SCHEDULING FOR AGGREGATED MULTIPLE ASYMMETRICAL LINKS

by

RONY PATRICE PUTIH

B.A.Sc., The University of British Columbia, 1999

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

(Electrical and Computer Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

August 2005

ABSTRACT

Increase in demand for bandwidth has propelled the need for connecting two communication nodes with more than one physical link. Scheduling among these links must maintain Quality of Service (QoS) usually attained in the case of a single link connection. Previous single-server scheduling disciplines do not take into account properties inherent to multiple-server systems such as: packet re-ordering due to competition among packets of the same flow, and the ability to schedule more than one flow simultaneously due to division of total bandwidth among multiple servers / links. A multi-server system scheduling discipline, called the MSF^2Q (Multiple Server Fair Queueing with bounded Fairness)[1], exploited the bandwidth division property using scheduling eligibility constraints to achieve better throughput fairness. However, MSF^2Q assumes equal division of total bandwidth. In this thesis, we extend MSF^2Q to provide better fairness in throughput for multiple-server systems where each server is of a different (asymmetrical) rate. This new discipline, called $MASF^2Q$ (Multiple Asymmetrical Server Fair Queueing with bounded Fairness), achieves the same theoretical delay bound and per-flow service discrepancy as MSF^2Q. Simulation results are provided for performance comparison.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

a_k	arrival time of packet p_k
b_k	scheduling time of packet p_k under GPS
$b_{k,\ MASF^2Q}$	scheduling time of packet p_k under MASF^{2}Q
$b_{k,\ MSFQ}$	scheduling time of packet p_k under MSFQ
$b_{k,\ MSF^2Q}$	scheduling time of packet p_k under MSF^{2}Q
d_k	departure time of packet p_k under GPS
$d_{k,\ PGPS}$	departure time of packet p_k under PGPS
$d_{k,MASF^2Q}$	departure time of packet p_k under MASF^{2}Q
$d_{k,\ MSFQ}$	departure time of packet p_k under MSFQ
$d_{k,\ MSF^2Q}$	departure time of packet p_k under MSF^{2}Q
$\mathcal{F}$	set of flow indices
$\mathcal{F}(t,\tau)$	set of backlogged flow indices within time period $[t, \tau]$
L	length of a packet
L_k	length of packet p_k
$L_{i,max}$	maximum packet length of flow i
L_{max}	maximum packet length from all flows
M	total number of flows existing and sharing an aggregated multiple server system
N	total number of servers
$\hat{o}_i(t)$	number of outstanding flow i packets in the MSF^{2}Q system at time t
p_k	k^{th} packet scheduled under packetized or multiple server systems
r	rate of each individual server in the case multiple equal-rate server systems
r_{min}	rate of the slowest server in the case of multiple asymmetrical server systems
r_{max}	rate of the fastest server in the case of multiple asymmetrical server systems
r_{total}	total output rate of all servers in the case of multiple asymmetrical server systems
$r_i(t)$	instantaneous rate of a backlogged flow i in GPS system at time t
$\hat{r}_i$	current bandwidth allocated to a backlogged flow i in the packetized system
$r_{i,\ MSF^2Q}(t)$	instantaneous rate of flow i at MSF^{2}Q system at time t

$r_{i,\ MASF^2Q}(t)$	instantaneous rate of flow i at MASF^{2}Q system at time t
$W(t, \tau)$	total number of bits served by GPS within time period $[t, \tau]$
$W_{PGPS}(t, \tau)$	total number of bits served by PGPS within time period $[t, \tau]$
$W_{MASF^2Q}(t, \tau)$	total number of bits served by MASF^{2}Q within time period $[t, \tau]$
$W_{MSFQ}(t, \tau)$	total number of bits served by MSFQ within time period $[t, \tau]$
$W_{MSF^2Q}(t, \tau)$	total number of bits served by MSF^{2}Q within time period $[t, \tau]$
$W_i(t, \tau)$	total number of flow i bits served by GPS within time period $[t, \tau]$
$W_{i,\ PGPS}(t, \tau)$	total number of flow i bits served by PGPS within time period $[t, \tau]$
$W_{i,\ MASF^2Q}(t, \tau)$	total number of flow i bits served by MASF^{2}Q within time period $[t, \tau]$
$W_{i,\ MSFQ}(t, \tau)$	total number of flow i bits served by MSFQ within time period $[t, \tau]$
$W_{i,\ MSF^2Q}(t, \tau)$	total number of flow i bits served by MSF^{2}Q within time period $[t, \tau]$
ϕ_i	a positive real number representing weight of flow i

LIST OF NOMENCLATURES

$(GPS, 1, Nr)$	a single server system utilizing the GPS scheduling discipline with total server output rate of Nr bps
$(MSFQ, N, r)$	an N-server system, where each server operates at fixed rate r bps, utilizing the MSFQ scheduling discipline with total server output rate of Nr bps
(MSF^2Q, N, r)	an N-server system, where each server operates at fixed rate r bps, utilizing the MSF^2Q scheduling discipline with total server output rate of Nr bps
$(MSF^2Q, N, r_{min}, r_{max}, r_{total})$	an N-server system, where at least one server operates at a different fixed rate, utilizing the MSF^2Q scheduling discipline with total server output rate of r_{total} bps. Each server's fixed rate falls within r_{min} and r_{max} bps.
$(MASF^2Q, N, r_{min}, r_{max}, r_{total})$	an N-server system, where at least one server operates at a different fixed rate, utilizing the $MASF^2Q$ scheduling discipline with total server output rate of r_{total} bps. Each server's fixed rate falls within r_{min} and r_{max} bps.
Scheduler	a system component that determines and controls the allocation of network interface bandwidth to outgoing network flows.
Server	a first-in-first-out packet processor of a certain rate where only one packet served at a time.

LIST OF ABBREVIATIONS

bps	bits per second
DDRR	Device-Driven Round Robin
DRR	Deficit Round Robin
GPS	Generalized Processor Sharing
kb	kilobits
kbps	kilobits per second
MAS	Multiple Asymmetrical Server
$MASF^2Q$	Multiple Asymmetrical Server Fair Queueing with bounded Fairness (the scheduling discipline introduced in this thesis)
Mb	Megabits
Mbps	Megabits per second
MS	Multiple Server
MSFQ	Multiple Server Fair Queueing
MSF^2Q	Multiple Server Fair Queueing with bounded Fairness
NIC	Network Interface Card
QoS	Quality of Service
RR	Round Robin
(SCFQ	Self-Clocked Fair Queuing
TCP	Transfer Control Protocol
WFQ	Weighted Fair Queueing
WF^2Q	Worst-case Fair Weighted Fair Queueing

ACKNOWLEDGMENTS

I would like to thank my co-supervisor Dr. Cyril Leung and Dr. Vincent Wong for providing technical input and general guidance in my research. I would also like to acknowledge the help of other graduate students in Electrical Engineering who may have assisted me in my attempts at simulation. Finally, I want to thank my co-supervisors for providing input and advice during the preparation of the thesis.

This work was supported by Natural Sciences and Engineering Research Council of Canada (NSERC) grant #OGP0001731.

CHAPTER 1 INTRODUCTION

1.1 Resource Sharing

Continuing increase in the amount and variety of network traffic has given rise to the need for fair resource sharing of servers with guarantees. In communication networks, link scheduling is an example of such resource sharing where multiple flows (sessions) compete for access to a fixed amount of link bandwidth. The bandwidth can be simply from a single physical link, or from an aggregation of multiple physical links. In either case, each of the competing flows will have its own service requirements that have to be satisfied while simultaneously sharing the bandwidth. Depending on the type of traffic, typical service requirements include an acceptable level of throughput and packet delay from the network. The Internet's inherent best-effort traffic model alone is insufficient to handle the various traffic types such as streaming, real-time, rate-adaptive, etc. Thus, link scheduling is required in the network to ensure that the competing flows receive the minimum guaranteed service needed for proper performance.

In addition to service guarantees, link scheduling can also provide protection from misbehaving flows that monopolize resources at the expense of others and thus cause congestion. This protection allows fairer access to resources for all flows at all times and leads to better congestion control.

1.2 Key Performance Measures

Performance measures allow the comparison of different link scheduling algorithms relative to a reference scheduling model. It is important to denote the key measures before analyzing, comparing or introducing new scheduling schemes. Typical key measures include worst-case packet delay, maximum per-flow service (throughput) discrepancy, and *fairness* [1].

The worst-case packet delay refers to the extent the packet departure time of a scheduling scheme lags behind that of a reference scheduling model. The maximum per-flow service discrepancy denotes the extent the service of a flow under a scheduling scheme lags

behind that of the corresponding flow under a reference scheduling model. It is desirable to have bounds for both these measures. Better approximation of a reference model should produce smaller bounds for one or both of them.

This thesis will focus on an important performance measure called fairness. Fairness refers to the extent the service of a flow under a scheduling scheme *leads* that of the corresponding flow under a reference scheduling model. A desirable property, called *bounded fairness* [1], means that the maximum bound on fairness for any flow is independent of the set of competing flows. Bounded fairness can prevent a scheduling scheme from favoring certain flows, and, thus, provide smoother (less bursty) overall output for all flows at a finer time scale [1]. A scheduling scheme that better approximates a reference model should have smaller bound on fairness and smoother output. Fairness is directly related to how fairly (relative to a reference model) a link scheduling scheme can distribute the total shared bandwidth among all flows. Thus, we can measure the smoothness of output in simulation by calculating the differences in flows' service rates among different scheduling schemes relative to a reference model.

1.3 Fair Queuing Service Disciplines

Fair Queuing [2] service disciplines are scheduling schemes that address the problem of network resource sharing by proportional allocation of bandwidth among competing flows in order to fulfill each of their service requirements.

The Generalized Processor Sharing (GPS) [3] service discipline is an idealized, work-conserving fair queuing discipline for single-server systems where bandwidth is assumed to be infinitely divisible to accommodate the throughput requirement of each flow. Thus, GPS cannot be implemented and is commonly used as a theoretical reference model because of its many desirable properties. One such property is the ability of GPS to provide service guarantee in the form of end-to-end bounded-delay to a flow whose traffic is constrained by a leaky bucket. This property provides the basis of provisioning guaranteed service traffic.

Many packet-based versions of GPS have been proposed over the years. These packetized versions can be implemented since the assumption is that a server can only serve one packet at a time. One of the early packetized versions, aptly named *packet-by-packet* GPS (PGPS), was introduced by Parekh and Gallager [3]. At the same time, they also proposed an

implementation of PGPS called Virtual Time implementation of PGPS. They used the concept of virtual time to track the progress of GPS. This implementation allows PGPS to closely approximate GPS by scheduling packets in a non-decreasing order of virtual time finishing (departure) times. PGPS is often referred to also as Weighted Fair Queuing (WFQ) [2]. Some other fair queuing disciplines that have been studied in the last decade include Worst-case Fair Weighted Fair Queuing [4], Self-Clocked Fair Queuing [5], Virtual Clock [6], and Deficit Round Robin [7].

Although Fair Queuing disciplines allow single-server sharing, they do not address the sharing of aggregated multiple servers. Multiple-server (MS) systems arise in areas of link aggregation, multiprocessor systems, etc. In scenarios such as bandwidth-on-demand, link aggregation allows flexible incremental scaling of bandwidth. Numerous aggregation techniques are available [8, 9, 10, 11], but they do not address the provision of QoS over the aggregated links.

An initial analysis of Fair Queuing on MS systems was provided by Blanquer and Ozden [1]. They showed that direct application of single server-based service disciplines such as PGPS is not suitable for MS systems. Specifically, these service disciplines do not take into account characteristics pertaining only to MS systems, and, so, they do not provide the desired smooth output and bounded fairness properties. In order to achieve these properties, Blanquer *et al.* proposed a new scheme called MSF^2Q (Multiple Server Fair Queueing with bounded Fairness) service discipline based on MS systems where each server is of equal rate. MSF^2Q is a non work-conserving modification of PGPS by the inclusion of additional constraints that decide which backlogged flows are eligible for scheduling at any time t. A flow is *backlogged* when it has packets in transmission or queued for transmission. Another recent paper that also analyzed multi-channel schedulers provides a means of transforming well-established single-channel service disciplines for use on multi-channel scheduler cases [12].

The main contribution of this thesis is the extension of the MSF^2Q service discipline so that it provides fairness property for Multiple Asymmetrical Server (MAS) systems where at least one server has a different rate. We call this extended service discipline $MASF^2Q$ (Multiple Asymmetrical Server Fair Queueing with bounded Fairness). Such MAS systems arise in link scheduling where not all the links are of the same rates. The original MSF^2Q assumes that all servers are of equal rate. Upon selection of a packet from an eligible flow, MSF^2Q may simply schedule the packet on any one of the available idle servers since there is

no preference. In MAS systems, MSF^2Q lacks the additional intelligence on deciding which of the available idle servers is best suited to serve a particular packet at a given time. We denote this additional ability as *packet-server pairing* decision. $MASF^2Q$ is a modification of MSF^2Q that includes additional constraints that provide packet-server pairings decision. These constraints preserve the desirable properties already achieved using MSF^2Q, but should also provide smoother output in MAS systems. $MASF^2Q$ is non work-conserving as well.

This thesis will consider worst-case packet delay, per-flow service discrepancy, and fairness for $MASF^2Q$ by looking into theoretical scenarios that will generate worst case bounds. These scenarios represent snap-shots of possible states at queuing points in a communication network.

The remainder of the thesis is organized as follows. Chapter 2 discusses related work in single server and multiple server service disciplines. Section 2.1 provides an overview of GPS and PGPS and their properties. A brief discussion of idealized and packetized service disciplines is provided. Other single-server scheduling disciplines are also discussed. Section 2.2 focuses on the properties of MS systems, and includes a brief discussion of direct implementation of PGPS on MS systems. Discussion on MSF^2Q follows focusing on the constraints that determine scheduling eligibility of backlogged flows. Another multi-channel scheduling approach by transforming a single-server scheduling discipline is discussed. Chapter 3 describes the original contribution of this thesis. This chapter introduces the new $MASF^2Q$ service discipline and its packet-server pairing idea. Derivation of theoretical bounds for worst-case packet delay, per-flow service discrepancy, and fairness are provided for MSF^2Q in MAS system. Simulation results and discussion related to fairness property are provided in Chapter 4. Chapter 5 presents the conclusions and provides recommendations for future work.

CHAPTER 2 RELATED WORK

2.1 Single Server Service Disciplines

2.1.1 GPS Service Discipline

Generalized Processor Sharing (GPS) is a Fair Queuing service discipline that addresses the proportional sharing of a single server among a set of competing flows [3]. As an idealized discipline, GPS cannot be implemented because it assumes that a server's bandwidth can be infinitely divisible. A GPS server operates at a fixed rate of r and is *work-conserving*. As a work-conserving service discipline, a GPS server is always busy whenever there are backlogged flows. A flow is considered *backlogged* when it has packets being transmitted or queued for transmission by the server – otherwise, a flow is *idle*.

GPS provides bandwidth guarantee to each competing flow i by assigning a positive real number called *weight*, ϕ_i. This weight determines the amount of service a flow i will receive within a certain time interval when the flow is continuously backlogged. Thus, for any flow i that is continuously backlogged within time interval $[\tau, t]$, a GPS server is defined as one which satisfies

$$\frac{W_i(\tau,t)}{W_j(\tau,t)} \geq \frac{\phi_i}{\phi_j}, \quad j \in \mathcal{F} \tag{2.1}$$

where

$W_i(\tau, t)$ = the amount of flow i traffic served by GPS server within time interval $[\tau, t]$

$\mathcal{F}$ = set of flow indices.

Within any time interval $[\tau, t]$ when the set of backlogged flows is unchanged, denoted by $\mathcal{F}(\tau, t)$, a GPS server also guarantees an *instantaneous rate* for flow i of

$$r_i(t) = \frac{\phi_i}{\sum_{j \in \mathcal{F}(\tau,t)} \phi_j} \cdot r, \quad i \in \mathcal{F}(\tau,t) \tag{2.2}$$

Based on the above formulation of flow rate, a desirable property of GPS is that it only provides guarantee on *minimum* bandwidth for a flow i. GPS can potentially provide flow i with full bandwidth of r if all other flows are idle. This property allows protection from misbehaving flows and efficient use of bandwidth. The minimum guaranteed service bandwidth for flow i is

$$r_i = \frac{\phi_i}{\sum_{j \in \mathcal{F}} \phi_j} \cdot r \ , \quad i \in \mathcal{F} \tag{2.3}$$

Another desirable property is the ability of GPS to provide service guarantee in the form of *end-to-end bounded-delay* to a flow i whose traffic is constrained by a leaky bucket [3]. This is important because any other service discipline that possesses a bound on its worst-case packet delay relative to that of GPS automatically shares the end-to-end bounded-delay property for similarly constrained flow traffic.

In this thesis, we denote a GPS service discipline with a single server operating at total rate of r as (*GPS,* 1, r).

2.1.2 PGPS Service Discipline

Packet-by-packet GPS (PGPS) is a packetized version of GPS [3]. PGPS possesses the same work-conserving nature as GPS. As a result, the busy periods of both GPS and PGPS coincide, i.e., the GPS server is busy if and only if the PGPS server is busy. Furthermore, being a packetized discipline allows PGPS to be implemented. A popular implementation of PGPS, called Virtual Time implementation, was introduced by Parekh and Gallager. This implementation allows PGPS to closely approximate GPS by scheduling packets in a non-decreasing order of virtual time finishing times. This essentially means that, at time τ when a PGPS server is idle, it picks from all backlogged flows the first packet that would complete service under GPS if no additional packets were to arrive after time τ.

One important property of PGPS is that a server cannot be both work-conserving and serve packets in the increasing order of GPS packet departure times. It is possible that at time τ when the PGPS server is idle, the next packet to depart under GPS has not yet arrived. These *late-arrival* packets tend to be of small sizes [3], and they may also cause a *re-ordering* of

PGPS packet departure relative to that of GPS. Parekh and Gallager showed that the PGPS *worst-case packet delay* satisfies

$$d_{k,\,PGPS} - d_k \leq \frac{L_{\max}}{r} \tag{2.4}$$

where

d_k = departure time of packet p_k under GPS

$d_{k,\,PGPS}$ = departure time of packet p_k under PGPS

L_{max} = maximum packet length from all flows

r = rate of PGPS server.

Using the worst-case packet delay, Parekh and Gallager derived the PGPS *maximum per-flow service discrepancy* relative to GPS [3]. The bound on this discrepancy satisfies

$$W_i(0,\tau) - W_{i,\,PGPS}(0,\tau) \leq L_{\max} \tag{2.5}$$

where

$W_i(0, \tau)$ = amount of flow i bits served by GPS within time period $[0, \tau]$

$W_{i,\,PGPS}(0, \tau)$ = amount of flow i bits served by PGPS within time period $[0, \tau]$.

The inequalities (2.4) and (2.5) allow us to readily translate bounds on GPS worst-case packet delay and backlog to PGPS. For instance, the per-flow service discrepancy bound can be used to calculate the minimum PGPS buffer requirements if GPS buffer requirements are known. As mentioned in Section 2.1.1, the worst-case packet delay basically provides the maximum additional network delay imposed on a particular packet by PGPS relative to GPS.

Despite closely approximating GPS, PGPS may produce *bursty* traffic patterns. Bursty traffic is caused by PGPS favoring a certain flow i during scheduling. This can cause its service of flow i to *arbitrarily* exceed that under GPS – effectively allowing packets of the flow to leave far ahead in PGPS than in GPS. PGPS maximum excess amount of service for all flows is not upper-bounded by a constant; i.e., PGPS does not possess *bounded fairness* property [1] [3]. Instead, the upper bound is a function of the set of existing flows which may change. Parekh and Gallager showed that, for *all* flow i, there is *no* constant $c \geq 0$ such that

$$W_{i,\,PGPS}(0,\tau) - W_i(0,\tau) \leq c \cdot L_{\max} \tag{2.6}$$

2.1.3 A Variation of PGPS: WF^{2}Q

Bennet and Zhang proposed a service discipline that tackles the burstiness of PGPS traffic [4]. They introduced a constraint that determines the *scheduling eligibility* of flows. Of all backlogged flows waiting for service by an idle PGPS server at time τ, this constraint restricts eligibility to only those packets that have already started or finished service in GPS. This condition effectively excludes certain backlogged flows that would have been considered by PGPS in the first place. Among the packets eligible at time τ, the next packet to schedule is still the one that would complete service first in the corresponding GPS system. This new discipline by Bennet and Zhang is called *Worst-case Fair* WFQ, or WF^{2}Q.

WF^{2}Q provides a fairness bound for each individual flow that leads to smoother traffic compared to GPS. The WF^{2}Q fairness bound for a flow i is

$$W_{i,\,WF^2Q}(0,\tau) \;-\; W_i(0,\tau) \;\le\; (1-\frac{r_i}{r})\cdot L_{i,\,\max} \qquad (2.7)$$

where:

$W_{i,WF^2Q}(0,\ \tau)$ = amount of flow i bits served by WF^{2}Q within time period $[0,\ \tau]$

$L_{i,max}$ = maximum packet length of flow i

r_i = minimum guaranteed service rate for flow i (as defined in GPS).

