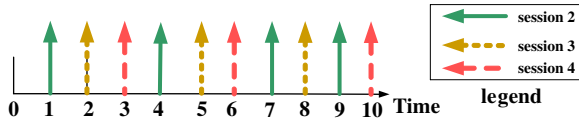Fig. 6. Virtual starting and finishing times of packets in WF²Q-M.

Fig. 7. WF²Q-M service order corresponding to Fig. 5.

constraint, it incurs overhead in maintaining the values of eligible times. In Theorem 2, we will show that the eligible time in formula (12) can be eliminated and that formula (12) can be reduced to the following formula. For $B_p(t)$ packets,

$$S_i^k = \max\{V(a_i^k), F_i^{k-1}\},$$
$$F_i^k = S_i^k + \frac{L_i^k}{P_i}/speedup(t). \tag{17}$$

To prove the theorem, we will present the following lemmas.

Lemma 2. *If $a_i^k \leqslant e_i^{k-1} + L_i^{k-1}/P_i$, then $S_i^k = F_i^{k-1}$ in formulas (12) and (17).*

Proof. The given condition $a_i^k \leqslant e_i^{k-1} + L_i^{k-1}/P_i$ and formula (11) imply $e_i^k = e_i^{k-1} + L_i^{k-1}/P_i$. From formula (11), $F_{i,\text{GPS-M}}^{k-1} = \max(e_i^{k-1}, F_{i,\text{GPS-M}}^{k-1}) + L_i^k/P_i \geqslant e_i^{k-1} + L_i^{k-1}/P_i$ leads to $F_{i,\text{GPS-M}}^{k-1} \geqslant e_i^k$. Since the real clock to virtual clock mapping function is monotonic, $F_{i,\text{GPS-M}}^{k-1} \geqslant e_i^k$ implies that $V(F_{i,\text{GPS-M}}^{k-1}) > V(e_i^k)$. Since $F_i^{k-1} = V(F_{i,\text{GPS-M}}^{k-1})$, we have $F_i^{k-1} > V(e_i^k)$, which further leads to $S_i^k = F_i^{k-1}$ in formula (12). For formula (17), since $F_i^{k-1} > V(e_i^k)$ and $V(e_i^k) > V(a_i^k)$, we have $F_i^{k-1} > V(a_i^k)$. Thus, $S_i^k = F_i^{k-1}$. □

Theorem 2. *The virtual starting and finishing times calculated using formula (12) are the same as formula (17).*

Proof. According to Lemma 2, when $a_i^k \leqslant e_i^{k-1} + L_i^{k-1}/P_i$, $S_i^k = F_i^{k-1}$ in formulas (12) and (17). Together with the fact that when $a_i^k > e_i^{k-1} + L_i^{k-1}/P_i$, S_i^k has the same value in both formulas (12)

and (17), no matter which value of $V(a_i^k)$ or F_i^{k-1} is larger. Since F_i^k can be computed from S_i^k, F_i^k also has the same value in both formulas (12) and (17), provided that S_i^k and F_i^k are the same whether they are computed using formula (12) or (17). □

The elimination of the eligible time e_i^k reduces the computation cost of calculating the virtual times and makes WF²Q-M similar to WF²Q in terms of representation and proofs of properties.

3.5. Packet processing algorithms

Having presented the models and their relationships, we can proceed to introduce WF²Q-M packet processing algorithms along with major functions, such as maintaining the virtual clock, calculating, adjusting the virtual starting and finishing times, and scheduling packet service order. In order to reduce the computation cost, the virtual clock advances only when a packet enters the system or the system backlog changes. For the same reason, only when a packet p_i^k reaches the head of the session, S_i^k and F_i^k are calculated according to formulas (14) and (17). If the backlogged sessions change, $r_i(t)$ and the virtual starting and finishing times of the head packets of sessions in $B_p(t)$ need to be recomputed.

Fig. 8 shows the algorithm *Enqueue*( ), which is activated when a packet arrives. The virtual clock is updated to the time of packet arrival. If the session is empty, the arrival of the packet backlogs the session and calls the procedure *Backlogchange*( ), shown in Fig. 9. When the system backlog changes, the shared bandwidths of MRC sessions must be recalculated to determine the saturated sessions using the procedure *findB_p(t)*, and the ratio *speedup(t)* must also be calculated. The procedure *Time_Adjust*( ) is used to adjust the virtual starting and finishing times of head packets in saturated sessions. The packet virtual starting and finishing times of the arriving packet are then calculated. Next, the procedure checks if the *idle_timer* is set, which indicates that the system has no eligible packet to service. If the timer is set, it cancels the *idle_timer*, and *Dequeue*( ) is used to select a packet to service.

As saturated sessions may become non-saturated after the system backlog changes, the virtual starting and finishing times of the head packet of sessions need to be adjusted using the algorithm *Time_Adjust*( ), shown in Fig. 10. (Note that the packet virtual times of sessions in $\overline{B_p}(t)$ need not

procedure *Enqueue*(p_i^k) {

1. Advance virtual clock via calculation of $V(a_i^k)$ using Formula (9)

2. Insert the packet at the end of session SE_i

3. if SE_i is empty {

    3.1 Call *Back* log *change*( )

    3.2 Calculate S_i^k and F_i^k using Formulas (14) or (17)

    3.3 if *idle_timer* is set, cancel the timer

    }

4. if (the system is idle or *idle_time*r is canceled)

    4.1 Call *Dequeue*( )

}

Fig. 8. Algorithm for packet *Enqueue*.

procedure *Back* log *change()*{

1. Calculate $r_i(t)$ for all $SE_i \in MRC$

        and determine $B_p(t)$

2. Calculate *speedup(t)* using Formula (10)

3. Call *Time _ Adjust*()}

Fig. 9. Algorithm for *Backlogchange*.