Upon closer inspection, for *any* flow, the inequality (2.7) has a theoretical maximum bound of L_{max} when the minimum guaranteed service rate for the flow becomes very small compared to the total system rate of r. WF^{2}Q essentially provides bounded fairness property for single-server systems.

2.1.4 A Comparison of GPS, PGPS, and WF^{2}Q

This section discusses fairness of single-server service disciplines relative to GPS. We borrowed a scenario from the paper by Bennet *et al.* [4]. This scenario considers 11 flows, F1 to F11, sharing the same link of rate r = 1 bps. We assume that all packets considered have the same size of 1 bit. Flow F1 has a weight of 0.5, while the other 10 flows have a weight of 0.05 each. At time t = 0 when the server is idle, we assume that flow F1 has 10 packets queued and each of the other 10 flows has 1 packet (Figure 2.1).

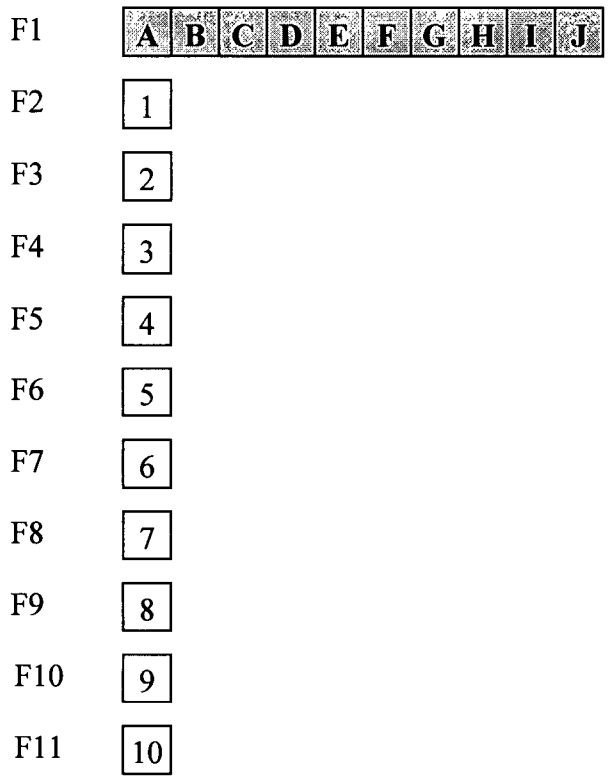


Figure 2.1: Packets arrived and queued at time $t = 0$

Because all flows are backlogged at time $t = 0$, GPS server will start scheduling them simultaneously (Figure 2.2). Note that in Figure 2.2, the horizontal axis represents time while the vertical axis represents the bandwidth of the GPS server. Since the sum of all flow weights is 1, flow F1 will be guaranteed at least half the total rate for as long as it is backlogged. Each of the other 10 flows will share the remaining bandwidth equally – resulting in at least 0.05 bps each. This works out to one packet from each of flows F2 to F11 receiving service simultaneously for 20 seconds. Within that same period, the GPS server is able to service 10 packets of flow F1. In this scenario, the busy period of GPS server starts at $t = 0$ and ends at $t = 20$ seconds. If the same arrival pattern occurs repeatedly, we will see the same GPS service pattern as the one shown on Figure 2.2 every 20-second interval.

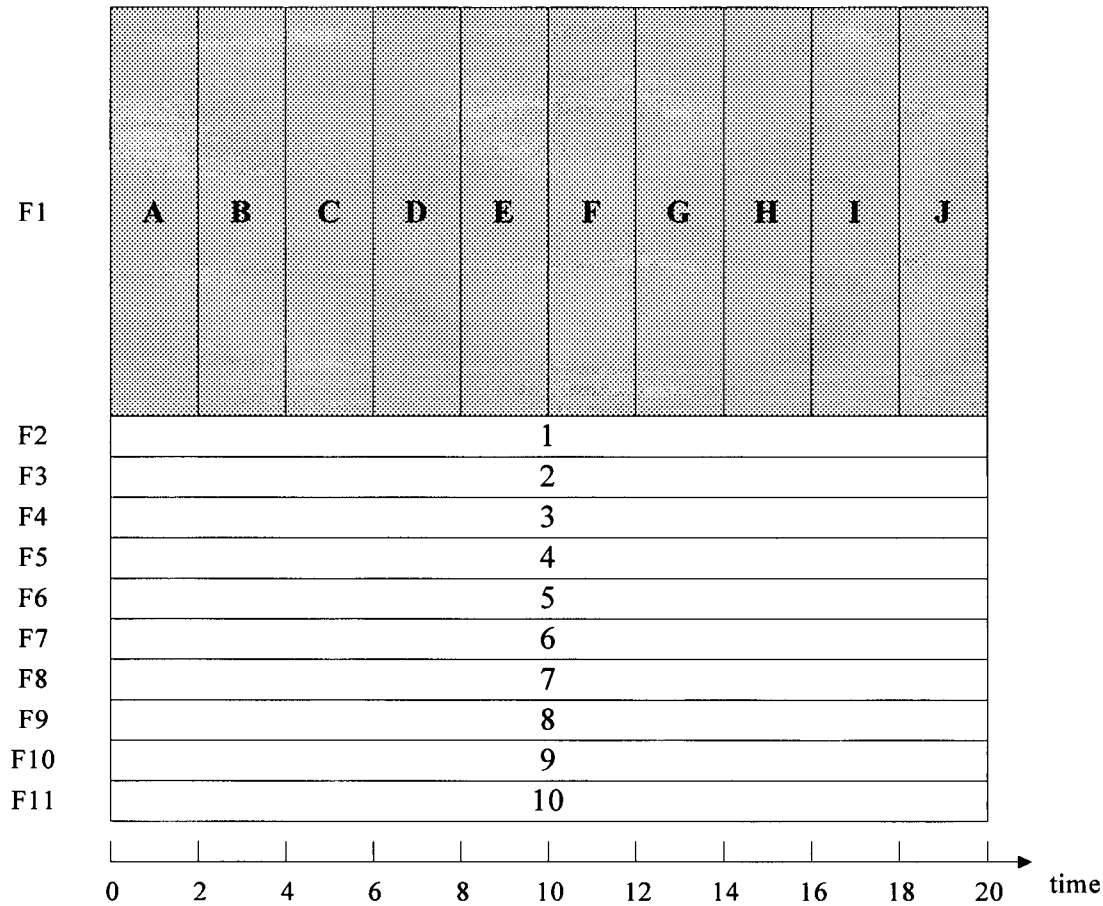


Figure 2.2: GPS packet scheduling and service order

PGPS can be implemented to approximate the above ideal scenario. Being both work-conserving, the busy periods of GPS and PGPS still coincide. As Figure 2.2 shows, packets A to I of flow F1 finish earlier than the rest of other packets in GPS. Thus, PGPS will serve packets A to I using full server bandwidth before all the others (Figure 2.3). When packet I finishes service at time $t = 9$, PGPS can start serving packet J, or packets 1 to 10 in arbitrary order since they all finish at time $t = 20$ in GPS. Without violating the PGPS rule, we are going to assume that the server schedules packet J at time $t = 9$ before packets 1 to 10. For the first 10 seconds, it is clear that PGPS favors flow F1 by servicing all its packets. Flow F1 enjoys full bandwidth during that period, and receives none afterwards – creating bursty traffic. If the same arrival pattern occurs repeatedly, flow F1 will experience bursty traffic every 20-second interval with the time lag between two consecutive busy periods as large as 10 seconds.

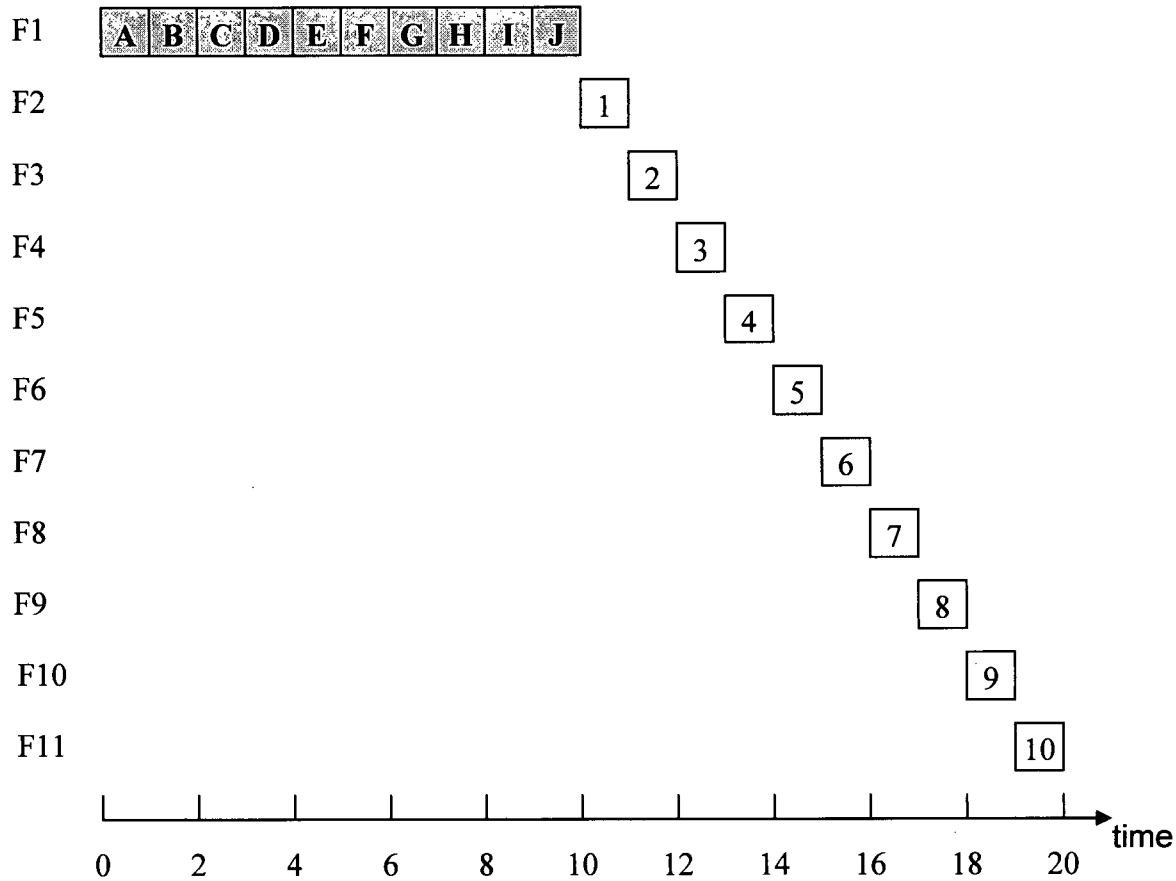


Figure 2.3: PGPS packet scheduling and service order

WF^2Q allocates the bandwidth more fairly at a finer time scale and solves the bursty traffic problem of flow F1. As shown in Figure 2.4, WF^2Q serves one packet of flow F1 every other second starting from time $t = 0$. Even when the same arrival pattern occurs repeatedly, the link bandwidth is made available to each packet of flow F1 without extensive time lag between two consecutive busy periods. In this particular example, the scheduling eligibility constraint of WF^2Q effectively permits the server to serve packets from other flows between every two packets of flow F1. Being a work-conserving discipline, the busy periods of WF^2Q also coincide with that of GPS.

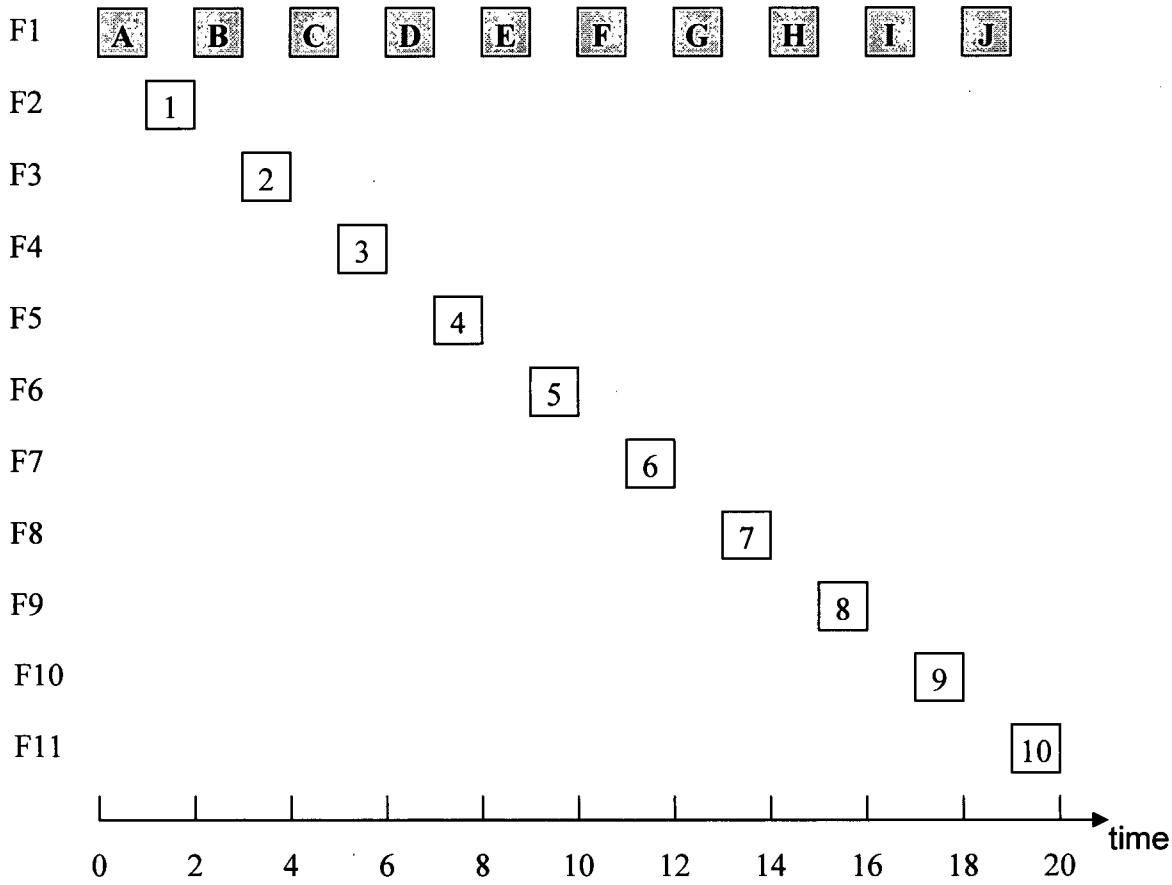


Figure 2.4: WF^2Q packet scheduling and service order

2.1.5 Other Single Server Service Disciplines

Besides PGPS and its variants, many other single-server based fair queuing disciplines have been proposed over the last decade. Many of them still try to closely approximate GPS system in terms of delay and throughput fairness. Many of the newer techniques attempt to also reduce the complexity needed to implement the packetized service compared to that of WFQ. Most of the time these attempts result in a trade-off between how close an approximation of ideal system and the simplicity to implement the scheme. We will briefly discuss a few of these schemes and provide references for further study.

One such scheme is called the Self-Clocked Fair Queuing (SCFQ) proposed by Golestani [5]. Similar to PGPS, it also uses the notion of virtual time. One very different aspect is in the definition of virtual time. While the virtual times of PGPS and WF^2Q are

directly related to the service done in GPS, the virtual time of SCFQ is calculated from the packetized system itself. So, there is no need to do GPS emulation in the background just to produce and keep track of virtual time. Each arriving packet in SCFQ is still stamped with a virtual time (also called service tags) before being enqueued. Service tags represent the virtual time at which the packets must already be serviced. Similar to PGPS, packets are serviced in order of the increasing service tags. The mathematics and implementation of updating the virtual time is similar to PGPS itself, except that generation of virtual time is simple and only involves the extraction of service tag of packet in service. Although SCFQ reduces implementation complexity, it has a large delay bound compared to GPS since its definition of virtual time is no longer tracking the service done by GPS.

Another algorithm is the Virtual Clock algorithm proposed by Zhang [6]. It is based on using a virtual clock of the server to assign a tag or deadline to each packet. This tag represents the clock value at which the packet must be transmitted. Each arriving packet is assigned a tag before being enqueued. Packets are serviced by a work-conserving server in increasing order of tags. In this algorithm, the notion of virtual clock is used as a counter that keeps track of service done by the server. So, each packet of a certain length served will increment the clock by a specific amount. The mathematics of updating the tags is also similar to that of other virtual time schemes. It involves adding time needed to service the packet at its guaranteed rate to the maximum of either the tag of previous packet (of the same flow) or the current value of server clock. The implementation is simpler because it does not emulate GPS, but Virtual Clock algorithm results in poor fairness property.

Another algorithm is called the Deficit Round Robin (DRR) [7]. It is based on the basic round-robin scheme [13] that services flows in turns or rounds. In every round, each flow transmits one packet. Basic round-robin scheme is fair if each flow has the same average packet length and bandwidth requirement. DRR is an improvement of round-robin and it can handle networks with flows having different average packet lengths. DRR introduced a so-called quantum of service that defines the additional quota made available to a flow within a certain round. DRR allows any unused portion of a quantum within a round (due to the size of the next packet being too large to send) to carry forward to the next round. This carry-forward allows the flow to catch up to other flows the next time it has the opportunity to send again. In order to use the carry-forward, DRR must keep state per flow that tracks the deficit of each flow. With no virtual time calculation, the enqueuing process can be constant in terms of

computation needed. However, in order to maintain constant complexity for the whole system, the quantum size has to be selected properly and has to be large enough to allow at least one packet served per flow per round. Despite achievement in complexity reduction, this scheme causes large delay compared to GPS.

2.2 Multiple-Server Service Disciplines

2.2.1 Multiple-Server Systems

This chapter provides an overview of fair queuing for aggregated multiple-server systems. Aggregated multiple-server (MS) system denotes a system where M competing flows share a collection (or aggregation) of N multiple *physical* servers. Each of the N physical servers is assumed to provide the same fixed rate of r bps – resulting in a total system bandwidth of Nr bps.

Blanquer and Ozden studied the case where packets from each of the flows access the servers through one *logical* interface which is the packetized scheduler (Figure 2.5) [1]. Using one scheduler allows efficient utilization of full system bandwidth, and prevents the assignment of a flow with bandwidth share higher than that of any single server.

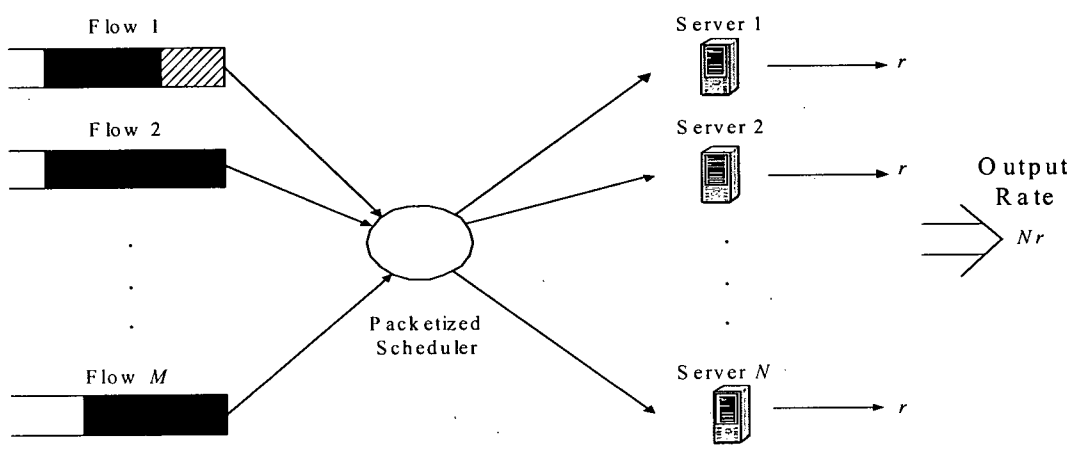


Figure 2.5: Packetized multiple-server system model

As with single-server systems, a reference model is required for comparison study. Figure 2.6 shows the reference model as an ideal GPS scheduler with a single server operating

at a rate of Nr bps – referred to as (*GPS*, 1, Nr) [1]. As far as the competing flows are concerned, both systems have the same output rate of Nr bps. Using GPS reference model allows performance comparison of MS scheduling schemes in this chapter with previously mentioned single-server service disciplines.