procedure *Time_Adjust()* {

1. for head packet p_i^k of $SE_i \in B_p(t^-)$

  1.1 if $V(t) < F_i^{k-1}${

    1.1.1 $S_i^k = V(t) + \dfrac{(F_i^{k-1} - V(t))*speedup(t^-)}{speedup(t)} * \dfrac{r_i(t^-)}{r_i(t)}$

    1.1.2 $F_i^k = S_i^k + \dfrac{L_i^k}{r_i(t)} * \dfrac{1}{speedup(t)}$

    }

  1.2 else

    1.2.1 $F_i^k = V(t) + \dfrac{(F_i^k - V(t))*speedup(t^-)}{speedup(t)} * \dfrac{r_i(t^-)}{r_i(t)}$

}

Fig. 10. Algorithm for virtual starting and finishing times recalculation.

be recalculated.) Let t^- denote the time immediately before time t when *Enqueue*( ) or *Dequeue*( ) is called. When $V(t) < F_i^{k-1}$, the virtual starting time of the head packet needs to be adjusted. Then the virtual finishing time is adjusted. $r_i(t^-)$ in *Time_Adjust*( ) is P_i, and $r_i(t)$ is calculated by Statement 1 in Fig. 9. The factor $r_i(t^-)/r_i(t)$ is used to adjust the virtual times as the allocated bandwidth changes.

In order to minimize the time required to recalculate the allocated bandwidths of MRC sessions, we must first find all the eventually saturated sessions and redistribute their excess bandwidths. Once the eventually saturated sessions are identified, the allocated bandwidths of non-saturated MRC sessions need to be calculated only once, which greatly reduces the computation cost. We find whether a session is eventually saturated is related to its weight and assigned maximum rate. The saturation index, SI_i, defined as the weight divided by the difference between the assigned maximum rate and the allocated bandwidth in the corresponding GPS, indicates the likelihood of SE_i becoming saturated. We can find saturated sessions more efficiently by using the saturation index.

Lemma 3. *For two sessions SE_i and $SE_j \in MRC$, and $SI_i > SI_j$ (as defined in Statement 2.3 in Fig. 11), if $SE_i \notin B_p(t)$, then $SE_j \notin B_p(t)$.*

Procedure *findB*$_p(t)$ (){

1. Set $B_p(t) = null$

2. for $SE_i \in MRC$ {

    2.1 $r\_GPS_i(t) = (\phi_i / \phi_{B(t)}) * C$

    2.2 if $(r\_GPS_i(t) > P_i)$ $B_p(t) = B_p(t) \cup SE_i$

    2.3 else $SI_i = \phi_i / (P_i - r\_GPS_i(t))$ }

3. for $SE_i \in \{MRC - B_p(t)\}$ in the descending order of SI_i {

    3.1 $r_i(t) = (C - \sum\limits_{k \in B_p(t)} P_k) * \phi_i / (\phi_{B(t)} - \phi_{B_p(t)})$

    3.2 if $(r_i(t) > P_i)$ {

    3.3 $B_p(t) = B_p(t) \cup SE_i$

    3.4 $r_i(t) = P_i$}

    3.5 else break }}

Fig. 11. The procedure *findsB*$_p(t)$.

Proof. Assume that there exists SE_o, $j > o > i$, and $SI_i > SI_o > SI_j$ such that SE_o is the first session after SE_i and $(C - \sum_{k \in B_p(t)} P_k) * \phi_o / (\phi_{B(t)} - \phi_{B_P(t)}) > P_o$. Since $SI_i > SI_o$, according to the definition of the saturation index, we have

$$\frac{\phi_i}{P_i - r\_GPS_i(t)} > \frac{\phi_o}{P_o - r\_GPS_o(t)}. \tag{18}$$

Note that the allocated bandwidth of SE_i, $(C - \sum_{k \in B_p(t)} P_k) * \phi_i / (\phi_{B(t)} - \phi_{B_P(t)})$, can be rewritten as $r\_GPS_i(t) + \frac{\phi_i}{\phi_{B(t)} - \phi_{B_P(t)}} * \sum_{k \in B_p(t)} (r\_GPS_k(t) - P_k)$. Let $C' = \sum_{k \in B_p(t)} (r\_GPS_k(t) - P_k)$. From $SE_i \notin B_p(t)$, we get

$$\frac{\phi_i}{\phi_{B(t)} - \phi_{B_P(t)}} * C' + r\_GPS_i(t) < P_i. \tag{19}$$

Since SE_o is the first session after SE_i such that $(C - \sum_{k \in B_p(t)} P_k) * \phi_o / (\phi_{B(t)} - \phi_{B_P(t)}) > P_o$, $B_P(t)$ remains the same, and the following condition holds:

$$\frac{\phi_o}{\phi_{B(t)} - \phi_{B_P(t)}} * C' + r\_GPS_o(t) > P_o. \tag{20}$$

From (19), we have

$$\frac{C'}{\phi_{B(t)} - \phi_{B_P(t)}} < \frac{P_i - r\_GPS_i(t)}{\phi_i}. \tag{21}$$

Similarly, from (20), we have

$$\frac{C'}{\phi_{B(t)} - \phi_{B_P(t)}} > \frac{P_o - r\_GPS_o(t)}{\phi_o}. \tag{22}$$

From (21) and (22), we get

$$(P_i - r\_GPS_i(t))/\phi_i > (P_o - r\_GPS_o(t))/\phi_o,$$

which can be rewritten as

$$\phi_i/(P_i - r\_GPS_i(t)) < \phi_o/(P_o - r\_GPS_o(t)). \tag{23}$$

Inequality (23) violates inequality (18). Therefore, such an SE_o cannot exist. Since no SE_o exists, $j > o > i$ such that $(C - \sum_{k \in B_p(t)} P_k) * \phi_o / (\phi_{B(t)} - \phi_{B_P(t)}) > P_o$, thus SE_i and SE_j have the same $B_p(t)$. The previous proof shows that $(C - \sum_{k \in B_p(t)} P_k) * \phi_j / (\phi_{B(t)} - \phi_{B_P(t)})$ must be less than P_j. Therefore, SE_j cannot be in $B_p(t)$. $\square$

Lemma 3 implies that if a session with SI_i is not saturated, then sessions with saturation indexes less than SI_i will not become saturated. In the other words, sessions with saturation indexes larger than SI_i must become saturated if SE_i is saturated. Therefore, we only need to identify the saturated session with the lowest saturation index by using

sorting or partitioning algorithms; then, the sessions with higher saturation indexes will be saturated sessions with their assigned maximum rates, while the sessions with lower saturation indexes will be allocated bandwidth proportional to their weights.

Fig. 11 presents algorithm $findB_p(t)$, used to determine $B_p(t)$. Statement 2 performs the first scan to find sessions in $B_p(t)$ and calculate SI_i. Before Statement 3 is executed, sessions not in $B_p(t)$ are sorted in the descending order of SI; i.e., for SE_i and $SE_j \in MRC - B_p(t)$, $i < j$ if $SI_i > SI_j$. Statement 3.2 is a termination test, which ensures that the algorithm stops if for a session SE_i, the condition $r_i(t) > P_i$ is not satisfied.

Theorem 3. *Algorithm $findB_p(t)$ correctly determines $B_p(t)$.*

Proof. Assume that Algorithm $findB_p(t)$ terminates when $i = k$ at Statement 3.5. From Lemma 3, sessions $SE_k, SE_{k+1}, \ldots, SE_{|MRC|}$ are not saturated. For sessions $SE_1, SE_2, \ldots, SE_{k-1}$, since they pass the condition $r_i(t) > P_i$ in Statement 3.2, the sessions must be saturated by definition. In other cases, the algorithm terminates with the all tests at Statement 3.2 being true such that all sessions are in $B_p(t)$. Therefore Algorithm $findB_p(t)$ correctly finds $B_p(t)$. $\square$

Algorithm $Dequeue(\ )$, shown in Fig. 12, is activated when the system is ready to serve the next available packet. Like WF^{2}Q, WF^{2}Q-M only considers the set of packets that have started receiving

```
procedure Dequeue() {
1. Let idx = ARGMIN{F_i^h| p_i^h = Head_Packet(SE_i) & S_i^h ≤ V(t)}
          i∈B(t)
2. if  idx == null {
    2.1 Let idx = ARGMIN{S_i^h | p_i^h = Head_Packet(SE_i)}
    2.2 idle_time = (S_idx^h − V(t))*speedup(t)
    2.3 Set_idleTimer(idle_time, Dequeue())
    2.4 return
    }
3 Serve the packet p_idx^h
4. if SE_idx ≠ null
    4.1 Calculate S_idx^{h+1} and F_idx^{h+1} using Formula (14) or (17)
5. else {
    5.1 back log change_time = (F_idx^h − V(t))* speedup(t)
    5.2  Set_back log Timer(back log change_time, Back log change())
    }}
```

Fig. 12. Algorithm for packet *Dequeue*.

service in the corresponding GPS-M system and selects the packet that will complete service first in GPS-M for servicing (i.e., with the smallest virtual finishing time). If the system has no eligible packet at time t, WF^{2}Q-M sets an *idle_timer* (as in Statement 2.2) for the packet with the smallest virtual starting time, and the system becomes idle. The *idle_timer* expires after an *idle_time*, and then the procedure calls itself *Dequeue*( ) and starts another busy period. After serving a packet, if the session becomes empty, and, thus, incurs a backlogged session change, the system will set another timer (*backlog_timer*). When *backlog_timer* expires at the time of the packet departure in the corresponding GPS-M system, the system will call the procedure *Backlogchange*( ). Note that WF^{2}Q-M needs only one additional timer (i.e., idle_timer). Since *idle_timer* is set only when the system is idle, the overhead is almost negligible. When a packet arrives at an idle system, the *idle_timer* will be cancelled as described in Statement 3.5 of *Enqueue*( ).

The computational costs of processing one packet in WF^{2}Q-M include one call to *Enqueue*( ) when the packet arrives and one call to *Dequeue*( ) when it is scheduled for service. When a packet arrives, the procedure takes constant time to advance the virtual clock and insert the packet into its associated session queue. The time complexity of *Dequeue*( ) is dominated by the maintenance of a heap data structure for storing the virtual starting and finishing times of head packets and the selection of the packet with the smallest virtual finishing time from the heap in Statement 1. Maintaining the set of eligible sessions sorted according to their virtual finishing times can be accomplished with $O(\log N)$ complexity [18], where N is the total number of sessions. *Time_Adjust*( ) is executed when backlogged sessions change, and its complexity is $O(M)$. (Note that M is the size of *MRC*.) If the session is empty before the arrival of the packet, $findB_p(t)$ is executed with time complexity $O(M \log M)$ [4], which is dominated by Statement 3, which sorts sessions in the descending order according to *SI*. $findB_p(t)$ also dominates the worst-case time complexity of *Enqueue*( ) and *Dequeue*( ). It has been proved that the number of backlog changes in an arbitrarily short interval of time is $O(N)$; hence, in such a case, the worst-case complexity of WF^{2}Q-M will be $O(N * M \log M)$. For the theoretical lower bound, SELECT and PARTITION [4] can be used to replace the sorting operation in algorithm $findB_p(t)$, shown in Fig. 11, and the complexity will be reduced

to $O(N * M)$. However, due to the high overhead when M is not sufficiently large, this approach is not practical and, thus not adopted in the proposed algorithm. Since the set of saturated sessions is determined only when the system backlog changes, not at every packet enqueue or dequeue, in practice, the real computation cost can be further reduced. Compared with WF^{2}Q, WF^{2}Q-M has extra overheads in calculating the shared rates of sessions and the virtual starting and finishing tags of head packets of the saturated sessions when the backlogged sessions change. Since the incurred overheads are proportional to the number of maximum rate constrained (MRC) sessions, the overheads may be negligible for the situations where the number of MRC sessions is only a small portion of total sessions. Additionally, the target of this algorithm is the edge routers, not in the backbone routers; therefore, the benefits of the maximum rate constraint outweigh the increased complexity.