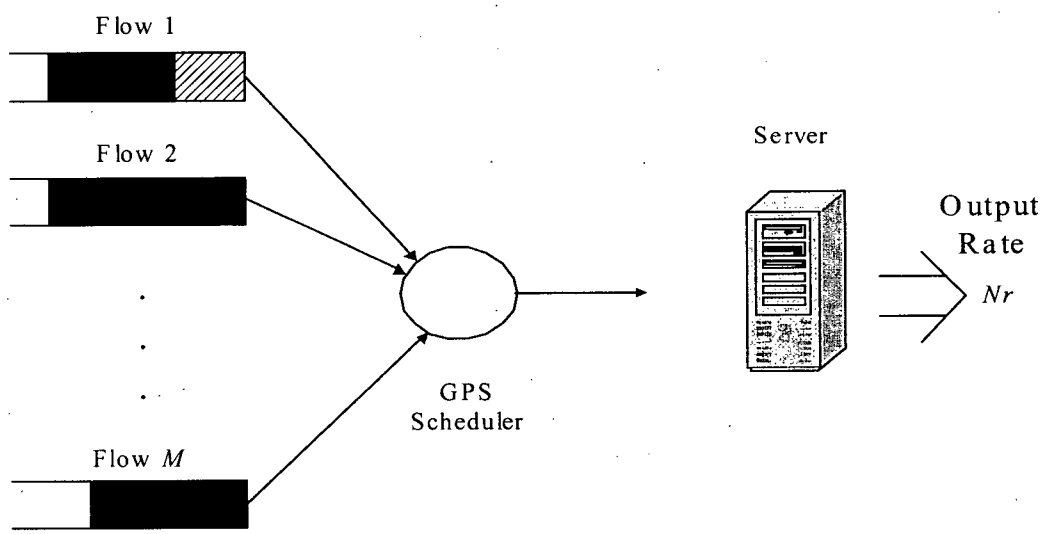


Figure 2.6: GPS model for multiple-server system

2.2.2 MSFQ Service Discipline

A first attempt at handling MS systems was to directly apply PGPS service discipline originally meant for single-server systems. Blanquer *et al.* did just that and referred to it as MSFQ service discipline [1]. An MSFQ scheduler with N servers, each operating at a fixed rate of r bps, is referred to as (*MSFQ*, N, r).

The study of MSFQ reveals some of the important properties inherent only to MS systems. One such property relates to the busy periods of MSFQ and GPS. Although both disciplines are work-conserving, their busy periods do not coincide. While MSFQ is always busy when GPS is busy, the converse is no longer true [1]. The reason is that, whenever at least one flow is backlogged, GPS always operates at full rate of Nr bps while MSFQ may use only some of the servers resulting in a rate less than Nr bps. This discrepancy can cause *service backlogs* for packetized MS systems relative to GPS. An example of this scenario is when both GPS and MSFQ are serving only one backlogged packet of length L at time $t = 0$.

GPS serves the packet at a rate of Nr bps and finishes at $L / (Nr)$ seconds. MSFQ, however, can schedule the packet on only one of the servers and has to serve the packets in L / r seconds. Blanquer *et al.* derived a bound for the service backlog for any time τ as follows:

$$W(0,\tau) - W_{MSFQ}(0,\tau) \leq (N-1) \cdot L_{\max} \tag{2.8}$$

where

$W(0, \tau)$ = amount of bits served by GPS within time period $[0, \tau]$

$W_{\mathrm{MSFQ}}(0, \tau)$ = amount of bits served by MSFQ within time period $[0, \tau]$

L_{max} = maximum packet length from all flows

N = number of physical servers in the MS system.

Another important property of MS systems is that packets may not depart MSFQ server in the same order as they are scheduled [1]. One reason is competition for service among packets of the same flow. For single-server systems, packets of the same flow are served in consecutive order *one at a time.* When there are more than one idle servers available at time τ, however, MSFQ may be able to schedule *simultaneously* more than one packets of a flow. This can happen when at least two packets of that flow are eligible to schedule at time τ. An example of this scenario is to assume p_k and p_{k+1} as the next two packets to schedule belonging to the same flow. At time τ, MSFQ will schedule packet p_k before packet p_{k+1} based on the increasing order of their GPS departure times. However, if the length of packet p_{k+1} is shorter than that of p_k, packet p_{k+1} will depart the MSFQ system before p_k since they are both served at the same rate of r bps. This kind of *packet reordering* does not happen in single-server systems.

Based on the above properties, Blanquer *et al.* derived bounds for both MSFQ worst-case packet delay and maximum per-flow service discrepancy [1]. For all packets p_k, the packet departure times satisfy

$$d_{k,\,MSFQ} - d_k \leq \frac{(N-1) \cdot L_k}{N \cdot r} + \frac{L_{\max}}{r} \tag{2.9}$$

where

d_k = departure time of packet p_k under GPS

$d_{k,\,MSFQ}$ = departure time of packet p_k under MSFQ

L_k = length of packet p_k.

For any flow i and time τ, MSFQ per-flow service discrepancy satisfies

$$W_i(0,\tau) - W_{i,\,MSFQ}(0,\tau) \leq N \cdot L_{\max} \tag{2.10}$$

where

$W_i(0, \tau)$ = amount of flow i bits served by GPS within time period $[0, \tau]$

$W_{i,\,MSFQ}(0, \tau)$ = amount of flow i bits served by MSFQ within time period $[0, \tau]$.

Like PGPS for single-server systems, MSFQ does not possess the bounded fairness property. This means that there is no constant $c \geq 0$ that satisfies the equivalent of equation (2.6) for MSFQ [1]. Thus, MSFQ can also favor a certain flow i during scheduling and produce bursty traffic pattern.

2.2.3 MSF^{2}Q Service Discipline

Applying WF^{2}Q directly to MS systems does not solve the bursty traffic problem. Instead, it turns the MS system into a non work-conserving version of MSFQ [1]. Blanquer *et al.* introduced a new scheduling scheme called MSF^{2}Q. MSF^{2}Q is simply MSFQ plus a new set of additional constraints that determine which of the backlogged flows are eligible to schedule when a server becomes available. Using the same naming format, we denote this scheme as (MSF^2Q, N, r). This new service discipline is non work-conserving, but it provides bounded fairness and smoother traffic while maintaining the same MSFQ bounds on packet delay and per-flow service discrepancy [1].

At time τ, when one or more servers become idle, MSF^{2}Q will schedule packets from only those backlogged flows that satisfy *either* one of the following two constraints:

$$W_{i,\,MSF^2Q}(0,\tau) < W_i(0,\tau) \tag{2.11}$$

or

$$W_{i,\,MSF^2Q}(0,\tau) = W_i(0,\tau) \text{ and } \hat{o}_i(\tau) < \left\lceil \frac{r_i(\tau)}{r} \right\rceil \tag{2.12}$$

where $\hat{o}_i(\tau)$ denotes the number of *outstanding* flow i packets at MSF^{2}Q system at time τ. A packet is outstanding at time τ if it is being transmitted or selected for transmission by the packetized system. In equation (2.12), $r_i(\tau)$ denotes the instantaneous *GPS* service rate of flow i at time τ as defined by equation (2.2). Packets among eligible backlogged flows are still scheduled based on the increasing order of GPS packet departure times.

At time τ, the *first constraint* allows scheduling of any flow whose service under MSF^{2}Q is lagging behind its corresponding flow under GPS. Except for the number of available idle servers, there is no limit to how many packets of the flow MSF^{2}Q can schedule at time τ. This constraint allows a flow to catch up to, and possibly exceed, its GPS service as soon as possible. A flow may actually schedule so many packets such that it temporarily occupies more bandwidth under MSF^{2}Q than under GPS. The next time one or more servers become idle, the service of this flow may have exceeded its GPS service. This will render the flow temporarily ineligible to schedule under MSF^{2}Q until GPS catches up or leads again. This leading and lagging in service of the flow can happen repeatedly.

At time τ, the *second constraint* allows scheduling of any flow whose service under MSF^{2}Q and GPS is the same, *and* that has not achieved at least its instantaneous *GPS* service rate, $r_i(\tau)$, under MSF^{2}Q. Essentially, the flow only tries to match its share of GPS bandwidth and possibly gets slightly ahead of GPS. Due to the ceiling function in $\hat{o}_i(\tau)$, a flow under MSF^{2}Q may actually schedule enough packets at time τ such that it temporarily occupies slightly more bandwidth than that under GPS. Thus, MSF^{2}Q flow may potentially exceed its GPS service and become ineligible to schedule until it again satisfies either one of the two eligibility constraints above.

The bounded fairness property of MSF^{2}Q is based on a flow's ineligibility to schedule. This ineligibility allows MSF^{2}Q to prevent a flow from arbitrarily exceeding its service in GPS. In their paper [1], Blanquer *et al.* derived the bounded fairness property as

$$W_{i,\,MSF^2Q}(0,\tau) - W_i(0,\tau) \le N \cdot L_{i,\max} \qquad (2.13)$$

where $L_{i,max}$ is the maximum packet length of flow i.

2.2.4 A Comparison of MSFQ and MSF^{2}Q

This section discusses fairness comparison of MS service disciplines relative to the GPS reference model (*GPS*, 1, *Nr*). We borrow the scenario used by Blanquer *et al.* [1] This is the same scenario used in Section 2.1.4, except that the flows are now sharing 4 output servers. Each server is operating at r = 0.25 bps for a system output rate of Nr = 1 bps. The same system output rate allows comparison of MS service disciplines with GPS and with the other single-server schemes. At time t = 0, all servers are assumed to be idle.

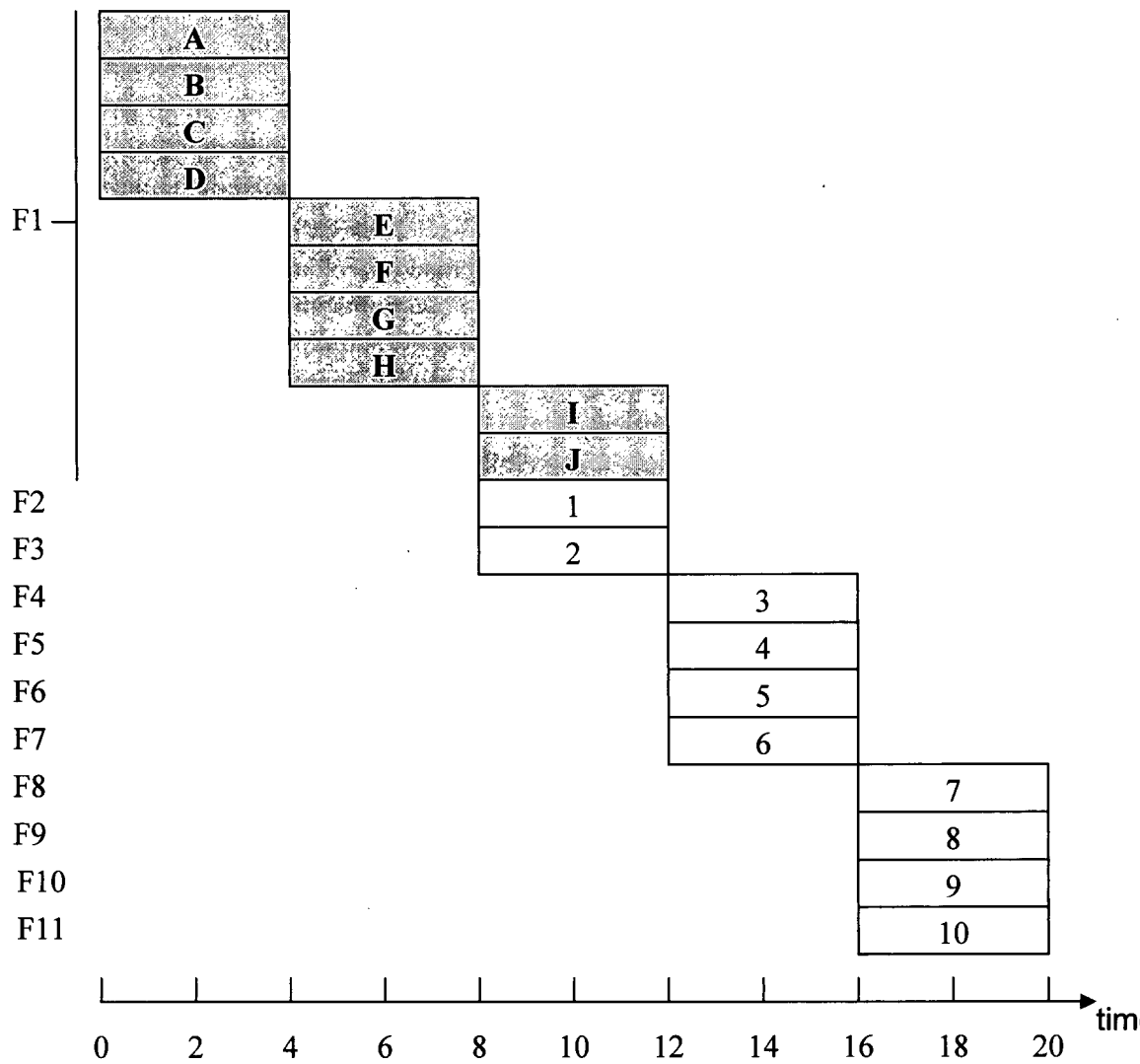


Figure 2.7: MSFQ scheduling for multiple-server system

Based on the increasing GPS packet departure times defined in equation (2.2), MSFQ will schedule all packets of flow F1 before those of other flows (Figure 2.7). Since there are 4 idle servers at time $t = 0$, packets A, B, C, and D of flow F1 will be scheduled simultaneously. It takes 4 seconds for each server to serve a packet of length 1 bit. At time $t = 4$, all servers will become idle and will be available to serve packets E, F, G, and H. At time $t = 8$ when all servers will again become idle, there are only 2 more packets of flow F1 queued. Only 2 servers will serve flow F1 (packets I and J), while others will start serving flows F2 and F3. Starting from time $t = 12$, it takes another 8 seconds to serve all packets of the remaining flows. It is obvious that MSFQ favors flow F1 by serving all of its packets first and causes bursty traffic pattern. Although all packets in the system are served within the same 20-second period as GPS (Figure 2.2), busy periods of MSFQ and GPS servers do not generally coincide.

Before scheduling any packets, MSF^2Q has to determine scheduling eligibility of backlogged flows (Figure 2.8). At time $t = 0$, all flows have zero service under both GPS and MSF^2Q, and none of the flows have packets scheduled yet. Thus, they are all eligible to schedule based on the second constraint defined in equation (2.12). Since flow F1 has a weight of 0.5, its instantaneous GPS service rate at time $t = 0$ is $r_1(t = 0) = 0.5$ bps (Figure 2.2). Under MSF^2Q, this translates into flow F1 being eligible to have at most 2 packets outstanding at time $t = 0$ before the flow finally violates the constraint and becomes ineligible. With other flows having a weight of 0.05 each, there can be at most 1 packet per flow outstanding at time $t = 0$. Thus, based on the increasing GPS departure times, packets 1 and 2 are scheduled following packets A and B. At time $t = 4$, these packets will finish service and all servers become idle. Flow F1 has the same amount of service under both MSF^2Q and GPS, and satisfies the second constraint once more. Flow F1 can again have at most 2 packets outstanding since $r_1(t = 4) = 0.5$ bps. On the other hand, flows F4 and F5 under MSF^2Q are lagging their GPS service and satisfy the first constraint defined in equation (2.11). These two flows can have as many packets outstanding as there are available idle servers. Thus, based on the increasing GPS departure times, packets 3 and 4 are scheduled following packets C and D at time $t = 4$. This process will repeat with the rest of the packets until all of them finish service at time $t = 20$. The resulting traffic of flow F1 thus becomes much smoother than that of MSFQ. Being a non work-conserving discipline, the busy periods of MSF^2Q do not generally coincide with those of GPS.

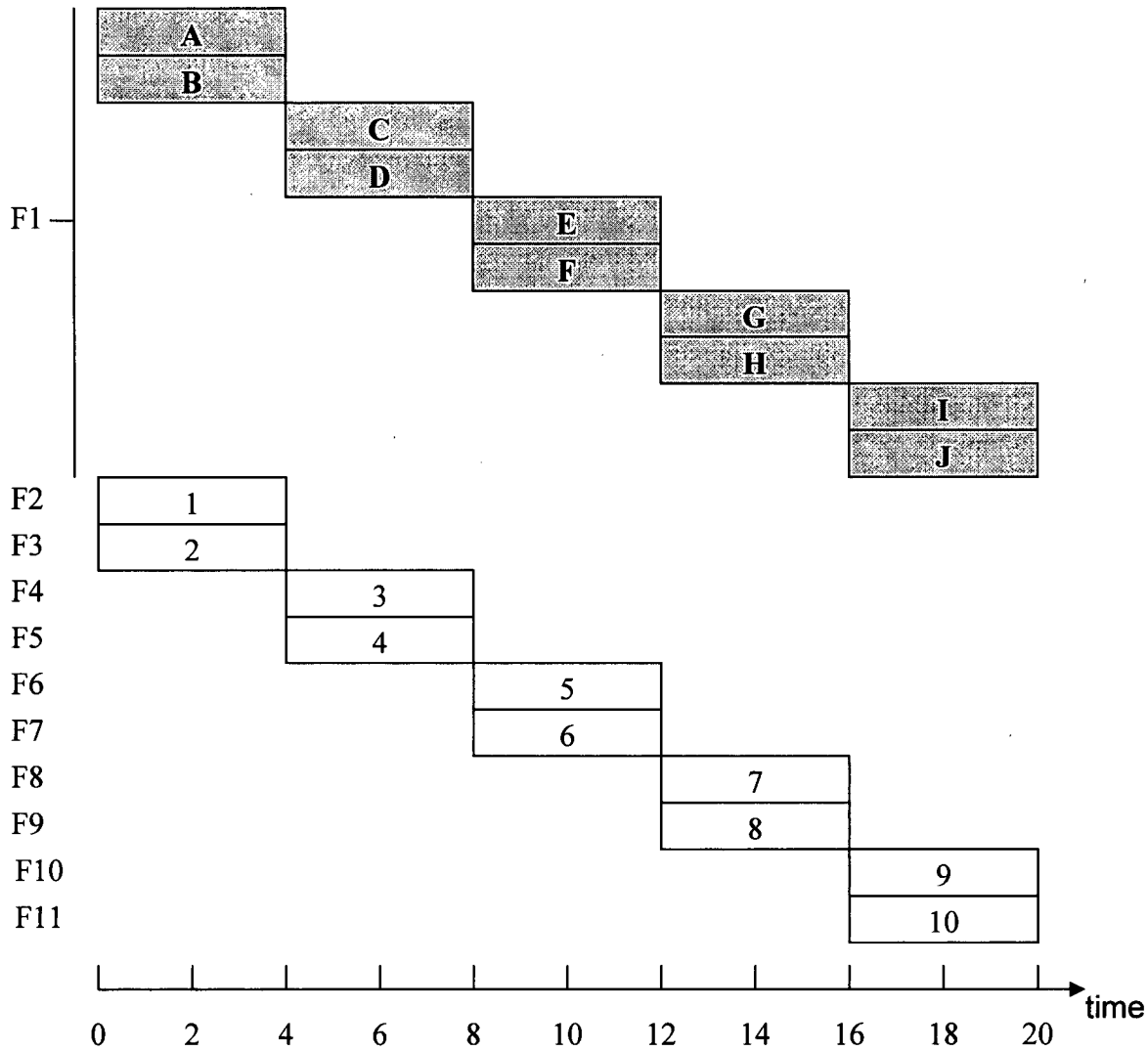


Figure 2.8: MSF^{2}Q scheduling for multiple-server system

2.2.5 Other Multiple-Server Service Disciplines

Research on service disciplines in the last decade has mostly focused on single-server systems. Only a few of them specifically discuss multiple-server systems. We are going to briefly look at another research paper on multi-channel schedulers by Cobb *et al.* [12]. Also, we will look at a research paper by Gopinath *et al.* [14] that studies the impact of packet reordering due to link aggregation on TCP performance.

In his paper [12], Cobb showed that there is a way to transform any work-conserving, single-server service discipline into an equivalent work-conserving, multi-server service

discipline. When applied to PGPS, this transformation produces the same packet delay at a scheduling node as MSFQ, but slightly higher service discrepancy. However, due to packet re-ordering inherent to multiple-server systems, the end-to-end packet delay bound can still be significantly high. In order to eliminate this large increase in delay, Cobb proposed two kinds of sorting techniques with *eligibility times* that are to be implemented in each scheduler that a flow passes through [12]. Assuming every packet of a flow carries an index upon its creation, each technique essentially re-arranges received packets back into the right order before they are processed by a server for transmission. For start-time schedulers, the jitter-reduction sorting technique assigns each transmitted packet a label that represents the difference between the packet's start-time exit bound and its exit time. This results in packet eligibility time increasing with each higher-index packet of the same flow at the *next* scheduler. Thus, packets become eligible and are served at the next scheduler in a sorted order. The fixed-delay sorting technique involves holding a re-ordered packet ineligible at a scheduler until all packets of the same flow with lower index have arrived and become eligible. Unlike the jitter-reduction technique, this later sorting method allows a flow to take advantage of a lightly loaded network by forwarding packets at higher than reserved rate. The trade-off for this efficient bandwidth utilization is that it makes the scheduler non-work conserving.