4. WF^{2}Q-M system properties

In this section, we will discuss the properties of WF^{2}Q-M, including its conformity in terms of bandwidth allocation with GPS-M, its delay and work bounds, and its work-conserving characteristics. In GPS-M, sessions in $B_p(t)$ receive their peak rates, and the remaining capacity of the system is shared by sessions in $\overline{B_p}(t)$ in proportion to their assigned weights. Theorem 4 shows that WF^{2}Q-M achieves the same rate allocation as GPS-M. The delay and work bounds of WF^{2}Q-M with respect to GPS-M, which are analogous to those of WF^{2}Q with respect to GPS, are given in Theorem 5. While GPS-M, according to its defined rate allocation, is not always work conserving, Theorem 6 presents the work-conserving property of WF^{2}Q-M.

Due to the service granularity difference between a packet system and a fluid system, the bandwidths allocated to sessions of WF^{2}Q-M and GPS-M cannot be equivalent at all time points. In the following, we will show the bandwidth allocation conformity of WF^{2}Q-M with GPS-M under certain conditions. Lemma 4 shows the relation of services received by a saturated session and a non-saturated session. Theorem 4 applies the results of Lemma 4 and concludes that WF^{2}Q-M conforms with GPS-M in bandwidth allocation under the given assumption.

Lemma 4. *In WF^{2}Q-M, if $SE_i \in B_p(t)$ and $SE_j \in \overline{B_p}(t)$ become backlogged at time 0, and the if*

last serviced packets of both sessions p_i^k and p_j^l have the same virtual finishing time, τ, then the proportion of their received services is as follows:

$$W_{i,\text{WF}^2\text{Q-M}}(0,\tau) : W_{j,\text{WF}^2\text{Q-M}}^{(0,\tau)} = P_i : \phi_j * Norm(t). \tag{24}$$

Proof. As both sessions are continuously backlogged, the virtual finishing times of p_i^k and p_j^l are $F_i^k = \sum_{m=1,k}(L_i^m/P_i)/speedup(t)$ and $F_j^l = \sum_{m=1,l} L_j^m/(\phi_j * C)$, respectively. Since $F_i^m = F_j^l = \tau$, we have

$$\sum_{m=1,k} L_i^m/(P_i * speedup(t)) = \sum_{m=1,l} L_j^m/(\phi_j * C). \tag{25}$$

As $W_{i,\text{WF}^2\text{Q-M}}(0,\tau) = \sum_{m=1,k} L_i^m$ and $W_{j,\text{WF}^2\text{Q-M}}(0,\tau) = \sum_{m=1,k} L_j^m$, we can replace them in (25) and obtain

$$\begin{aligned} W_{i,\text{WF}^2\text{Q-M}}(0,\tau) &: W_{j,\text{WF}^2\text{Q-M}}(0,\tau) \\ &= P_i * speedup(t) : \phi_j * C. \end{aligned} \tag{26}$$

Extending $speedup(t)$ using formula (10), we have

$$\begin{aligned} W_{i,\text{WF}^2\text{Q-M}}(0,\tau) &: W_{j,\text{WF}^2\text{Q-M}}(0,\tau) \\ &= P_i : \phi_j * C \Bigg/ \left(\frac{(\phi_{B(t)} - \phi_{B_P(t)}) * C}{(C - \sum_{k \in B_p(t)} P_k)} \right), \end{aligned} \tag{27}$$

which can be further reduced to

$$W_{i,\text{WF}^2\text{Q-M}}(0,\tau) : W_{j,\text{WF}^2\text{Q-M}}(0,\tau) = P_i : \phi_j * Norm(t). \quad \square$$

Theorem 4. *In WF^2Q-M, if all $SE_k \in B_p(t)$ and $SE_j \in \overline{B}_p(t)$ become backlogged at time 0, and if the last serviced packets of both sessions have the same virtual finishing time, τ, then the transmission rate of sessions $SE_k \in B_p(t)$ is P_k and that of $SE_j \in \overline{B}_p(t)$ is $\phi_j * Norm(t)$.*

Proof. Let tr_i be the average transmission rate of SE_i from time 0 to $V^{-1}(\tau)$. Then $tr_i = W_{i,\text{WF}^2\text{Q-M}}(0,\tau)/V^{-1}(\tau)$. As $\text{WF}^2\text{Q-M}$ is work conserving, $\sum_{i \in B(t)} tr_i = C$. From Lemma 4, we derive

$$tr_k : tr_j = P_k : \phi_j * Norm(t), \tag{28}$$

where $SE_k \in B_p(t)$ and $SE_j \in \overline{B}_p(\tau)$. Totaling all tr_i in $B(t)$, we have

$$\sum_{i \in B_P(t)} \alpha P_i + \sum_{j \in \overline{B_P}(t)} \alpha \left(\frac{\phi_j}{\phi_{B(t)} - \phi_{B_P(t)}} \right) \left(C - \sum_{k \in B_p(t)} P_k \right) = C, \tag{29}$$

where α is the constant.

Since $\sum_{j \in \overline{B_P}(t)} \alpha \left(\frac{\phi_j}{\phi_{B(t)} - \phi_{B_P(t)}} \right) = 1$, formula (29) can be reduced to

$$\alpha \sum_{k \in B_p(t)} P_k + 1 * \left(C - \sum_{k \in B_p(t)} P_k \right) = C \Rightarrow \alpha C = C \Rightarrow \alpha = 1.$$

Therefore, $tr_k = P_k$ if $SE_k \in B_p(t)$, and $tr_j = \phi_j * Norm(t)$ if $SE_j \in \overline{B}_p(t)$. $\square$

The above theorem shows that the resource allocation of WF^{2}Q-M conforms with the GPS-M service discipline (i.e., conforms with formula (8)) under the given condition. However, the condition is too strict to be applied in general cases. Theorem 5 shows the delay and work bounds of WF^{2}Q-M with respect to the corresponding model of GPS-M.

Theorem 5. *The following inequalities hold for WF^2Q-M.*

$$d_{i,\text{WF}^2\text{Q-M}}^k - d_{i,\text{GPS-M}} \leqslant \frac{L_{\max}}{C}, \tag{30}$$

$$W_{i,\text{GPS-M}}(0,\tau) - W_{i,\text{WF}^2\text{Q-M}}(0,\tau) \leqslant L_{\max}, \tag{31}$$

$$W_{i,\text{WF}^2\text{Q-M}}(0,\tau) - W_{i,\text{GPS-M}}(0,\tau) \leqslant \left(1 - \frac{r_i}{C} \right) L_{i,\max}, \tag{32}$$

$$d_{i,\text{WF}^2\text{Q-M}}^k - a_i^k \leqslant \frac{Q_{i,\text{WF}^2\text{Q-M}}(a_i^k)}{r_i} + \frac{L_{i,\max}}{r_i} - \frac{L_{i,\max}}{C} + \frac{L_{\max}}{C}, \tag{33}$$

where $Q_{i,\text{SD}}(t)$ is the queue size of session i at time t under SD.

Proof. The proof is similar to those given in Section 3 of [1] hence, is not repeated here to save space. A detailed proof can be found in [11]. $\square$

When there exists at least one non-MRC backlogged session or when the sum of the peak rates of saturated sessions is greater than the link capacity, the system is called *ingress-busy*. If a system is not ingress-busy, the sum of the peak rates of the saturated sessions is less than the link capacity, and obviously, the system is not work conserving. An ingress-busy GPS-M system is obviously work conserving since its channel capacity is fully utilized during busy periods. The following theorem shows that if a WF^{2}Q-M system is ingress-busy, it is work conserving; i.e., there is at least one packet whose virtual starting time is less than or equal to the system virtual clock when WF^{2}Q-M selects the next packet for servicing, as indicated n Statement 1 of *Dequeue*( ) in Fig. 12.

To prove the work-conserving property, we will employ the concept of the rate-controlled service discipline. As introduced in Section 2, a rate-controlled service discipline has two stages: a rate controller and a scheduler. We will consider two rate-controlled service disciplines, R-WFQ-M and R-GPS-M, shown in Figs. 13 and 14 respectively, which have the same rate controllers but different schedulers. The schedulers for R-WFQ-M and R-GPS-M are WFQ-M and GPS-M, respectively. WFQ-M has the same scheduling algorithm as WF^2Q-M, i.e., the virtual clock adjustment and the bandwidth sharing policy, but WFQ-M only selects packets with the smallest virtual finishing time. We call the GPS-M server embedded within the R-GPS-M service discipline GPS-M* to distinguish it from the standalone GPS-M server. The only difference between GPS-M and GPS-M* is that the packet arrival pattern of GPS-M* is shaped by the rate-controller of R-GPS-M. Also, we refer to the WFQ-M server embedded in R-WFQ-M as WFQ-M*. The eligible time for p_i^k in the rate controller is defined as $e_i^k = S_{i,\text{GPS-M}}^k$, where $S_{i,\text{GPS-M}}^k$ is the time when the packet starts being serviced in the corresponding GPS-M system. If two scheduling systems are said to be *equivalent,* the instantaneous service rates and the departure times of packets for each session at any given time are exactly the same for any arrival sequence.

Theorem 6. *If WF^2Q-M is ingress-busy, it is work conserving.*

Proof. We will show that the busy periods of R-GPS-M and R-WFQ-M are the same when ingress-busy. For the same input pattern, we have:

1. R-GPS-M and GPS-M* have identical system busy time because GPS-M* is the scheduler embedded in R-GPS-M,
2. GPS-M* and WFQ-M* have identical busy time because WFQ-M* is a work-conserving service discipline since it selects packets with the smallest virtual finishing time, and GPS-M* is always busy in this situation,
3. R-WFQ-M and WFQ-M* have identical busy time because WFQ-M* is the embedded scheduling algorithm of R-WFQ-M.

They follows that R-GPS-M and R-WFQ-M have identical busy times, so, GPS-M and WF^2Q-M have the same busy period since R-GPS-M and
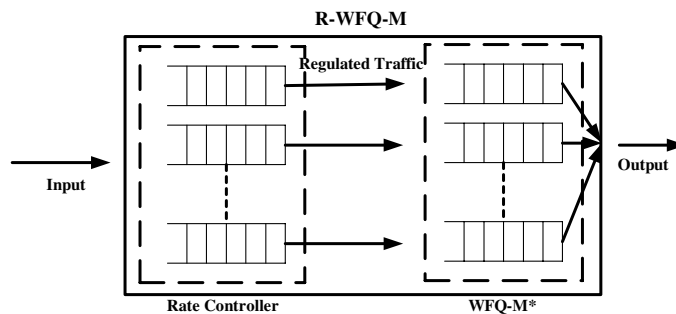
R-WFQ-M
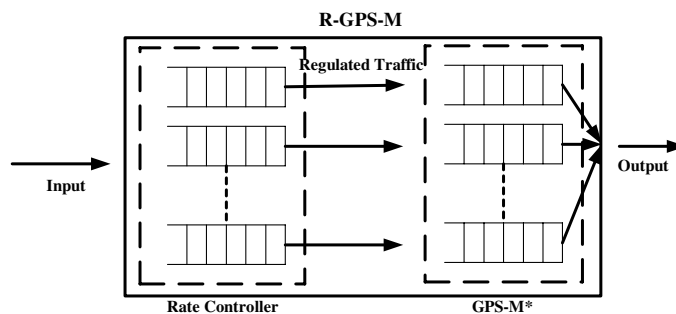

Fig. 13. R-WFQ-M.

R-GPS-M

Fig. 14. R-GPS-M.