In another paper [14], Gopinath studied the impact of packet reordering (due to link aggregation) on TCP performance, and proposed schemes to mitigate it. TCP is a common protocol used in the Internet and its implementation is optimized for ordered delivery of packets. Packet reordering can cause unnecessarily large number of fast retransmits that will degrade the throughput of TCP. The authors focused on homogeneous link aggregation techniques that are based on round robin. In particular, a comparison was performed between a *Round Robin (RR)* technique [15] and a proposed variant by the author called *Device-Driven Round Robin (DDRR)* [14]. The proposed scheme is different in two ways. Firstly, all Network Interface Cards (NIC's) comprising the logical link of DDRR share only one interface queue – a queue that bridges the upper-level protocol in the stack with the aggregated NIC's. Secondly, the NIC driver is modified to *pull* packets waiting in this shared queue for transmission rather than letting the aggregation module push them to the NIC. It is explained in the paper that these two properties allow DDRR to avoid very low (almost zero) throughput as experienced by RR when packet fragmentation occurs. The very low throughput is because RR may cause certain NIC's to send only the larger fragments of the packets, while others

only the smaller fragments of the same. In devising solutions to packet reordering, the authors introduced two metrics on measuring the *degree* of packet reordering at a receiver. One such metric, *reorder-queue-length*, is the length of a hypothetical buffer at the receiver that is used to store out-of-order packets that arrive before the next expected packet. The authors showed that reorder-queue-length can be quite large, especially for DDRR, when the NIC's have large transmit buffers and transmit interrupt thresholds. An example in their paper illustrated that when the NIC's large buffers fill up with many packets in an asymmetrical fashion, they cause excessive packet reordering. The other metric, *reorder-burst-length*, is the maximum number of packets which form a continuous ordered sequence of packets once the next expected packet arrives. Higher than 90 percent of the time, the latter metric was shown to be quite small for both round robin techniques, and it was in the order of the number of homogeneous links. Based on these two metrics, some solutions were proposed, such as: reducing the transmit interrupt threshold, modifying transmit routines to spread consecutive packets across NIC's, and implementing reorder correction scheme at the receiver. At the expense of increasing the host system load, reducing the interrupt threshold to one can provide fine-grained scheduling with almost no packet reordering. The multi-server service disciplines discussed in this thesis (MSFQ, MSF^2Q, etc.) actually have fine-grained scheduling because packets are de-queued one at a time from backlogged flow buffers only after a server becomes idle. By de-queueing packets one at a time essentially allows consecutive packets of a flow to be spread across the aggregated links. Also, all de-queued packets have to pass through the same *logical* packetized scheduler before reaching a server (Figure 2.5). In essence, it is consistent with a DDRR with small interrupt threshold. In their paper, Gopinath showed that a reordering correction scheme can be implemented in a lower-layer protocol to prevent the TCP protocol to even detect any reordering. Since this thesis does not specifically focus on solving the packet reordering issue, it would be interesting to see in the future how the proposed correction scheme can alleviate reordering at the receiver. The correction scheme should not affect the fairness property of the multi-server schedulers because it is implemented entirely at the receiver side. Gopinath *et al.* are still working on improving this correction scheme especially in terms of detecting and handling packet loss scenarios using heuristics approach.

2.3 Summary

This chapter introduced the idea of fair queuing by discussing service disciplines for single-server systems. The GPS reference system was discussed and shown to provide the ideal flow isolation that can provide throughput guarantee and protect flows from misbehaving traffic. Packetized implementation of GPS, namely the PGPS, using virtual time implementation was shown to closely approximate GPS by producing worst-case delay bounds and per-flow service discrepancy. Another variant of PGPS, the WF^2Q, further improves PGPS by also adding another desired property of fairness relative GPS. A simple scenario was presented to illustrate the various service disciplines. Other recent popular fair queuing algorithms were briefly discussed. These newer disciplines attempt to find some trade-off between implementation complexity and approximation of GPS. One thing to note is that these schemes are developed based on the assumption of single server..

This chapter also introduced the multiple-server systems and the properties of such systems. One important property is packet re-ordering due to competition from packets of the same flow. The effect of direct implementation of PGPS on such systems was shown by providing new bounds on delay, per-flow service discrepancy and fairness. With total bandwidth now divided into several equal-rate servers, it becomes possible for more than one flows to be serviced simultaneously by packetized systems. MSF^2Q service discipline exploited this fact to provide better fairness for flows compared to GPS by introducing flow scheduling eligibility constraints. The downside of such constraints is non-work conserving system. Another multiple-server service discipline based on transformation and sorting techniques was also discussed at the end of this chapter. Sorting techniques that return packets to the right order at the next server can theoretically avoid large end-to-end delay bound. However, this would require additional sorting computation at each scheduler traversed by a flow. The next chapter will look at adapting MSF^2Q to handle multiple-server systems where the servers are not of equal rates. The focus will be to closely approximate GPS in terms of instantaneous bandwidth usage.

CHAPTER 3 MULTIPLE ASYMMETRICAL SERVER FAIR QUEUEING WITH BOUNDED FAIRNESS

3.1 Multiple Asymmetrical Server Systems

The Multiple Asymmetrical Server (MAS) system is a system similar to MS system in Chapter 3, except that the servers are no longer operating at the same fixed rate of r. In MAS system, at least one of the N servers is operating at a different fixed rate. The fixed server rates range from a minimum rate, r_{min}, to a maximum rate, r_{max}. The total system bandwidth is r_{total} (Figure 3.1). For comparison purposes, the reference model is denoted as (GPS, 1, r_{total}).

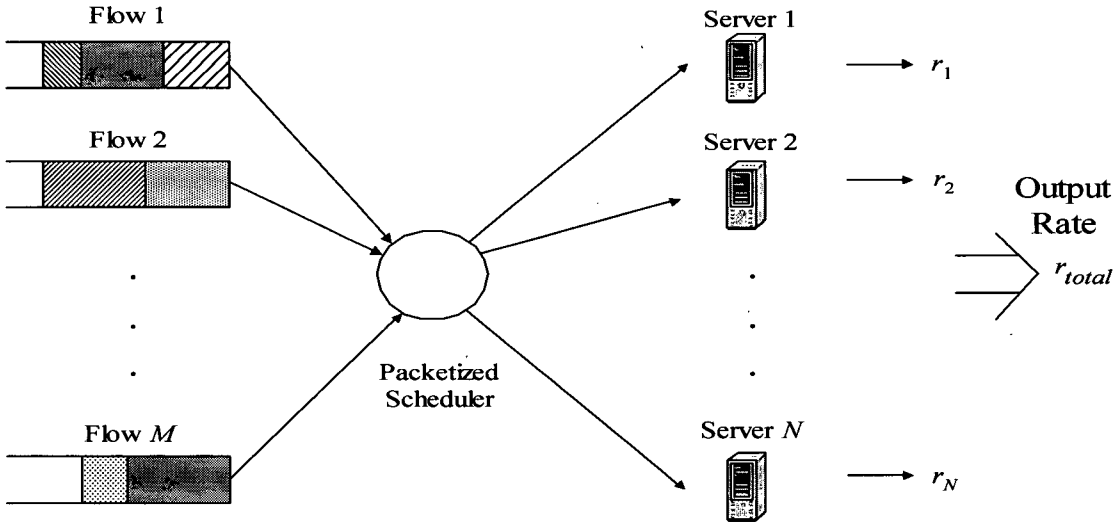


Figure 3.1: Packetized Multiple Asymmetrical Server system

In a link aggregation scenario, the MAS system may occur due to the flexibility of incremental scaling of bandwidth. When new and faster servers are installed to increase the total system bandwidth, they may operate in conjunction with older servers running at slower speed. Although MSF^{2}Q works well in MS systems, it does not specifically take into account the differing server rates of MAS systems. As a result, the traffic pattern generated may no longer be as smooth even though MSF^{2}Q still provides bounded fairness for MAS systems. The new scheme proposed in this thesis, named MASF^{2}Q service discipline, will preserve all

the desired bounds of MSF^{2}Q while it should provide more efficient bandwidth allocation and smoother traffic for MAS systems.

3.2 MSF^{2}Q Service Discipline in MAS Systems

Since MSF^{2}Q was originally developed based on MS systems [1], the next logical step in this thesis is to investigate whether the scheme works well in MAS systems. Following similar steps used by Blanquer *et al.* [1], this chapter determines MSF^{2}Q bounds for worst-case packet delay, maximum per-flow service discrepancy, and fairness in MAS systems. Let (MSF^2Q, N, r_{min}, r_{max}, r_{total}) denotes an N-server MAS system that utilizes MSF^{2}Q service discipline where the servers' rates range from r_{min} to r_{max} for a system bandwidth of r_{total}.

The eligibility constraints defined in equations (2.11) and (2.12) originally proposed by Blanquer *et al.* for MS systems need to be modified to accommodate the different rates of servers in MAS systems. The *modified* scheduling eligibility constraints of MSF^{2}Q for MAS systems are

$$W_{i,\,MSF^2Q}(0,\tau) < W_i(0,\tau) \tag{3.1}$$

or

$$W_{i,\,MSF^2Q}(0,\tau) = W_i(0,\tau) \text{ and } \hat{r}_i < r_i(\tau) \tag{3.2}$$

where $\hat{r}_i$ is the bandwidth *currently* allocated to a backlogged flow i in the packetized system. Mathematically, $\hat{r}_i$ can be written as follows:

$$\hat{r}_i = \sum_{j=1}^{N} s_{ij}(\tau) \cdot r_j \tag{3.3}$$

where: r_j = the rate of server j

$s_{ij}(\tau)$ = 1, if at time τ server j is serving a packet of flow i

= 0, otherwise.

Outstanding packets as calculated in equation (2.12) no longer makes sense here since it was calculated using the common server rate, r. In this case, outstanding packets of a flow directly informs us of the bandwidth occupied by the flow. In MAS system, there is no common rate for the servers. Simply redefining r as the average of server rates, $(r_1 + r_2 + \ldots + r_N) / N$, does produce the equivalent of outstanding packets for MAS systems. Instead, we need to know

how the packets of a flow i are distributed and served among the servers – thus, the use of $\hat{r}_i$.

Theorem 1: As in MS systems, the busy periods of MSF^{2}Q and GPS also do not coincide in MAS systems. This is due to MSF^{2}Q being non work-conserving and the fact that it may operate at less than the system bandwidth r_{total}. While MSF^{2}Q is always busy when GPS is busy, the converse is not true. For any time τ and for maximum packet length L_{max}, this causes a backlog for the packetized system that satisfies

$$W(0,\tau) - W_{MSF^2Q}(0,\tau) \leq (N-1)\cdot L_{\max} \qquad (3.4)$$

Proof: The slope of W is either 0 or r_{total}. The slope of W_{MSF^2Q} ranges from 0 to r_{total}. Thus, the difference $W(0, \tau) - W_{MSF^2Q}(0, \tau)$ is non-decreasing until a busy period ends under GPS. Let τ be such time where GPS server turns idle. At time τ, the maximum difference is achieved with some flows possibly still backlogged under MSF^{2}Q but completely served under GPS. These remaining backlogged flows in MSF^{2}Q system satisfy only the first constraint defined in equation (3.1). Thus, they are eligible to schedule as soon as one or more of the MSF^{2}Q servers become idle on or after time τ.

Case 1: *At most* $(N-1)$ *MSF^{2}Q servers are busy at time* τ:

As mentioned in the previous paragraph, $W(0, t) - W_{MSF^2Q}(0, t)$ is a non-decreasing function up until a busy period ends under GPS at time t. To derive an upper-bound for the function, we are interested in the time period on and after t. During this period, all flows must have finished service under GPS. Some flows may have finished service under MSF^{2}Q, while some may very well still be backlogged (since MSF^{2}Q is non-work conserving). Let us assume that at time t, some of the backlogged flows satisfy the second constraint in equation (3.2). This constraint states that $W_i(0, t) = W_{i,MSF^2Q}(0, t)$ for a backlogged flow i – meaning, that the flow has just caught up to GPS and, thus, has no more packets queued or backlogged beyond t. Therefore, after time t, the remaining backlogged flows in MSF^{2}Q can only satisfy the first constraint defined in equation (3.1). Since there are no restrictions on how many packets to schedule under the first constraint, when there are at least one server idle at time $\tau \geq t$, there must be no more packets queued (in all flows) for scheduling at τ. Let k be the number of busy

servers at time τ, $0 \le k \le (N-1)$. In the worst case, all k servers just started serving packets of length L_{max} each. Thus,

$$W(0,\tau) - W_{MSF^2Q}(0,\tau) \le k \cdot L_{\max} \le (N-1) \cdot L_{\max}$$

Case 2: *All MSF^{2}Q servers are busy at time τ:*

Let $[\tau_o, \tau]$ be the largest interval in which all MSF^{2}Q servers are busy. Since the slope of W_{MSF^2Q} is r_{total} within $[\tau_o, \tau]$,

$$W(0,\tau) - W_{MSF^2Q}(0,\tau) \le W(0,\tau_o) - W_{MSF^2Q}(0,\tau_o)$$

If $\tau_o = 0$, then $W(0,\tau) = W_{MSF^2Q}(0,\tau)$ even though MSF^{2}Q is generally non work-conserving. If $\tau_o > 0$, then at most $(N-1)$ MSF^{2}Q servers are busy at time τ_o. Using Case 1 above,

$$W(0,\tau_o) - W_{MSF^2Q}(0,\tau_o) \le (N-1) \cdot L_{\max} \qquad \square$$

The fairness bound for MSF^{2}Q in MAS systems is unchanged from the one Blanquer *et al.* derived in MS systems [1]. This happens because the bounded fairness property of MSF^{2}Q is independent of the rates of the servers. Therefore, the bounded fairness property also holds in MAS systems and has the same mathematical expression of

$$W_{i,\,MSF^2Q}(0,\tau) - W_i(0,\tau) \le N \cdot L_{i,\max} \qquad (3.5)$$

Proof: A backlogged flow i can be ahead in service under MSF^{2}Q compared to GPS. In this case, GPS will continually be serving flow i until such time that it matches or is ahead of MSF^{2}Q. This is true whether or not there are packets of flow i currently in transmission under MSF^{2}Q. However, when there are no packets of flow i in service under MSF^{2}Q, the difference function $W_{i,\,MSF^2Q}(0, t) - W_i(0, t)$ can only be non-increasing. To derive the upper bound for the difference, we are only interested in the case when MSF^{2}Q is serving packets of flow i. Consider the scenario where MSF^{2}Q and GPS servers are both idle just before time 0, and only N packets of flow i arrived at time 0. In the absence of other backlogged flows, GPS will attempt to serve the packets sequentially with full bandwidth r_{total}. MSF^{2}Q will schedule all the packets in parallel on all its servers at time 0, thus achieving full bandwidth too. Let us assume that a flow j, $j \neq i$, becomes backlogged slightly after time 0. GPS will immediately

service flow j and reduces the bandwidth serving flow i. MSF^{2}Q cannot service flow j because all servers are busy. We can further assume that flow j has a larger weight and larger packet sizes than flow i. In the limit where the weight and the packet size of flow j is very large ($\phi_j >> \phi_i$ and $L_j >> L_i$), flow i may have received inifinitessimally small amount of service from GPS when MSF^{2}Q finishes serving all N packets at time $t > 0$. At time t, MSF^{2}Q will be at most $N \cdot L_{i,max}$ ahead in service compared to GPS since $L_i < L_{i,max}$. This means that flow i temporarily satisfy neither of the eligibility constraints of MSF^{2}Q as defined in equations (3.1) and (3.2). Even though flow i may still be backlogged, MSF^{2}Q cannot schedule any of its packets until GPS catches up. Thus, $N \cdot L_{i,max}$ is the fairness bound for a backlogged flow i.

Example 1: To illustrate how the worst-case packet delay may happen, let us consider a scenario where GPS is ahead in service compared to MSF^{2}Q. Let $t > 0$ be the time when a busy period of GPS ends. According to theorem 1, MSF^{2}Q can be lagging in service by as much as $(N-1) \cdot L_{max}$ where L_{max} is the largest packet size in the system. Assume that this is the case at time t and that only $(N-1)$ of the MSF^{2}Q servers are busy. This means that each of the busy servers has just started to serve a packet of length L_{max}, and that there are no more packets waiting for service in any of the flows. Now, let a packet p_m of length L_{max} arrive immediately after. GPS will attempt to service the packet at full bandwidth. At time t, each of the flows in MSF^{2}Q satisfies either one of the two eligibility constraints as defined in equations (3.1) and (3.2); thus, MSF^{2}Q will also service packet p_m on the last remaining idle server. Assume that more packets $p_{m+1}, \ldots, p_{k-1}, p_k$ arrive immediately after packet p_m has been scheduled for service. Assume these packets arrive on other flows with much heigher weights that finish service under GPS before packet p_m. Let the lengths of these new packets be small enough that they all finish under GPS before MSF^{2}Q finishes serving any of the N packets of length L_{max}. Them the flows containing the newly arrived packets will remain eligible for scheduling under MSF^{2}Q, and the packets will be scheduled according to increasing GPS departure times. Let the subscripts $m, m+1, \ldots, k-1,$ *and* k indicate the order in which MSF^{2}Q schedules these packets. Then, packet p_k is the last to depart GPS and its departure time d_k satisfies:

$$d_k \geq t + \frac{1}{r_{total}} \cdot \sum_{i=m+1}^{k} L_i$$

Before MSF^{2}Q schedule packet p_k, it must service at most N packets of length L_{max} and packets $p_{m+1}, \ldots, p_{k-1}$. Thus, the departure time of packet p_k under MSF^{2}Q, $d_{k,\,MSF^2Q}$, satisfies:

$$d_{k,\,MSF^2Q} \le t + \frac{1}{r_{total}} \cdot (N \cdot L_{\max} + \sum_{i=m+1}^{k-1} L_i) + \frac{L_k}{r_{\min}}$$

The packet delay is then the difference of the packet departure times:

$$d_{k,\,MSF^2Q} - d_k \le (\frac{1}{r_{\min}} - \frac{1}{r_{total}}) \cdot L_k + \frac{N \cdot L_{\max}}{r_{total}}$$

Theorem 2: The worst-case packet delay for MSF^{2}Q in MAS systems is bounded relative to GPS, and satisfies

$$d_{k,\,MSF^2Q} - d_k \le (\frac{1}{r_{\min}} - \frac{1}{r_{total}}) \cdot L_k + \frac{N \cdot L_{\max}}{r_{total}} \qquad (3.6)$$

where

d_k = departure time of packet p_k under GPS

$d_{k,\,MSF^2Q}$ = departure time of packet p_k under MSF^{2}Q

L_k = length of packet p_k.

Proof: Let p_k be the kth packet that is scheduled under MSF^{2}Q. Let a_k be the arrival time of packet p_k. Let b_k and $b_{k,\,MSF^2Q}$ be the scheduling times of packet p_k under GPS and MSF^{2}Q, respectively. In order to derive the bound for worst-case packet delay, we have to consider the following cases.

Case 1: *There is an interval* $[t,\ b_{k,\,MSF^2Q}]$ *in which at least one MSF^{2}Q server is continuously idle.*

There are two ways this case can happen and they are shown in the following sub-cases.

Case 1.1: *Packet p_k arrives at flow i that satisfies the first scheduling eligibility constraint of MSF^{2}Q.*

In this case, p_k is the last packet in the flow and it will be scheduled immediately by MSF^{2}Q upon its arrival. Thus, the following equations hold:

$$b_{k,\ MSF^2Q} = a_k$$

$$\text{For GPS:}\quad d_k \geq a_k + \frac{L_k}{r_{total}} \qquad \text{since } b_k \geq a_k$$

$$\text{For MSF}^2\text{Q:}\quad d_{k,\ MSF^2Q} \leq b_{k,\ MSF^2Q} + \frac{L_k}{r_{\min}}$$

$$\text{Thus:}\quad d_{k,\ MSF^2Q} - d_k \leq \left(\frac{1}{r_{\min}} - \frac{1}{r_{total}}\right) \cdot L_k$$

Case 1.2: *Packet p_k arrives at flow i that was initially ineligible to schedule, but the flow becomes eligible later by satisfying either one of the two scheduling eligibility constraints of MSF^2Q.*

At time $b_{k,\ MSF^2Q}$, packet p_k will be scheduled on one of the idle servers possibly along with other packets eligible at that time. Thus, the following relationships are valid.

$$b_{k,\ MSF^2Q} > a_k$$

$$\text{For GPS:}\quad d_k \geq b_{k,\ MSF^2Q} + \frac{L_k}{r_{total}} \qquad \text{since } b_k \geq b_{k,\ MSF^2Q}$$

$$\text{For MSF}^2\text{Q:}\quad d_{k,\ MSF^2Q} \leq b_{k,\ MSF^2Q} + \frac{L_k}{r_{\min}}$$

$$\text{Thus:}\quad d_{k,\ MSF^2Q} - d_k \leq \left(\frac{1}{r_{\min}} - \frac{1}{r_{total}}\right) \cdot L_k$$

Case 2: *There is an interval $[t,\ b_{k,\ MSF^2Q}]$ in which all MSF^2Q servers are continuously busy.*

Let g be the smallest integer such that all MSF^{2}Q servers are busy between $b_{g,\ MSF^2Q}$ and $b_{k,\ MSF^2Q}$. Let m be the largest integer greater than or equal to g that satisfies both $g \leq m \leq k-1$ and $d_m > d_k$. Thus, for $m < i < k$,

$$d_m > d_k \geq d_i$$

The integer m accounts for possible packet re-ordering under MSF^{2}Q due to late arrival of packets. These late arrivals may prevent scheduling process under MSF^{2}Q that is strictly based

on the increasing GPS packet departure times. Because of these late arrivals, two further sub-cases are possible.