R-WFQ-M systems are equivalent to its corresponding GPS-M and WF2Q-M systems, respectively, when the system is ingress-busy. Hence, WF^2Q-M is work conserving since GPS-M is always busy when the system is ingress-busy. Note that this proof is similar to the one in Section 3 of [1]. □

5. Simulations

In this section, we will present simulations conducted to examine the performance of WF^2Q-M. We implemented WF^2Q, GPS-M, and WF^2Q-M modules on the ns-2 [14] simulator. The simulation topology is shown in Fig. 15. Each of the senders, S1, S2, S3, and S4 generated 5 Mbps of CBR UDP traffic to the receivers R1, R2, R3, and R4, respectively. A sender and a receiver pair formed a session such that the transmission periods were from 1 s to 10 s, 1 s to 12 s, 1 s to 14 s, and 1 s to 5 s for sessions 1 (S1–R1), 2 (S2–R2), 3 (S3–R3), and 4 (S4–R4), respectively. The packet sizes were uniformly distributed between 100 and 1500 bytes. The assigned weights at the output port of router n2 were 10%, 15%, 25%, and 50% for sessions 1, 2, 3, and 4, respectively, while S3–R3 was assigned a maximum rate of 3 Mbps.

We measured the throughput of each session on the bottleneck link between n2 and n3. Fig. 16a shows the results when WF^2Q was used as the scheduling policy of the bottleneck link, and Fig. 16b shows the same simulation context when WF^2Q-M was used. As shown in Fig. 16, when there was no session in $B_p(t)$ from 1 s to 5 s, WF^2Q and WF^2Q-M produced the same result. After 5 s, since the allocated bandwidth of session 3 was more than 3 Mbps in GPS-M, the transmission rate of the session was restricted to 3 Mbps in WF^2Q-M, and the excess bandwidth (2 Mbps in this case) was distributed between session 1 and 2 according to the ratio
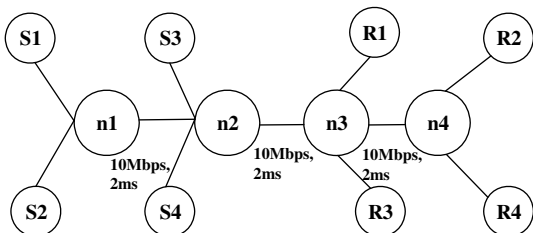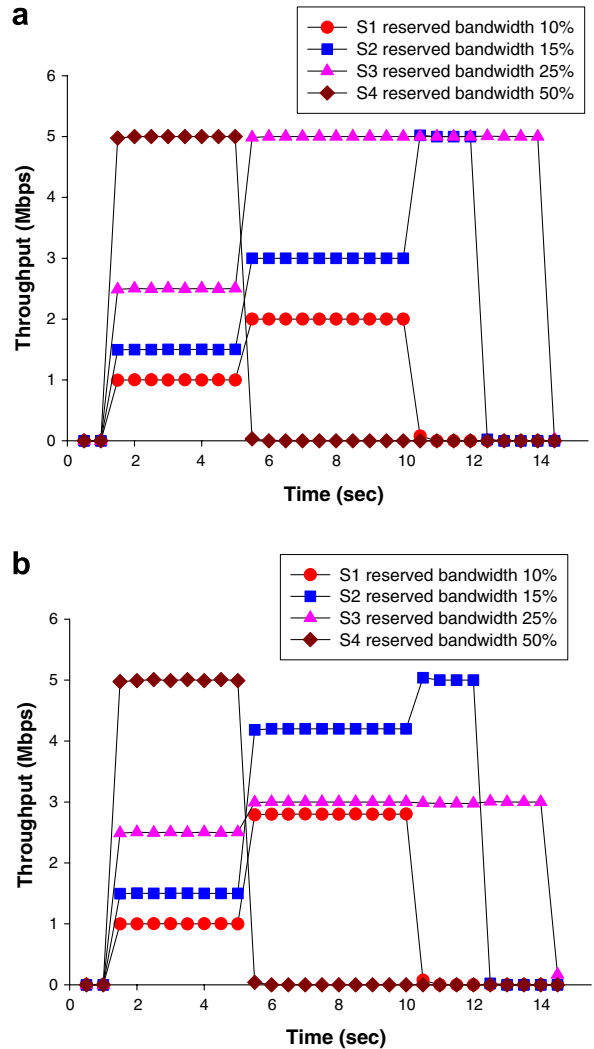

Fig. 15. Simulation topology.

Fig. 16. The performance of (a) WF^2Q and (b) WF^2Q-M.

of their assigned weights. At 10 s, S1 stopped transmitting data, and its bandwidth was shared by session 2 and session 3 in WF^2Q, while in WF^2Q-M, because the transmission rate of session 3 was restricted to 3 Mbps, the system became non-work conserving.

Another measurement shows the relationship between the amount of service received from sessions under WF^2Q-M and GPS-M. Let $W_{i,WF^2Q\text{-}M}(0,t)$ and $W_{i,GPS\text{-}M}(0,t)$ be the amount of service received by session i from time 0 to t under WF^2Q-M and GPS-M, respectively. To show the relationship between $W_{i,WF^2Q\text{-}M}(0,t)$ and $W_{i,GPS\text{-}M}(0,t)$, let $W_{i,\text{upper-bound}}$ be $W_{i,GPS\text{-}M} + \left(1 - \frac{r_i}{C}\right)L_{i,\max}$, and let $W_{i,\text{lower-bound}}$ be $W_{i,GPS\text{-}M} - L_{\max}$. Fig. 17

a

b
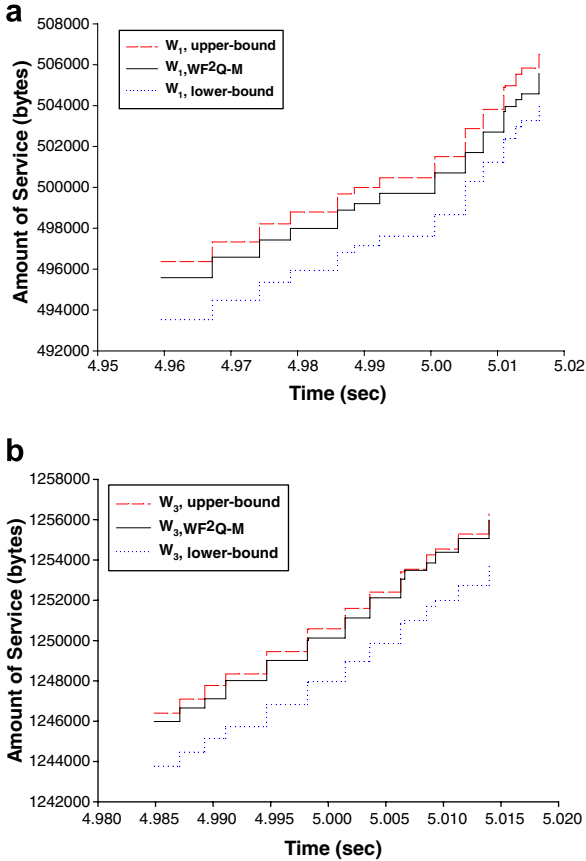
Fig. 17. The relationship between $W_{i,\mathrm{WF^2Q\text{-}M}}(0,t)$ and $W_{i,\mathrm{GPS\text{-}M}}(0,t)$: (a) session 1: a $\overline{B_p}(\tau)$ session and (b) session 3: a $B_p(\tau)$ session.

shows amount of the services received near time 5 s. It can be seen that the properties $W_{i,\mathrm{upper\text{-}bound}}(0,t) \geqslant W_{i,\mathrm{WF^2Q\text{-}M}}(0,t) \geqslant W_{i,\mathrm{lower\text{-}bound}}(0,t)$ hold for both non-saturated sessions (session 1 in Fig. 17a) and saturated sessions (session 3 in Fig. 17b). In other words, inequalities (31) and (32) of Theorem 5 are verified.

Finally, we show the performance evaluation about the execution time of $findB_p(t)$, which dominates the worst-case time complexity of *Enqueue*( ) and *Dequeue*( ). We implemented the algorithm on a PC-based platform with an Intel P3 550 CPU, a 256M RAM with OS LINUX RedHat 7.2. We used RDTSC (read time stamp counter), which returns the number of clock cycles from the time the CPU is powered up or reset, instead of function *gettimeofday,* for accurate measurements. In the evaluation, the average run time from ten runs for each different number of MRC sessions is reported, as
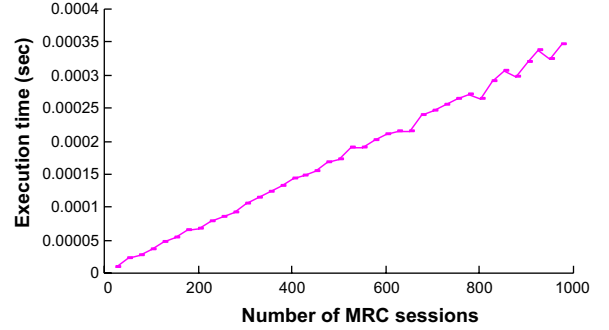

Fig. 18. The average running time for MRC sessions from 1 to 1000.

shown in Fig. 18. The weights and maximum rates of MRC sessions were randomly selected. We can see that for the case of 1000 MRC sessions, the execution time is only 0.35 ms.

6. Discussion and conclusions

In this paper, we have proposed a new service discipline, $\mathrm{WF^2Q\text{-}M}$, that can simultaneously provide minimum service rate guarantees and a maximum service rate constraint. We have shown that a packet's eligible time can be merged into its virtual starting time to reduce the complexity and make $\mathrm{WF^2Q\text{-}M}$ similar to $\mathrm{WF^2Q}$. The virtual clock adjustment allows the sharing of excess bandwidth without requiring recalculation of the packet virtual starting and finishing times. Additionally, $\mathrm{WF^2Q\text{-}M}$ takes advantage of the good properties of $\mathrm{WF^2Q}$ to achieve weighted fair queuing. We also have shown that a fluid reference model, similar to $\mathrm{WF^2Q}$ bounded by the GPS model, theoretically bounds the performance of $\mathrm{WF^2Q\text{-}M}$.

This paper has shown that $\mathrm{WF^2Q\text{-}M}$ can effectively enforce maximum transmission rate constraints of sessions. In some situations, users want to control the both the average and maximum rates. The $\mathrm{WF^2Q\text{-}M}$ service discipline combined with a token bucket policer can achieve this goal since the architecture is similar to that shown in Fig. 2. The token bucket controls the average rate of a session, but the bucket size of the token bucket allows burst traffic to be transmitted with a rate that exceeds the maximum rate of control. By using $\mathrm{WF^2Q\text{-}M}$ as the packet scheduler, we can control the maximum transmission rate. As a result, the combination of the token bucket and $\mathrm{WF^2Q\text{-}M}$

along with a careful selection of control parameters can support both average and maximum rate constraints.

References

[1] J.C.R. Bennett, H. Zhang, WF^{2}Q: worst-case fair weighted fair queueing, in: Procedings of the IEEE INFOCOM'96, San Francisco, CA, March 1996.

[2] J.C.R. Bennett, H. Zhang, Hierarchical packet fair queueing algorithms, IEEE/ACM Transactions on Networking 5 (Oct) (1997) 675–689.

[3] H.J. Chao, J.S. Hong, Design of an ATM shaping multiplexer with guaranteed output burstiness, Computer Systems Science and Engineering. 12 (2) (1997).

[4] H. Cormen, E. Leiserson, L. Rivest, Clifford Stein, Introduction to Algorithms, second ed., MIT Inc., 2001.