Case 2.1: *No such integer m exists.*

Packets p_g, p_{g+1}, ..., p_{k-1} are all scheduled before packet p_k under MSF^{2}Q, and they all departed before p_k under GPS. In order for $b_{g,\ MSF^2Q}$ to be the start of a continuously busy period under MSF^{2}Q, there needs to be at least one idle server before and until $b_{g,\ MSF^2Q}$. Then, as in Cases 1.1 and 1.2 above,

$$b_g \geq b_{g,\ MSF^2Q}$$

Under GPS, it follows that

$$d_k \geq b_{g,\ MSF^2Q} + \frac{1}{r_{total}} \cdot \sum_{i=g}^{k} L_i$$

Under MSF^{2}Q, at most $(N-1)$ other packets that started service before $b_{g,\ MSF^2Q}$ can still be in service at $b_{g,\ MSF^2Q}$. Thus, before $b_{k,\ MSF^2Q}$, MSF^{2}Q has to serve at most $(N-1)$ packets of length L_{max} in addition to p_g, p_{g+1}, ..., p_{k-1}. Assuming that packet p_k is scheduled onto the slowest server, then the following upper-bound is obtained

$$d_{k,\ MSF^2Q} \leq b_{g,\ MSF^2Q} + \frac{1}{r_{total}} \cdot \{(N-1) \cdot L_{\max} + \sum_{i=g}^{k-1} L_i\} + \frac{L_k}{r_{\min}}$$

Thus, the worst-case packet delay is

$$d_{k,\ MSF^2Q} - d_k \leq (\frac{1}{r_{\min}} - \frac{1}{r_{total}}) \cdot L_k + \frac{(N-1) \cdot L_{\max}}{r_{total}}$$

Case 2.2: *Such an integer m exists.*

Since MSF^{2}Q schedules packets of eligible backlogged flows in increasing order of GPS packet departure times, such an integer m can only exist in the following sub-cases:

Case 2.2.1: *Packets p_{m+1}, ..., p_k arrive after $b_{m,\ MSF^2Q}$.*

Since

$$\min(a_{m+1}, \ldots, a_k) > b_{m,\ MSF^2Q},$$

it is impossible that MSF^{2}Q could have considered packets p_{m+1}, ..., p_k for scheduling at $b_{m,\ MSF^2Q}$ even if they have earlier GPS packet departure times than packet p_m. This implies that

$$\min(b_{m+1}, \ldots, b_k) > b_{m,\ MSF^2Q}$$

Case 2.2.2: *Packets p_{m+1}, ..., p_k arrived before $b_{m,\ MSF^2Q}$.*
There are further 2 sub-cases to be considered.

Case 2.2.2.1: *Packets p_{m+1}, ..., p_k are queued on flows that satisfy $W_{i,\ MSF^2Q} > W_i$ at time $b_{m,\ MSF^2Q}$.*

In this case, packets p_{m+1}, ..., p_k are not eligible for scheduling by MSF^{2}Q at $b_{m,\ MSF^2Q}$. In order for both GPS and MSF^{2}Q to be able to schedule any of these packets, GPS has to catch up to MSF^{2}Q in terms of its service of the flows. This translates into

$$\min(b_{m+1}, \ldots, b_k) > b_{m,\ MSF^2Q}$$

Case 2.2.2.2: *Packets p_{m+1}, ..., p_k are queued on flows that satisfy $W_{i,\ MSF^2Q} = W_i$ at time $b_{m,\ MSF^2Q}$, and these flows in MSF^{2}Q have achieved or exceeded their GPS instantaneous rates.*

In this case, packets p_{m+1}, ..., p_k are not eligible for scheduling by MSF^{2}Q at $b_{m,\ MSF^2Q}$. This means that

$$\min(b_{m+1}, \ldots, b_k) > b_{m,\ MSF^2Q}$$

Under GPS, it follows that for Case 2.2:

$$d_k \geq b_{m,\ MSF^2Q} + \frac{1}{r_{total}} \cdot \sum_{i=m+1}^{k} L_i$$

As in Case 2.1 above, under MSF^{2}Q, at most $(N-1)$ other packets of length L_{max} that started service before $b_{m,\ MSF^2Q}$ may still be in service at $b_{m,\ MSF^2Q}$. Thus, before $b_{k,\ MSF^2Q}$, MSF^{2}Q has to serve at most $(N-1)$ packets of length L_{max} in addition to p_m, p_{m+1}, ..., p_{k-1}. The latest time

before MSF^{2}Q has to schedule packet p_k can be found by making sure that all packets prior to p_k finish service at exactly the same time. Thus, for Case 2.2, the bound for $b_{k,\ MSF^2Q}$ satisfies

$$b_{k,\ MSF^2Q} \le b_{m,\ MSF^2Q} + \frac{1}{r_{total}} \cdot \{(N-1) \cdot L_{\max} + L_m + \sum_{i=m+1}^{k-1} L_i\}$$

Assuming that packet p_k is scheduled onto the slowest server, then the bound on MSF^{2}Q packet departure times for Case 2.2 can be calculated as

$$\begin{aligned} d_{k,\ MSF^2Q} &\le b_{k,\ MSF^2Q} + \frac{L_k}{r_{\min}} \\ &\le b_{m,\ MSF^2Q} + \frac{1}{r_{total}} \cdot \{(N-1) \cdot L_{\max} + L_m + \sum_{i=m+1}^{k-1} L_i\} + \frac{L_k}{r_{\min}} \end{aligned}$$

Thus, the worst-case packet delay for Case 2.2 is

$$\begin{aligned} d_{k,\ MSF^2Q} - d_k &\le (\frac{1}{r_{\min}} - \frac{1}{r_{total}}) \cdot L_k + \frac{N \cdot L_{\max}}{r_{total}} + \frac{(L_m - L_{\max})}{r_{total}} \\ &\le (\frac{1}{r_{\min}} - \frac{1}{r_{total}}) \cdot L_k + \frac{N \cdot L_{\max}}{r_{total}} \end{aligned}$$

since $0 \le L_m \le L_{max}$. □

Example 2: To illustrate how the maximum per-flow service discrepancy may happen, let us consider a scenario similar to that in Example 1. For simplicity, let the total rate, r_{total}, be 1 bps for both GPS and MSF^{2}Q. Assume that there are 3 servers under MSF^{2}Q with the rates: r_1 = 0.5 bps, and $r_2 = r_3$ = 0.25 bps. Let the maximum packet length, L_{max}, be 1 bit. Assume that at time t = 0, both GPS and MSF^{2}Q are idle. Assume that 3 packets p_1, p_2, and p_3 of length L_{max} each arrive at a flow i at time t = 0. These packets will immediately be scheduled by both GPS and MSF^{2}Q. Assume another packet p_4 of length L_{max} arrives shortly after at a flow j that has a flow weight much larger than that of flow i ($\phi_j >> \phi_i$). GPS will immediately schedule packet p_4 with almost full bandwidth and finishes service slightly after t = 1 second. The earliest time MSF^{2}Q can consider scheduling packet p_4 is at $t = L_{max} / r_1$ = 2 seconds when server 1 of MSF^{2}Q becomes idle. But before then, let us assume that 3 more new packets p_5, p_6, and p_7 of length L_{max} each arrive at a flow k, that has a flow weight $\phi_k >> \phi_l$, shortly before GPS finishes

servicing packet p_4 at time $t = 1$ second. After packet p_4, GPS will serve all these new packets sequentially at almost full bandwidth and will finish at slightly after $t = 4$ seconds. At time $t = 2$ seconds, both flow j and k are lagging in service under MSF^{2}Q and are eligibile for scheduling. At this instant, MSF^{2}Q selects packet p_4 because it finished first in GPS relative to the 3 newer packets. So, by time $t = 4$ seconds, all packets p_5, p_6, and p_7 of flow k would have finished service under GPS, but MSF^{2}Q would have just started servicing these packets on all its servers. Thus, in this case, $W_k(0, 4) - W_{k, MSF^2Q}(0, 4) \leq 3 \cdot L_{max}$. It can be shown later that, for a more general N server case, the bound is $N \cdot L_{max}$.

Theorem 3: The maximum per-flow service discrepancy for MSF^{2}Q in MAS systems is bounded relative to GPS, and, for any time τ and flow i, satisfies

$$W_i(0,\tau) - W_{i,\ MSF^2Q}(0,\tau) \leq N \cdot L_{\max} \qquad (3.7)$$

where

$W_i(0, \tau)$ = amount of flow i bits served by GPS within time period $[0, \tau]$

$W_{i, MSF^2Q}(0, \tau)$ = amount of flow i bits served by MSF^{2}Q within time period $[0, \tau]$.

Proof: Maximum per-flow service discrepancy occurs at a point when the slope of W_i decreases enough, or the slope of W_{i, MSF^2Q} increases enough. The slope of W_i decreases either when flow i becomes idle in GPS, or, when a flow $j, j \neq i$, becomes backlogged in GPS while flow i is still backlogged in GPS. The slope of W_{i, MSF^2Q} *can* increase when a packet of flow i is scheduled in MSF^{2}Q. The case of flow i becoming idle in GPS is simply a special case of a packet of flow i leaving GPS.

Case 1: *Packet p_k is scheduled in MSF^{2}Q before it is fully serviced in GPS* ($b_{k, MSF^2Q} \leq d_k$). Let p_k be a packet belonging to flow i. When GPS schedules packet p_k, it must have already finished serving all other packets belonging to flow i that were queued before p_k. At b_{k, MSF^2Q}, flow i must satisfy either one of the two scheduling eligibility constraints of MSF^{2}Q in order for p_k to be scheduled by the packetized system. Because there could be multiple idle servers, MSF^{2}Q may be transmitting packet p_k along with a maximum of $(N - 1)$ other packets of flow i that were scheduled before b_{k, MSF^2Q}. Thus,

$$W_i(0,d_k) - W_{i,\,MSF^2Q}(0,b_{k,\,MSF^2Q}) \le (N-1)\cdot L_{i,\max} + L_k \qquad (3.8)$$

Case 1.1: *Packet p_k of flow i departs GPS.*

This case includes the above scenario when the last packet of flow i departs GPS and the flow becomes idle in GPS. Since $W_{i,\,MSF^2Q}(0, b_{k,\,MSF^2Q}) \le W_{i,\,MSF^2Q}(0, d_k)$, it follows from equation (3.8) that

$$\begin{aligned} W_i(0,d_k) - W_{i,\,MSF^2Q}(0,d_k) &\le (N-1)\cdot L_{i,\max} + L_k \\ &\le N\cdot L_{i,\max} \end{aligned}$$

since $0 \le L_k \le L_{i,\,max}$.

Case 1.2: *Packet p_k of flow i is scheduled in MSF^2Q.*

Since $W_i(0, b_{k,\,MSF^2Q}) \le W_i(0, d_k)$, it follows from equation (3.8) that

$$\begin{aligned} W_i(0,b_{k,\,MSF^2Q}) - W_{i,\,MSF^2Q}(0,b_{k,\,MSF^2Q}) &\le (N-1)\cdot L_{i,\max} + L_k \\ &\le N\cdot L_{i,\max} \end{aligned}$$

since $0 \le L_k \le L_{i,\,max}$.

Case 2: *Packet p_k is scheduled in MSF^2Q after it is fully serviced in GPS ($b_{k,\,MSF^2Q} > d_k$).*

We will consider per-flow service discrepancy for any time $\tau \in [d_k, b_{k,\,MSF^2Q}]$. This permits simultaneous consideration of both Cases 1.1 and 1.2 above. Let p_k be the kth packet scheduled under MSF^2Q. For any time τ, packet p_k belongs to a flow i that has to satisfy the first scheduling eligibility constraint of MSF^2Q as defined in equation (3.1). Thus, flow i is eligible for scheduling at any time within $[d_k, b_{k,\,MSF^2Q}]$. The fact that packet p_k is scheduled at $b_{k,\,MSF^2Q}$ means that all servers are busy serving other packets (of any flows) within $[d_k, b_{k,\,MSF^2Q}]$. This is because if a server is idle within that interval, packet p_k would have been scheduled earlier. So, let there be an interval $[t, b_{k,\,MSF^2Q}]$, $t < d_k$, in which all MSF^2Q servers are busy. Let g be the smallest integer such that all MSF^2Q servers are busy between $b_{g,\,MSF^2Q}$ and $b_{k,\,MSF^2Q}$. Let m be the largest integer greater than or equal to g that satisfies both $g \le m \le k-1$ and $d_m > d_k$. Thus, for $m < i < k$,

$$d_m > d_k \ge d_i.$$

The integer m accounts for possible packet re-ordering under MSF^{2}Q due to late arrival of packets. Because of these late arrivals, two further sub-cases are possible.

Case 2.1: *No such integer m exists.*

Packets p_g, p_{g+1}, ..., p_{k-1} are all scheduled before packet p_k under MSF^{2}Q, and they all departed before p_k under GPS. In order for $b_{g,\ MSF^2Q}$ to be the start of a continuously busy period under MSF^{2}Q, there needs to be at least one idle server before and until $b_{g,\ MSF^2Q}$. Then, as shown in Cases 1.1 and 1.2 of Theorem 1,

$$b_g \geq b_{g,\ MSF^2Q}.$$

In $[b_{g,\ MSF^2Q}, b_{k,\ MSF^2Q}]$, MSF^{2}Q may be servicing at most $(N-1)$ other packets of length L_{max} besides packets p_g, p_{g+1}, ..., p_{k-1}. In the worst case, GPS starts service of flow i packets p_g p_g, p_{g+1}, ..., p_{k-1} at time $b_g = b_{g,\ MSF^2Q}$ and with maximum bandwidth r_{total}. Thus, in the worst case, GPS service of flow i is

$$W_i(b_{g,\ MSF^2Q}, \tau) = (\tau - b_{g,\ MSF^2Q}) \cdot r_{total}$$

Based on when packet p_k finishes service in GPS relative to the $(N-1)$ other packets of length L_{max}, there are 3 sub-cases.

Case 2.1.1: $(\tau - b_{g,\ MSF^2Q}) \geq (L_{max} / r_{min})$

In this case, all of the $(N-1)$ packets of length L_{max} have finished service in MSF^{2}Q at time τ. Thus, MSF^{2}Q is behind in its service of flow i by the equivalent of $(N-1)$ packets of length L_{max} and satisfies

$$W_i(0,\tau) - W_{i,\ MSF^2Q}(0,\tau) \leq (N-1) \cdot L_{\max}$$

Case 2.1.2: $(L_{max} / r_{max}) \leq (\tau - b_{g,\ MSF^2Q}) < (L_{max} / r_{min})$

In this case, only some of the $(N-1)$ packets of length L_{max} may have finished service in MSF^{2}Q at time τ. Let the packets that have finished service occupy servers numbered 1 to x. These packets cause $(x \cdot L_{max})$ of lost service in MSF^{2}Q to packets p_g, p_{g+1}, ..., p_k scheduled at

or after $b_{g,\ MSF^2Q}$. For a server j, $x+1 \le j \le N-1$, MSF^{2}Q also suffers $(L_{max} - (\tau - b_{g,\ MSF^2Q}) \cdot r_j)$ of lost service to packets scheduled before packet p_g, and $(\tau - b_{g,\ MSF^2Q}) \cdot r_j$ of lost service to packets $p_g, p_{g+1}, \ldots, p_k$. Assuming that these $(N-1)$ packets belong to flow i, adding all the lost service in MSF^{2}Q at time τ results in

$$W_i(0,\tau) - W_{i,\ MSF^2Q}(0,\tau) \le (N-1) \cdot L_{\max}$$

Case 2.1.3: $(L_{max} / r_{max}) \le (\tau - b_{g,\ MSF^2Q}) < (L_{max} / r_{min})$

In this case, none of the $(N-1)$ packets of length L_{max} have finished service in MSF^{2}Q at time τ. Since this is a special case of Case 2.1.2 above (where x equals to zero), the same bound on per-flow service discrepancy holds.

Case 2.2: *Such an integer m exists.*

In this case, we focus on the service performed on packets $p_m, \ldots, p_k$. $b_{m,\ MSF^2Q} < d_k$ since it has been shown in Case 2.2 of Theorem 2 that

$$\min(b_{m+1}, \ldots, b_k) > b_{m,\ MSF^2Q}$$

In order for p_m to be the re-ordered packet due to late arrivals, packet p_m must belong to flows other than the flow i being considered. In $[b_{m,\ MSF^2Q},\ b_{k,\ MSF^2Q}]$, MSF^{2}Q may be servicing at most $(N-1)$ packets of length L_{max} and packet p_m besides packets $p_{m+1}, p_{m+2}, \ldots, p_{k-1}$ of flow i. In the worst case, GPS starts service of flow i packets $p_{m+1}, p_{m+2}, \ldots, p_k$ at the earliest time $b_{m+1} \approx b_{m,\ MSF^2Q}$ and with maximum bandwidth r_{total}. Thus, in the worst case, GPS service of flow i is

$$W_i(b_{m,\ MSF^2Q}, \tau) = (\tau - b_{m,\ MSF^2Q}) \cdot r_{total}$$

Based on when packet p_k finishes service in GPS relative to the $(N-1)$ packets of length L_{max}, there are 3 sub-cases similar to those of Cases 2.1.1, 2.1.2, and 2.1.3.

Case 2.2.1: $(\tau - b_{m,\ MSF^2Q}) \ge (L_{max} / r_{min})$

In this case, all of the $(N-1)$ packets of length L_{max} and packet p_m have finished service in

MSF^{2}Q at time τ. Thus, MSF^{2}Q is behind in its service of flow i by the equivalent of $(N-1)$ packets of length L_{max} plus a packet of length L_m, and satisfies

$$\begin{aligned} W_i(0,\tau) - W_{i,\,MSF^2Q}(0,\tau) &\le (N-1)\cdot L_{\max} + L_m \\ &\le N \cdot L_{\max} \end{aligned}$$

since $0 \le L_m \le L_{max}$.

Case 2.2.2: $(L_{max} / r_{max}) \le (\tau - b_{m,\,MSF^2Q}) < (L_{max} / r_{min})$

Only some of the $(N-1)$ packets of length L_{max} may have finished service in MSF^{2}Q at time τ. Let the packets that have finished service occupy servers numbered 1 to x. These packets cause $(x \cdot L_{max})$ of lost service in MSF^{2}Q to packets $p_{m+1}, p_{m+2}, \ldots, p_k$ scheduled after $b_{m,\,MSF^2Q}$. For a server j, $x+1 \le j \le N-1$, MSF^{2}Q also suffers $(L_{max} - (\tau - b_{g,\,MSF^2Q}) \cdot r_j)$ of lost service to packets scheduled before packet p_m, and $(\tau - b_{g,\,MSF^2Q}) \cdot r_j$ of lost service to packets $p_{m+1}, p_{m+2}, \ldots, p_k$. As packet p_m belongs to a flow other than flow i, it causes additional lost service in MSF^{2}Q equivalent to its length L_m. Assuming that these $(N-1)$ packets belong to flow i, adding all the lost service in MSF^{2}Q at time τ results in

$$\begin{aligned} W_i(0,\tau) - W_{i,\,MSF^2Q}(0,\tau) &\le (N-1)\cdot L_{\max} + L_m \\ &\le N \cdot L_{\max} \end{aligned}$$

since $0 \le L_m \le L_{max}$.

Case 2.2.3: $(\tau - b_{m,\,MSF^2Q}) < (L_{max} / r_{max})$

None of the $(N-1)$ packets of length L_{max} have finished service in MSF^{2}Q at time τ. As packet p_m belongs to a flow other than flow i, it causes additional lost service in MSF^{2}Q equivalent to its length L_m. Since this is a special case of Case 2.2.2 above (where x equals to zero), the same bound on per-flow service discrepancy holds.

Case 3: *An idle flow j, $j \ne i$, becomes backlogged in GPS at time t while flow i is still backlogged in GPS.*

Let p_k be the kth packet of flow i that completes under GPS. Since flow j can only become backlogged in GPS between two consecutive flow i packet departure times in GPS, t is in $[d_k, d_{k+1}]$ for *any* integer k. Since this interval $[d_k, d_{k+1}]$ has been considered by all the cases above,

the proof of case 3 follows from the proofs of the previous cases, and the same bound on per-flow service discrepancy holds. □

As shown above, MSF^{2}Q provides bounded fairness and bounds for worst-case packet delay and per-flow service discrepancy in MAS systems. The bounds derived in this chapter are general bounds that include the smaller bounds of MS systems as special case. The bounds in this chapter can be easily translated to MS systems by the following substitutions:

$$r_{total} = N \cdot r$$

$$r_{min} = r_{max} = r_i = r \qquad \forall\, i \in [1, N].$$

Despite the bounds, we will show using examples in Section 3.4 that MSF^{2}Q produces bursty traffic pattern in MAS systems compared to GPS. The bursty pattern is caused by the inability of MSF^{2}Q to determine which of the available idle servers is the most suitable to serve a particular packet scheduled for transmission. In Section 3.3, additional rules will be included in MSF^{2}Q to handle this issue and to generate smoother traffic pattern in MAS systems.