[5] S. Floyd, V. Jacobson, Link-sharing and resource management models for packet networks, IEEE/ACM Transactions on Networking 3 (Aug) (1995) 365–386.

[6] Andrea Francini, Fabio M. Chiussi, Minimum-latency dual-leaky-bucket shapers for packet multiplexers: theory and implementation, IEEE Workshop on QoS 2000.

[7] P. Goyal, H.M. Vin, H. Chen, Start-time fair queuing: a scheduling algorithm for integrated services, in: Proceedings of the ACM-SIGCOMM, Palo Alto, CA, 96 August 1996, pp. 157–168.

[8] S. Golestani, A stop-and-go queueing framework for congestion management, in: Proceedings of ACM SIGCOMM'90, Philadelphia, PA, September 1990, pp. 8–18.

[9] S. Golestani, A self-clocked fair queueing scheme for broadband applications, in: Proceedings of IEEE INFOCOM'94, Toronto, CA, January 1994, pp. 636–646.

[10] D. Hang, H.R. Shao, W. Zhu, Y.Q. Zhang, TD^{2}FQ: an integrated traffic scheduling and shaping scheme for DiffServ networks, in: Proceedings of IEEE HPSR 2001.

[11] J.F. Lee, Y. Sun, M.C. Chen, On maximum rate control of worst-case weighted fair queueing, Technical Report, Institute of Information Science, Academia Sinica, Taiwan ROC, TR-IIS-02-003, 2002.

[12] J.F. Lee, Y. Sun, M.C. Chen, WF2Q-M: a worst-case fair weighted fair queueing with maximum rate control, in: Proceedings of IEEE GLOBECOM 2002, Taiwan, vol.2, pp. 1576–1580.

[13] S. Lu, V. Bharghavan, R. Srikant, Fair scheduling in wireless packet networks, IEEE/ACM Transactions on Networking 7 (4) (1999).

[14] ns-2. Available from: <http://www.isi.edu/nsnam/ns/>.

[15] C. Kalmanek, H. Kanakia, S. Keshav, Rate controlled servers for very high-speed networks, in: IEEE Global Telecommunications Conference, San Diego, California, December 1990, pp. 300.3.1–300.3.9.

[16] A.K. Parekh, R.G. Gallager, A generalized processor sharing approach to flow control in integrated services networks: the single-node case, IEEE/ACM Transactions on Networking 1 (3) (1993) 344–357.

[17] J.L. Rexford, A.G. Greenberg, F.G. Bonomi, A. Wong, Scalable architectures for integrated traffic shaping and link scheduling in high-speed ATM switches, IEEE Journal on Selected Areas in Communications 15 (5) (1997) 938–950.

[18] I. Stoica, H. Abdel-Wahab, Earliest eligible virtual deadline first: a flexible and accurate mechanism for proportional share resource allocation, Technical Report TR-95-22, Old Dominion University, November 1995.

[19] D. Stiliadis, A. Varma, A general methodology for designing efficient traffic scheduling and shaping algorithms, in: Proceedings of IEEE INFOCOMM, 1997.

[20] A.S. Tanenbaum, Computer Networks, third ed., Prentice-Hall, Inc., 1996, pp. 380–381.

[21] D. Verma, H. Zhang, D. Ferrari, Guaranteeing delay jitter bounds in packet switching networks, in: Proceedings of Tricomm'91, Chapel Hill, North Carolina, 1991, pp. 35–46.

[22] H. Zhang, D. Ferrari, Rate-controlled static priority queueing, in: Proceedings of IEEE INFOCOM'93, San Francisco, California, 1993, pp. 227–236.

[23] H. Zhang, D. Ferrari, Rate-controlled service disciplines, Journal of High Speed Networks. 3 (4) (1994) 389–412.

[24] H. Zhang, Service disciplines for guaranteed performance service in packet-switching network, Proceedigs of the IEEE 83 (October) (1995) 1374–1396.

[25] K. Zhu, Y. Zhuang, Y. Viniotis, Achieving end-to-end delay bounds by EDF scheduling without traffic shaping, in: Proceedings of IEEE INFOCOM 2001.

Jeng Farn Lee received the B.S. and M.S. degrees in the Department of Information Management from National Taiwan University, Taiwan, in 1998 and 2000, respectively. Since then he has been with Institute of Information Science, Academia Sinica, Taiwan as a research assistant since January 2001 and Department of Electrical Engineering, National Taiwan University as a Ph.D. Student since June 2003. His current research interests include QoS networking, scheduling, and wireless access network.

Meng Chang Chen received the B.S. and M.S. degrees in Computer Science from National Chiao-Tung University, Taiwan, in 1979 and 1981, respectively, and the Ph.D. degree in Computer Science from the University of California, Los Angeles, in 1989. He joined AT&T Bell Labs in 1989 as Member of Technical Staff and led several projects in the area of data quality of distributed databases for mission critical systems. He was an Associate Professor at Department of Information Management, National Sun Yat-Sen University, Taiwan, from 1992 to 1993. Since then he has been with Institute of Information Science, Academia Sinica, Taiwan. He has held Associate Research Fellowship since July 1996, and assumed the responsibility of Deputy Director from August 1999 to July 2002. His current research interests include QoS networking, wireless network, multimedia systems and transmissions, operating systems, and data and knowledge engineering.

Yeali Sun received her B.S. degree in Computer Science from National Taiwan University, and the M.S. and Ph.D. degrees, both in Computer Science, from the University of California, Los Angeles (UCLA) in 1984 and 1988, respectively. From 1988 to 1993, she was with Bell Communications Research Inc. (Bellcore; now Telcordia), where she was involved in the area of planning and architecture design of information networking, broadband networks, and network and system management. In August 1993, she jointed National Taiwan University and is currently a professor of the Department of Information Management. Her research interests are in the area of architecture, protocols and implementation of Mobile Internet, Internet Quality of Service, high-performance content switch, and Internet pricing. She is also responsible for several National Experimental Projects, inclduing IPv6, Mobile wireless LAN and VoIP.