3.3 MASF^{2}Q Service Disciplines

When considering MSF^{2}Q in MS systems (Section 2.2.3), the main issue is to decide which of the flows are eligible to schedule at any given time. When a decision has been made, packets from eligible flows are *arbitrarily* assigned to one of the idle servers because the rate for each server is the same. MSF^{2}Q in MS systems achieved bounded fairness property and smoother output traffic pattern. Applying MSF^{2}Q directly in MAS systems still produced desirable bounded fairness property along with bounds for packet delay and per-flow service discrepancy (Section 3.2), but it may generate a burstier output pattern as will be shown in a theoretical example in Section 3.4.

When a packet from eligible flows is scheduled for transmission in MAS systems, it may be desirable or preferable at the time to assign the packet to a *specific* server among the available idle servers. This assignment is called *packet-server pairing* and it makes use of the fact that servers have different fixed rates. Without good packet-server pairings, MSF^{2}Q may

inefficiently distribute available bandwidth among eligible flows in MAS systems. This may lead certain eligible flows to either lead much ahead or lag much behind in their service in MSF^2Q relative to GPS. This exaggerated lead or lag in service causes the packetized system to deviate from approximating GPS at a finer time scale. We propose a new scheduling scheme called *$MASF^2Q$* that combines MSF^2Q scheduling eligibility constraints and packet-server pairing algorithms. The reason we have chosen MSF^2Q over other schemes for MAS systems is that it already uses per-flow service and per-flow rate variables to calculate flow eligibility. These variables are the basis for deciding packet-server pairings.

A simple example of inefficient bandwidth allocation is to assume two backlogged flows at time $t = 0$ when the two servers are idle. Let flow F1 have a weight of 0.25, and a packet of length 1 bit queued at time $t = 0$. Let flow F2 have a weight of 0.75, and a packet of length 3 bits queued at time $t = 0$. Let servers S1 and S2 have rates of 0.75 bps and 0.25 bps, respectively. In GPS with a total bandwidth of 1 bps, both packets of flows F1 and F2 finish simultaneously at time $t = 4$. Thus, MSF^2Q can choose either one of the packets to schedule first. Assume it chooses packet of flow F1 first. MSF^2Q can now arbitrarily assign this packet to either server S1 or S2. Assume that there is a general preference to schedule onto the fastest server available. Then, packet of flow F1 is scheduled onto server S1, and packet of flow F2 onto server S2. In this MAS system, flow F1 finishes service at $t = 1.333$, and flow F2 at $t = 12$. Due to inefficient bandwidth allocation, flow F1 leads so much ahead in service relative to GPS at the expense of flow F2 that lags so much behind in service. $MASF^2Q$ scheme would have assigned packets of flows F1 and F2 to servers S2 and S1, respectively. This will allow both flows to finish at $t = 4$ in MAS systems and better approximate GPS.

In this thesis, we propose $MASF^2Q$ with the following steps of scheduling packets onto idle servers in MAS systems. At any time t when there is at least one idle server:

1. Determine scheduling eligibility of flows using the modified MSF^2Q eligibility constraints as defined in equations (3.1) and (3.2).
2. If there are v number of servers idle at time t, $v \leq N$, select h number of packets, $h \leq v$, among eligible flows in the increasing order of GPS packet departure times.
3. Determine packet-server pairing *assignments* of selected packets based on the rules to be established below.
4. Schedule packets onto idle servers according to the assignments in Step 3.

In order to achieve good packet-server pairing, the following rules have been proposed.

Rule 1: *If a packet scheduled for transmission at time t comes from a flow i that satisfies the second eligibility constraint of MSF^{2}Q as defined in equation (3.2), with $\hat{r}_i$ as defined in equation* (3.3), *then:*

Rule 1.1: *Attempt to assign the packet to the fastest unassigned idle server at time t such that $\hat{r}_i \leq r_i(t)$ remains valid.*

This rule allows flow *i* in MASF^{2}Q to approximate its GPS service by competing for the fastest available server and, *at the same time*, trying not to exceed its current GPS rate. The effort of competing for the fastest available server prevents flow *i* in MSF^{2}Q from lagging too much in service relative to GPS. The effort of not exceeding its GPS rate may allow flow *i* to schedule more packets at time *t*.

Rule 1.2: *If Rule 1.1 cannot be followed, attempt to assign the packet to the slowest unassigned idle server at time t such that $\hat{r}_i \geq r_i(t)$.*

This rule allows flow *i* in MSF^{2}Q to approximate its GPS service by ensuring that any excess bandwidth relative to GPS will be as small as possible. This prevents flow *i* in MSF^{2}Q from leading too much in service compared to GPS at the expense of other flows.

Rule 2: *If a packet scheduled for transmission at time t comes from a flow i that satisfies the first eligibility constraint of MSF^{2}Q as defined in equation (3.1), with $\hat{r}_i$ as defined in equation* (3.3), *then:*

Rule 2.1: *If $\hat{r}_i < r_i(t) \neq 0$, attempt to assign the packet to the slowest unassigned idle server at time t such that $\hat{r}_i \geq r_i(t)$.*

This rule allows flow *i* in MASF^{2}Q to get slightly more bandwidth than its current GPS rate in order to attempt catching up to its GPS service. By getting only slightly more bandwidth, flow *i* in the packetized system will not lead too much in service relative to GPS at the expense of other flows.

Rule 2.2: *If* $\hat{r}_i < r_i(t) \neq 0$ and there are no idle servers available that satisfy Rule 2.1, *assign the packet to the fastest unassigned idle server at time t.*

This rule allows flow *i* in MSF^{2}Q to get as much bandwidth as possible to close the rate gap relative to GPS. This rule may allow flow *i* in the packetized system to schedule more packets.

Rule 2.3: *If* $\hat{r}_i \geq r_i(t) \neq 0$, *assign the packet to the slowest unassigned idle server at time t.*

This rule attempts to limit excess bandwidth of flow *i* in MSF^{2}Q relative to GPS. Since flow *i* already possesses a rate larger than its current GPS rate, it has lower priority on obtaining more bandwidth in the packetized system. This allows other flows that are behind their GPS service to use remaining available bandwidth to catch up to the GPS system.

Rule 2.4: *If* $r_i(t) = 0$, *assign the packet to the fastest unassigned idle server at time t.*

In this case, GPS has completely served flow *i* packets. Flow *i* in MSF^{2}Q may lag its GPS service by a large margin at time *t*. Thus, this rule gives flow *i* higher priority and allows flow *i* in MSF^{2}Q to catch up to its service in GPS.

Note that the packet-server pairing decision is performed as a separate process after a packet has been selected among eligible flows based on increasing order of GPS departure times. The bounds for worst-case packet delay, maximum per-flow service discrepancy and bounded fairness are related only to the initial packet selection process. Thus, the same bounds hold for both MASF^{2}Q and MSF^{2}Q in MAS. However, as will be shown in Section 3.4, MASF^{2}Q provides smoother output pattern by evenly distributing bandwidth at a finer time scale in the packetized system.

3.4 A Comparison of MSF^{2}Q and MASF^{2}Q

This section discusses fairness comparison of MAS service disciplines relative to the GPS reference model (*GPS*, 1, r_{total}). We borrow the scenario used by Blanquer *et al.* [1]. This is the same scenario used in Section 2.2.4 above, except that the flows are now sharing 4 output servers with different rates. The servers are named S1, S2, S3, and S4 with rates 0.4

bps, 0.3 bps, 0.2 bps, and 0.1 bps, respectively. The total system output rate is r_{total} = 1 bps. The same system output rate allows comparison of MAS service disciplines with GPS and with the other scheduling disciplines in MS systems. At time t = 0, all servers are assumed to be idle.

Note that we have changed the layout used in Figure 3.2 in which the vertical axis contains the servers S1, S2, S3, and S4 instead of flows F1 to F11. The size of each server is representative of the rate of server. From Figure 3.2, MSF^{2}Q produces bursty pattern for flow F1 in the sense that there are times of service inactivity for flow F1. Whenever there is service for flow F1, the flow tends to always use more than the guaranteed bandwidth of 0.5 bps.

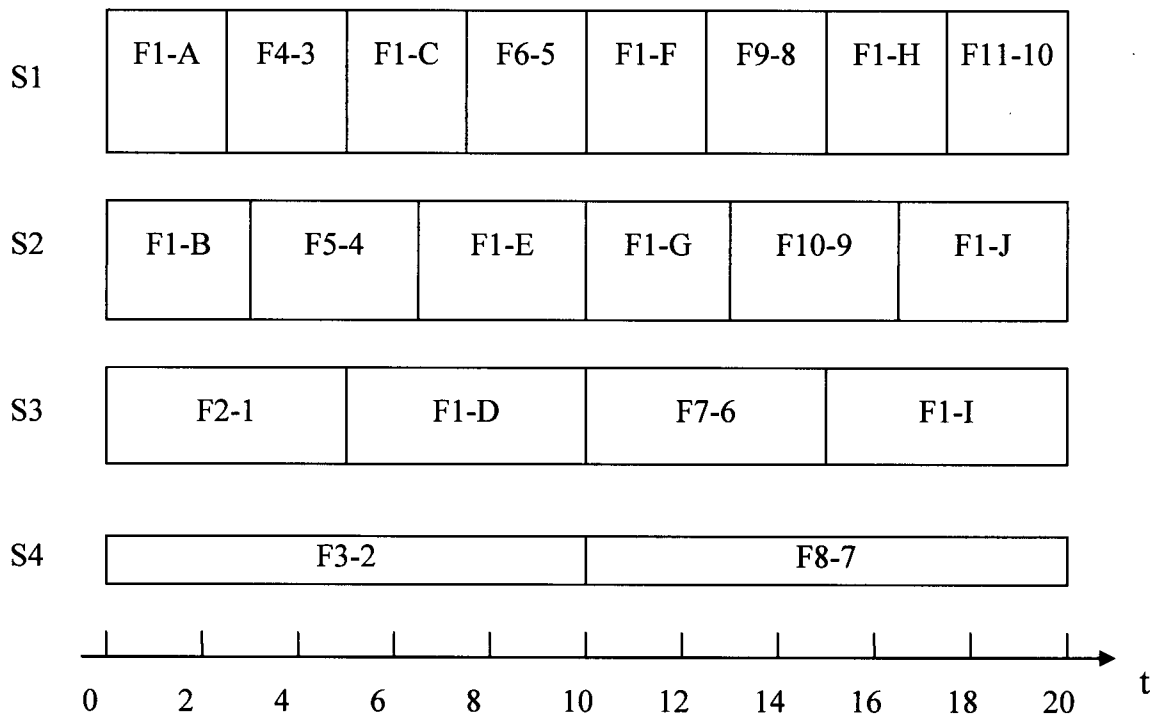


Figure 3.2: MSF^{2}Q scheduling discipline in MAS system

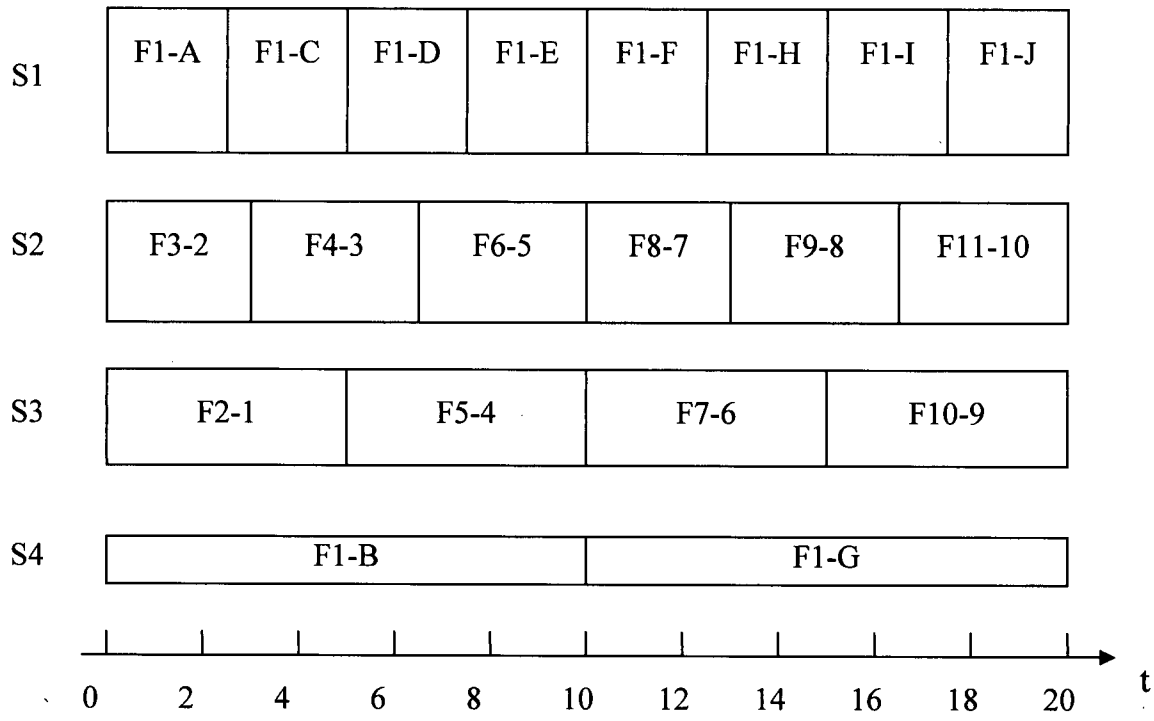


Figure 3.3: MASF^{2}Q scheduling discipline in MAS system

We use similar layout in Figure 3.3 as well. From this figure, MASF^{2}Q produces consistently smoother pattern for flow F1 in which flow F1 traffic is more evenly distributed throughout the whole busy period. Whenever there is service for flow F1, the flow tends to always track the minimum guaranteed bandwidth (which is also its current GPS bandwidth in this example) of 0.5 bps. This allows other flows to use larger pieces of the remaining available system bandwidth.

3.5 Summary

In this chapter, we modified the MSF^{2}Q to handle the Multiple Asymmetrical Server (MAS) system case. The theoretical bounds on delay and per-flow service discrepancy were derived. These bounds were also shown to reduce to the specific case of MSF^{2}Q for equal-rate server system. Then, MASF^{2}Q was introduced to improve throughput fairness by including rules selecting packet-server pairings at any scheduling instances. Packet-server pairing matches packet of an eligible flow to an idle server of a specific rate to approximate better

GPS instantaneous throughput. Theoretical example was given to illustrate the better throughput fairness property. The next chapter presents simulation results as comparison of various multiple-server service disciplines applied onto a couple of traffic scenarios.

CHAPTER 4 SIMULATION RESULTS

A simulation model was used to assess the resulting fairness property of the new scheme compared to MSF^2Q, MSFQ, and WFQ in MAS systems. The software used to implement the simulation is OPNET Modeler ver. 9.1.A [16]. The simulation models were developed by including additional features onto the pre-existing acb-fifo queue-model that comes with the software.

The simulation scenario consists of 5 traffic flows sharing 6 servers for a total simulation period of 120 minutes. All flows start generating packets after the first 10 seconds. Each flow, indexed 0 to 4, has its own flow weight which represents the minimum guaranteed rate as a percentage of the total system bandwidth. Similarly, each server has a server weight which denotes the rate of the server as a percentage of the total system bandwidth. The servers have the following percentage of the total bandwidth: 30%, 30%, 20%, 10%, 5%, and 5%, respectively – making a total of 1 Mbps.

In order to measure the fairness property relative to GPS, we keep track of the change in *instantaneous bandwidth* of each flow every time an event occurrs that changes either the bandwidth share of flows in packetized system or GPS system. Since all the service disciplines in the simulation are based on virtual time (simulation of GPS in background), keeping track of bandwidth for analysis simply requires additional storage for the packetized flow rate of each flow.

4.1 Simulation Scenario 1

In this scenario, the traffic generation pattern was continuous – meaning that packets of all flows could be generated at any time within the simulation period. The simulation period started at the 10-second mark and ended at the 2-hour mark. The statistical packet generation parameters included specifications of packet lengths and flow rates. We assumed that each flow generated Poisson-type traffic arrival. Flow 0 had an average inter-arrival time of 0.02 seconds, while each of the other flows had an average inter-arrival time of 0.10 seconds. The inter-arrival times had an exponential distribution. Packets of all flows had an average packet length of 10 kb with exponential distribution. This means that flow 0 had a weight of

approximately 55.56%, while each of the remaining flows had 11.11%. In this scenario, the maximum length packet for the duration of the simulation had a size of 132,235 bits.

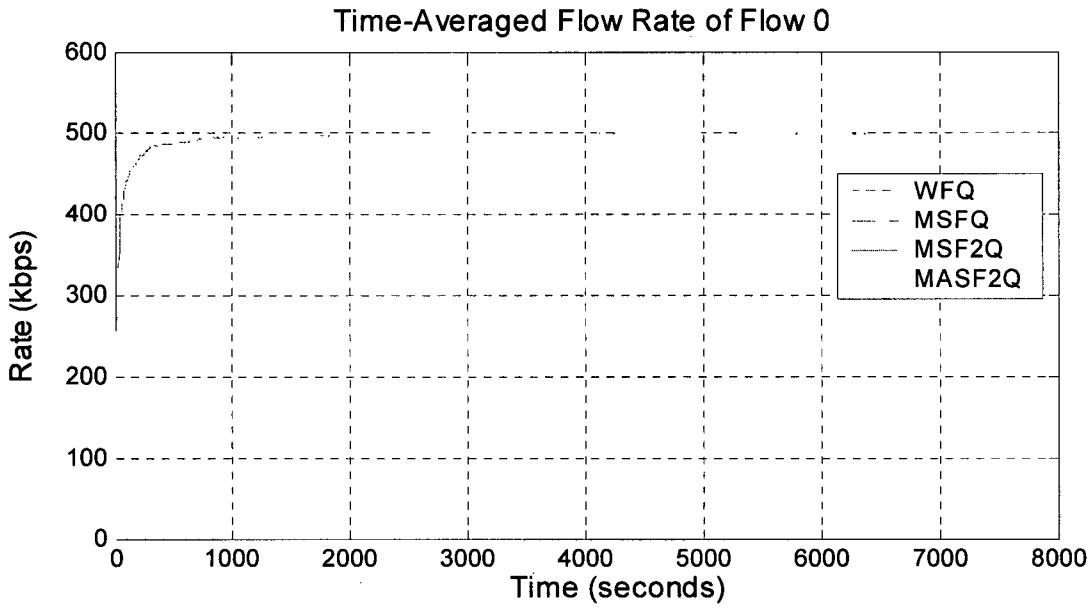


Figure 4.1: Time-averaged flow rate of flow 0

The first step was to verify that all schemes (MASF^{2}Q, MSF^{2}Q, MSFQ, and WFQ) indeed allowed the flows to achieve their guaranteed average rates. We presented the average flow rate results for all flows. In Figure 4.1, flow 0 achieved its desired average rate of 500 kbps within seconds of the start of simulation. The rate was calculated by dividing the amount of service received within a small enough duration over the time duration itself. Figure 4.2 to 4.5 showed other flows achieving their desired average flow rate of 100 kbps each.

approximately 55.56%, while each of the remaining flows had 11.11%. In this scenario, the maximum length packet for the duration of the simulation had a size of 132,235 bits.

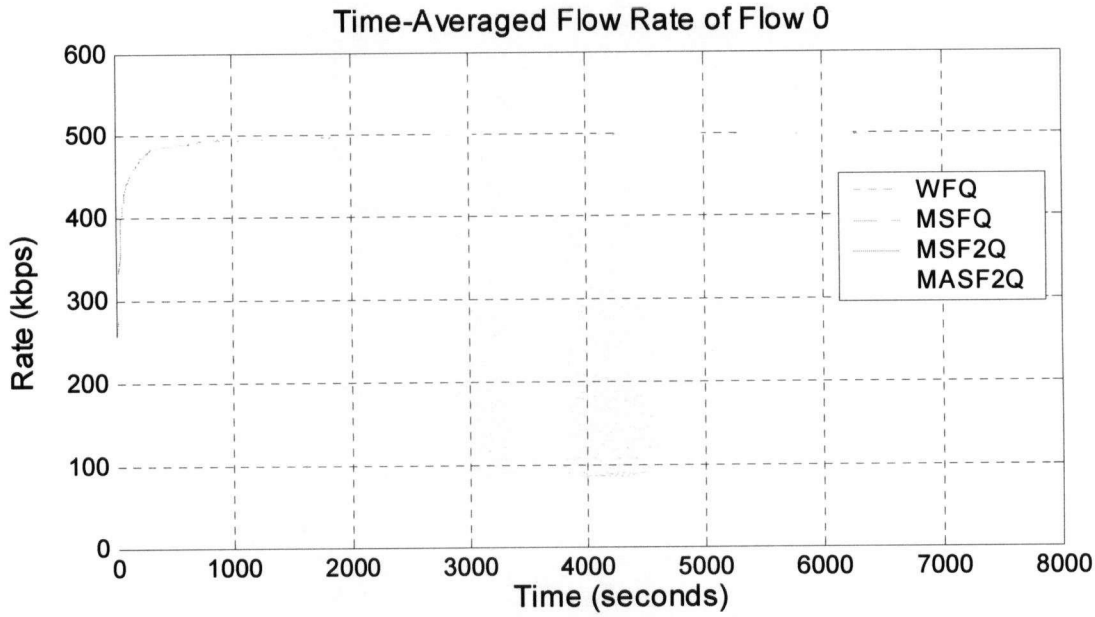


Figure 4.1: Time-averaged flow rate of flow 0

The first step was to verify that all schemes (MASF^{2}Q, MSF^{2}Q, MSFQ, and WFQ) indeed allowed the flows to achieve their guaranteed average rates. We presented the average flow rate results for all flows. In Figure 4.1, flow 0 achieved its desired average rate of 500 kbps within seconds of the start of simulation. The rate was calculated by dividing the amount of service received within a small enough duration over the time duration itself. Figure 4.2 to 4.5 showed other flows achieving their desired average flow rate of 100 kbps each.

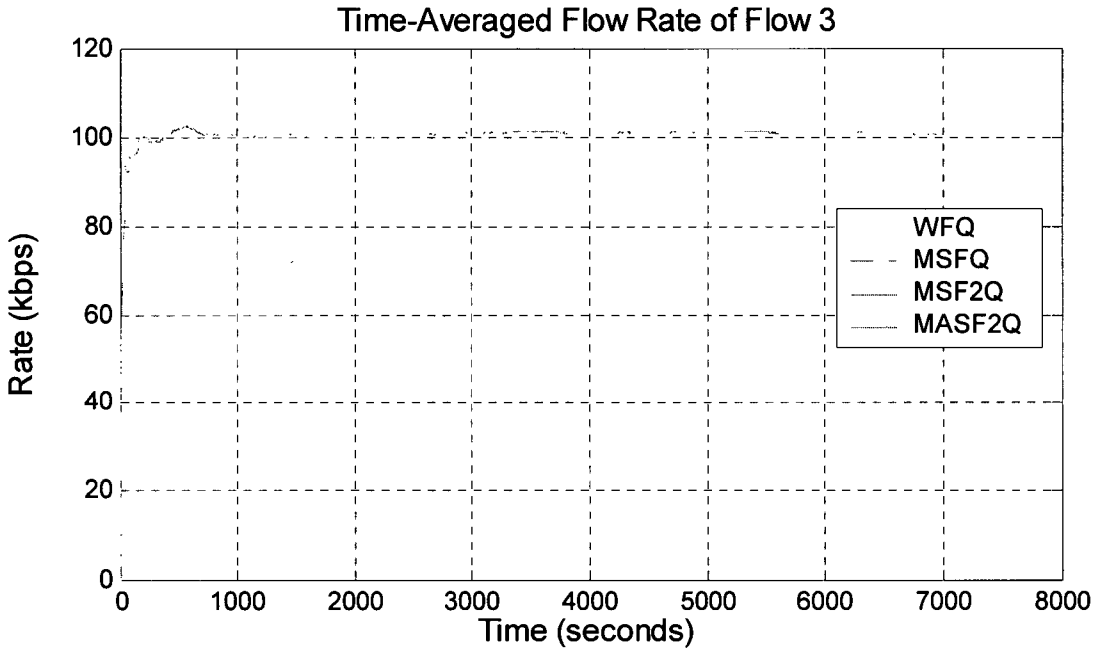


Figure 4.4: Time-averaged flow rate of flow 3

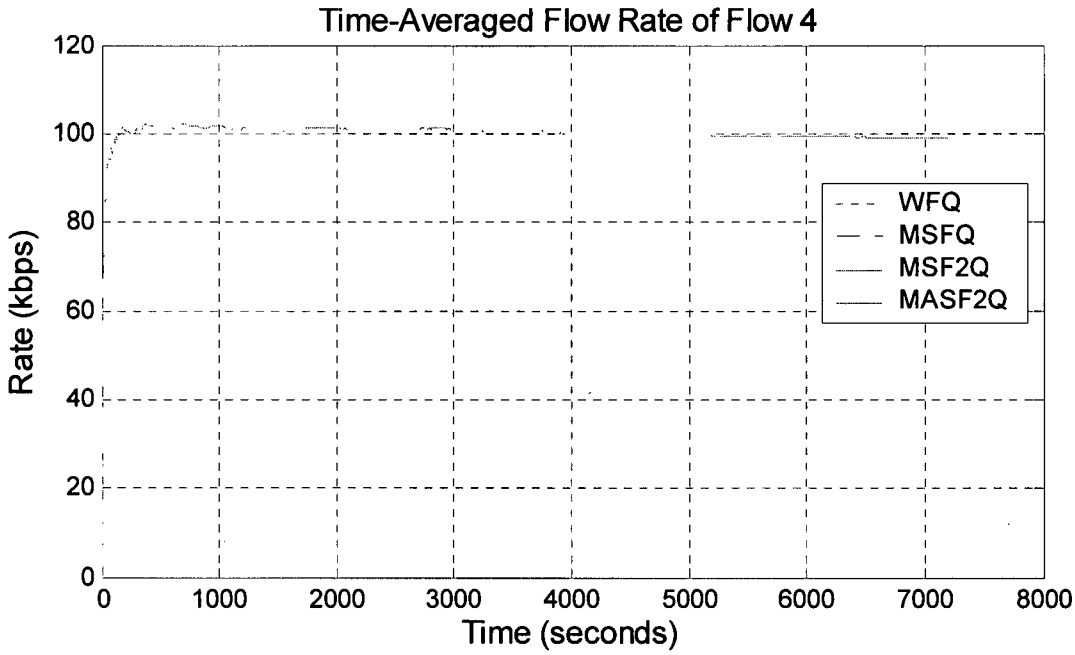
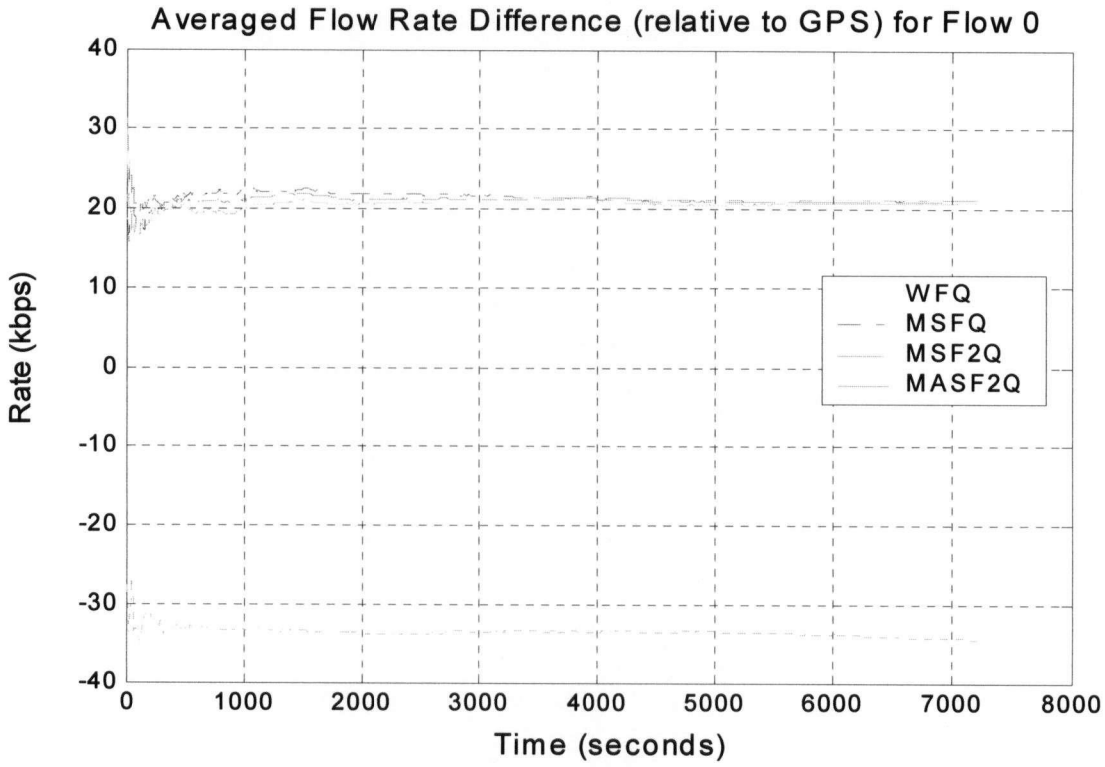


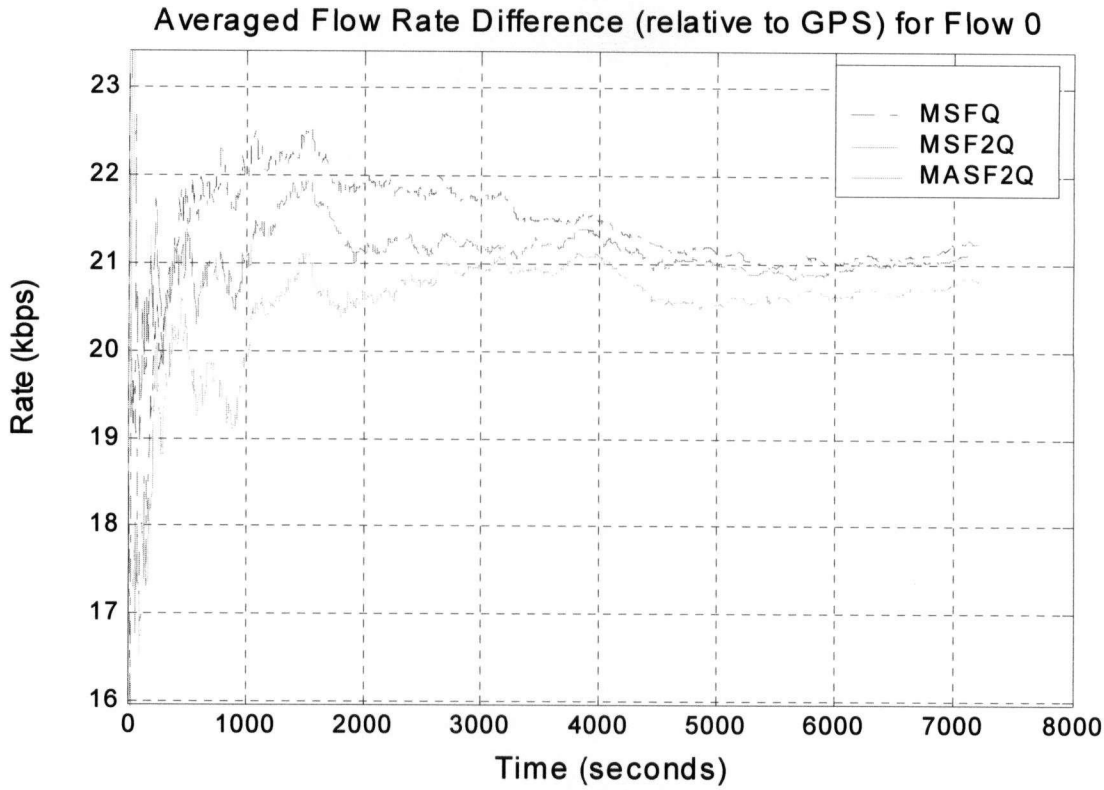
Figure 4.5: Time-averaged flow rate of flow 4

To better illustrate how MASF^{2}Q may improve fairness in MAS systems, we measured the bandwidth difference (in bps) between GPS flow rate and packetized flow rate, or $r_i(\tau) - \hat{r}_i\ (\tau)$, every time any one of these rates changed. This gave a snapshot of how fair

each scheme was toward every flow compared to GPS. Due to the large number of data points generated, useful information can be deduced by averaging the results over small time intervals. Theoretically, for the packetized schemes, the closer the line plots of flow rate difference were to zero (or time axis), the better the fairness achieved relative to GPS. The trends in flows 0 to 2 (Figures 4.6 to 4.8) showed that MASF^{2}Q was at least similar in fairness compared to MSF^{2}Q, and, for the most part, approached GPS more closely. These plots suggested that a suitable packet-server pairing could allow MASF^{2}Q to achieve better fairness. However, the trends in flows 3 and 4 (Figures 4.9 and 4.10) showed that MASF^{2}Q was equal or somewhat worse in fairness compared to MSF^{2}Q. This seemed to suggest that not all flows enjoyed similar improvements in fairness under MASF^{2}Q. It might also suggest that some flows performed slight worse for others to improve. This could have been caused by only one server becoming idle at any one time most of the time. This limited the efficiency of packet-server pairing because it needed at least two or more idle servers to choose from. In simulation scenario 2 below, we would try to investigate this phenomenon by looking at a bursty traffic pattern that created a lot more instances where two or more servers were idle.

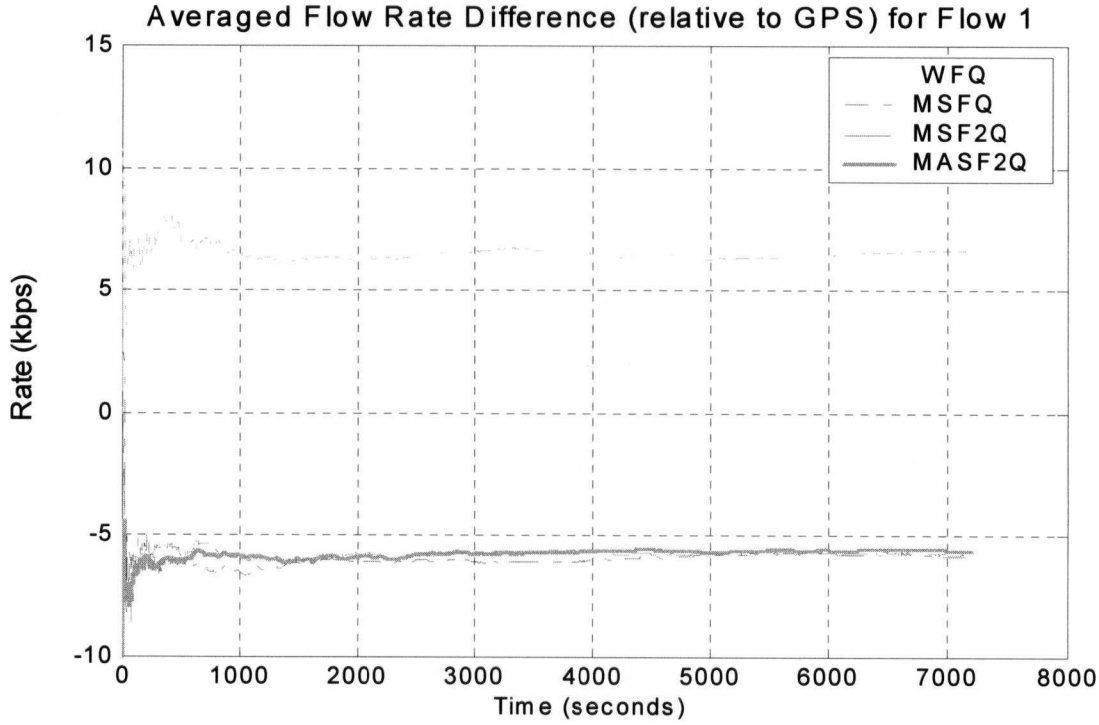


(a)

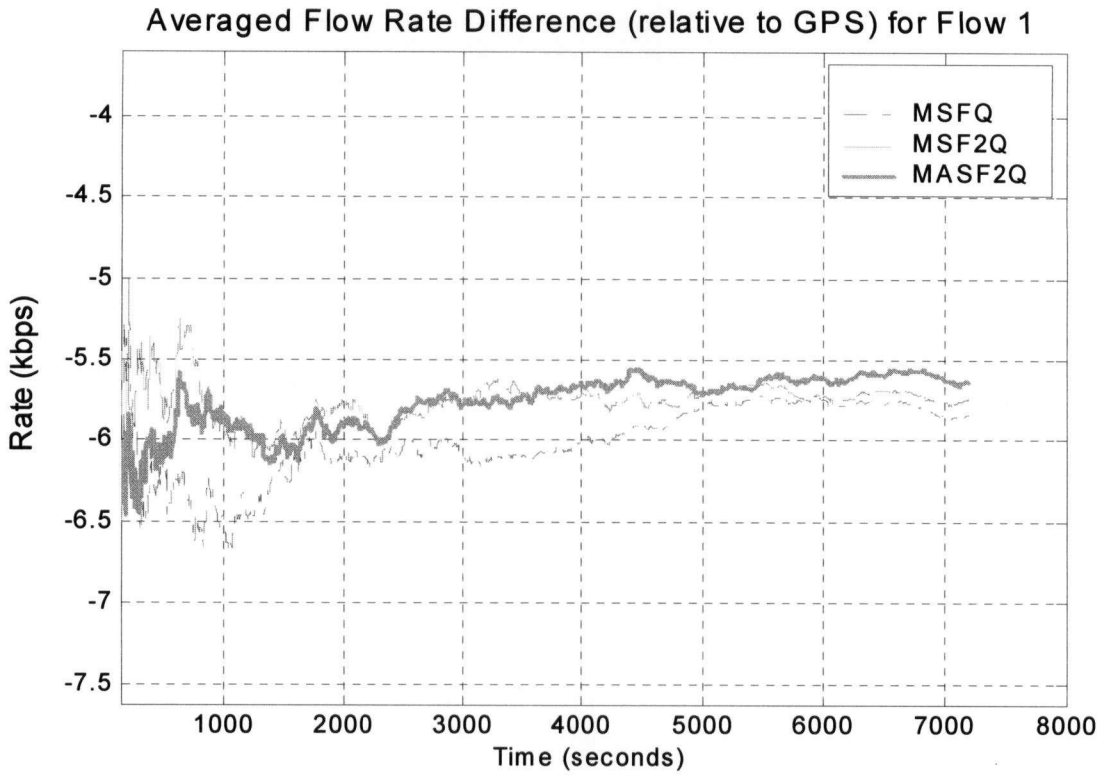


(b)

Figure 4.6: Flow rate difference of flow 0 relative to GPS system: (a) overall view showing WFQ, and (b) detailed view.

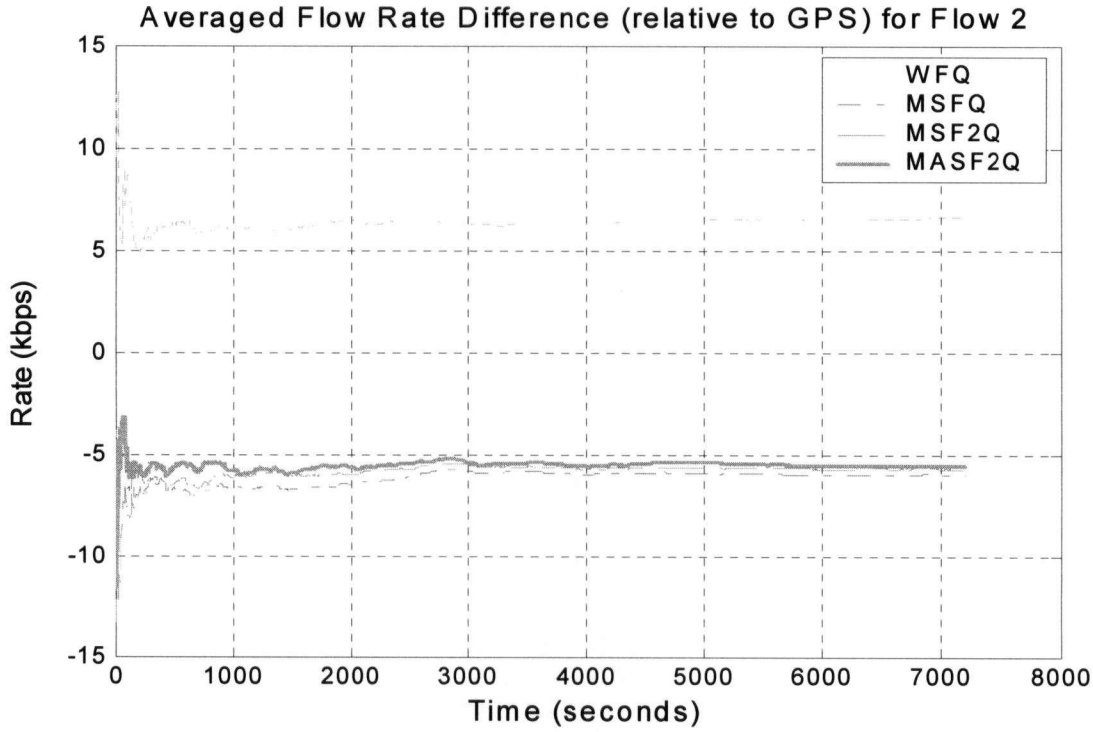


(a)

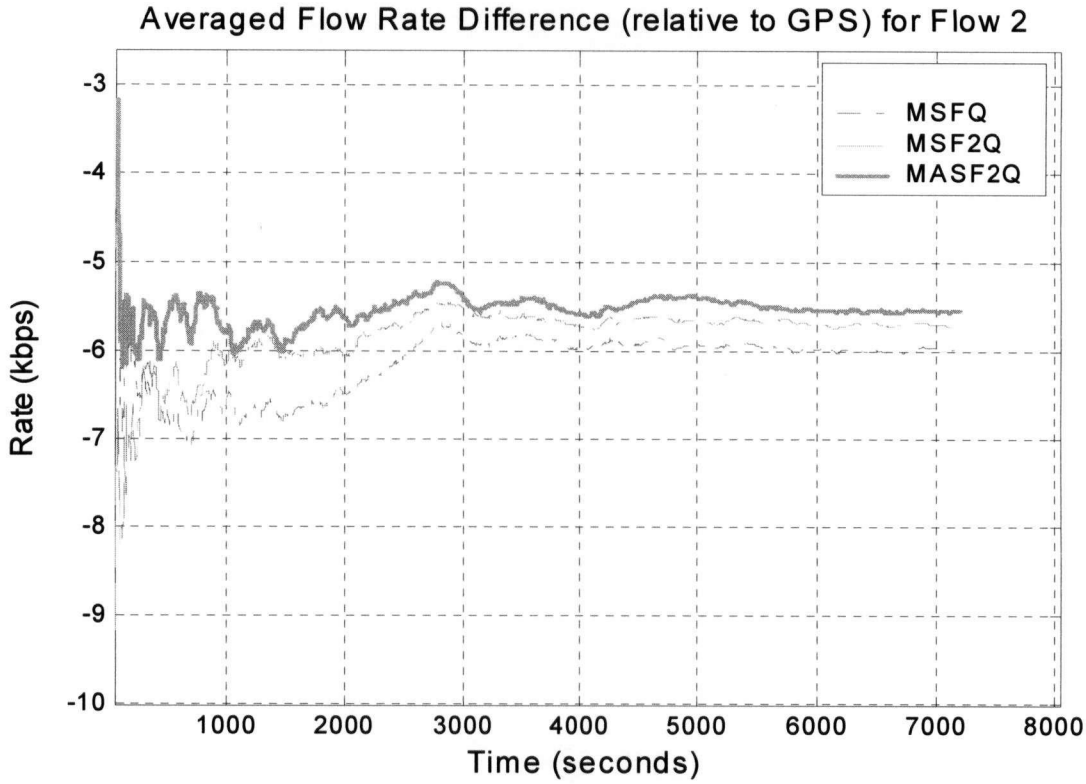


(b)

Figure 4.7: Flow rate difference of flow 1 relative to GPS system: (a) overall view showing WFQ, and (b) detailed view.

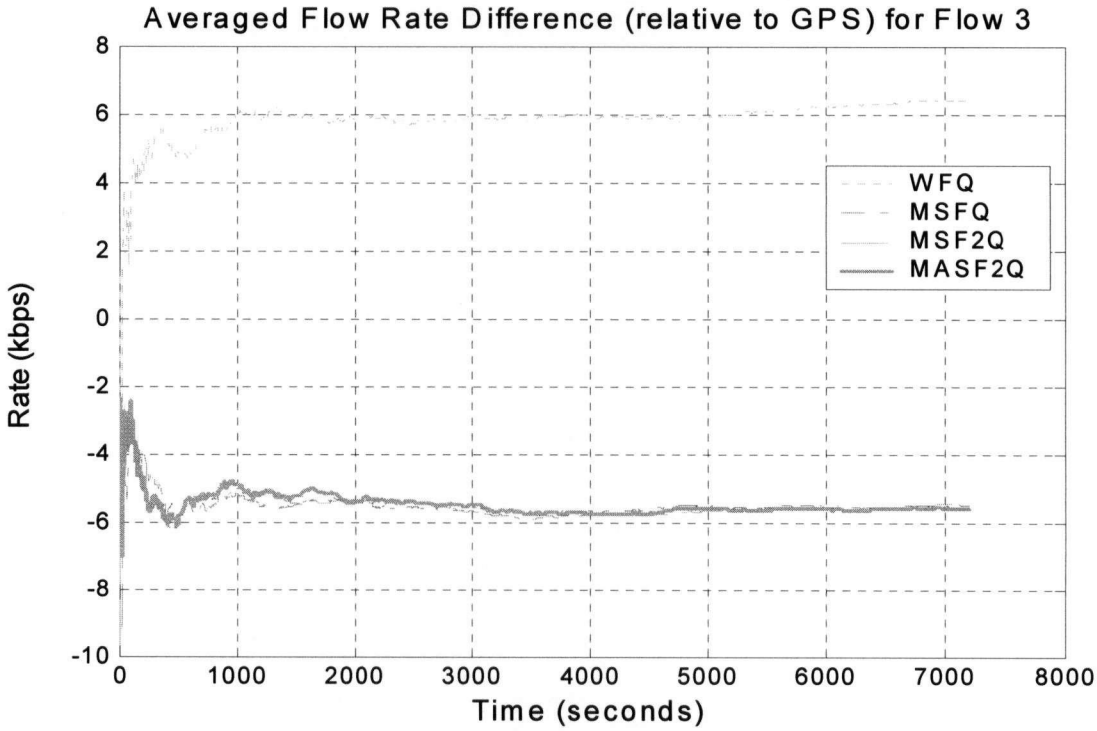


(a)

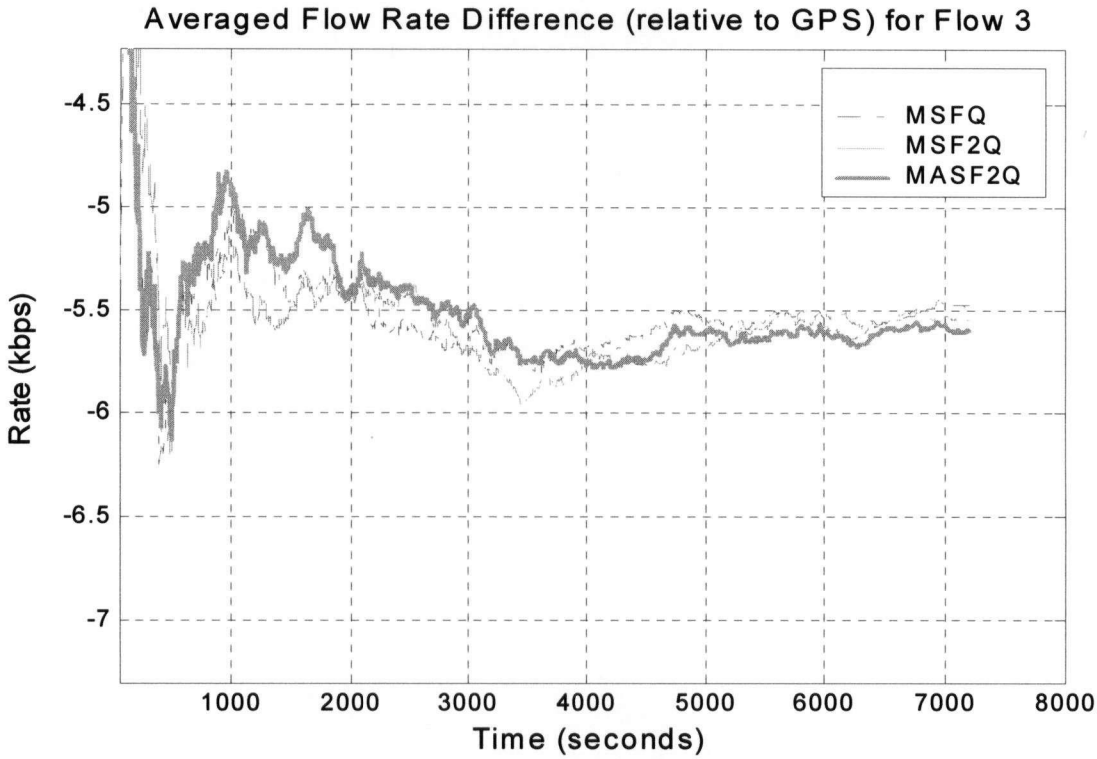


(b)

Figure 4.8: Flow rate difference of flow 2 relative to GPS system: (a) overall view showing WFQ, and (b) detailed view.

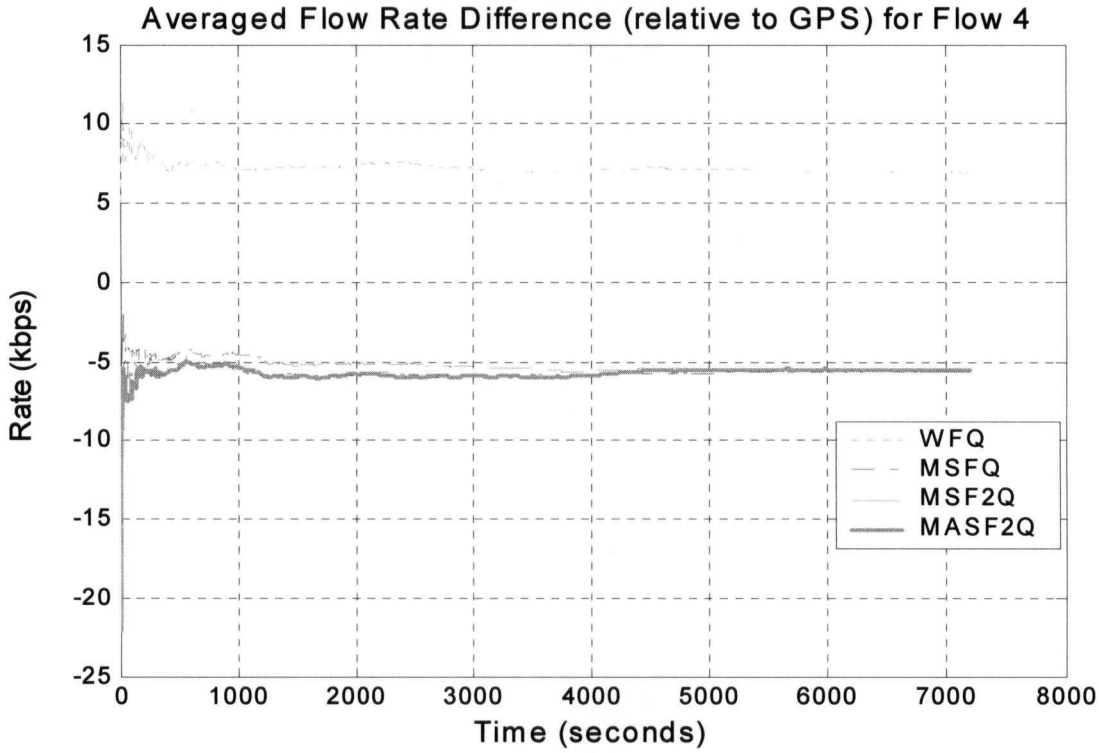


(a)

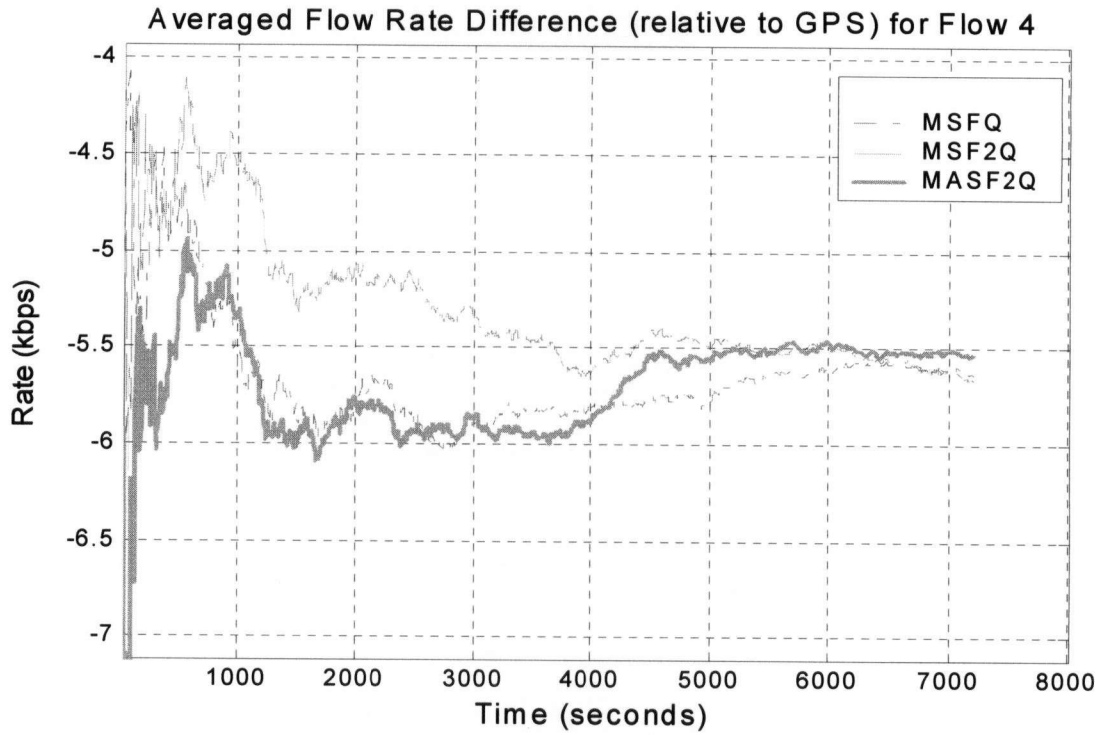


(b)

Figure 4.9: Flow rate difference of flow 3 relative to GPS system: (a) overall view showing WFQ, and (b) detailed view.



(a)

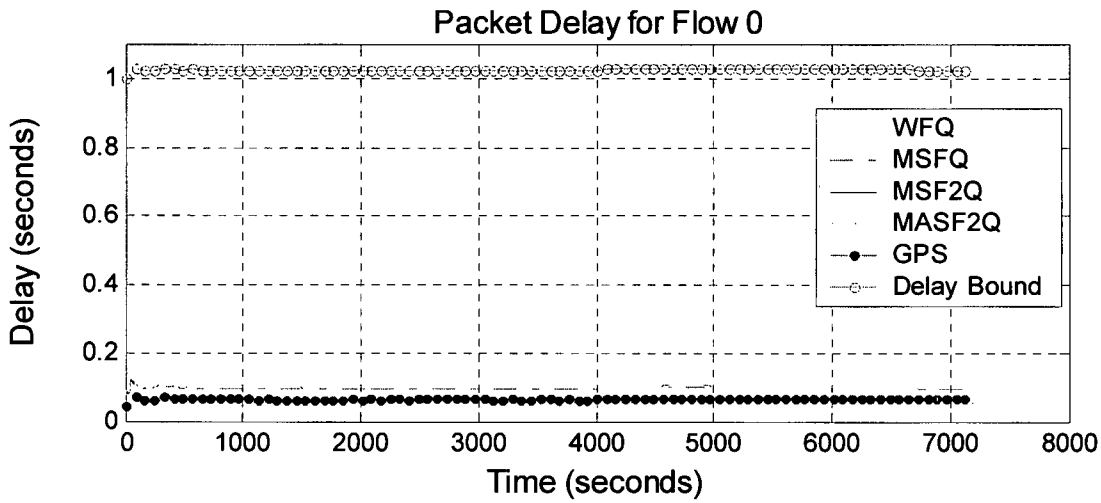


(b)

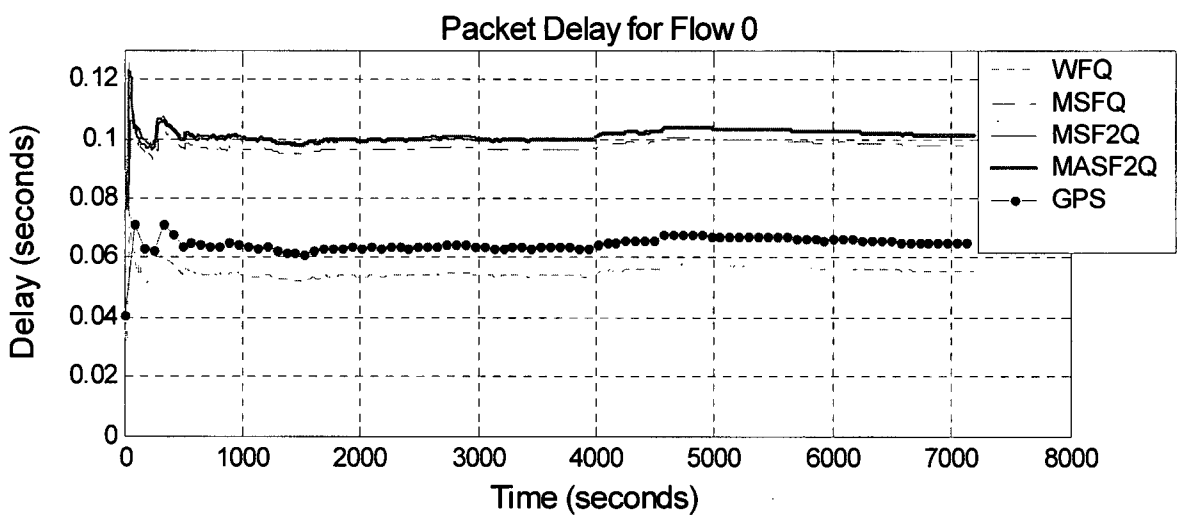
Figure 4.10: Flow rate difference of flow 4 relative to GPS system: (a) overall view showing WFQ, and (b) detailed view.

As for packet delay, MASF^{2}Q maintained very similar performance for all flows compared to the original MSF^{2}Q scheme (Figures 4.11 to 4.15). As expected, this showed that the packet-server pairing rules did not affect packet delay in any significant way. Although not significant, MASF^{2}Q packet delay tends to be slightly larger than that of MSF^{2}Q. Also, MASF^{2}Q and MSF^{2}Q packet delays are both larger than that of MSFQ. MSFQ had a better average packet delay because it was a work-conserving scheme. Idle servers would be utilized immediately by backlogged flows, and packets would start and finish service sooner. On the other hand, MSF^{2}Q eligibility conditions sacrifice this work-conservation to gain fairness in service completed; thus, resulting in increase of packet delay. The additional rules introduced onto MASF^{2}Q in this thesis are meant to provide *better* fairness compared to MSF^{2}Q through another form of sacrifice. Because of these additional rules, packets of eligible backlogged flows were no longer always served starting from the fastest idle servers available. This caused the average packet delay to worsen slightly compared to MSF^{2}Q. MASF^{2}Q packet delay was well within the worst-case packet delay bound as derived in Theorem 2 of Section 3.2. The packet delay bound was then 0.96008 seconds larger than the GPS packet delay.

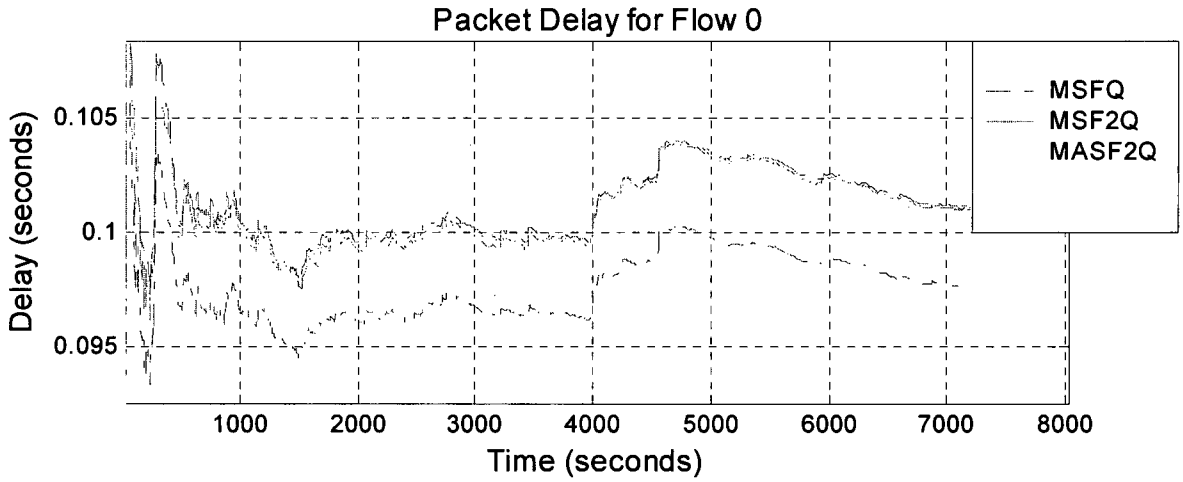
Compared to those of single-server schemes, all the multi-server schemes here had greater overall delays. This was caused by packets in WFQ and GPS finishing much earlier because the packets tend to occupy up to full bandwidth of the system. This was not true for the multi-server schemes because total bandwidth was fragmented into a number of servers with fixed rates. Unless there were more packets backlogged than servers available, the full bandwidth would not be utilized.



(a)

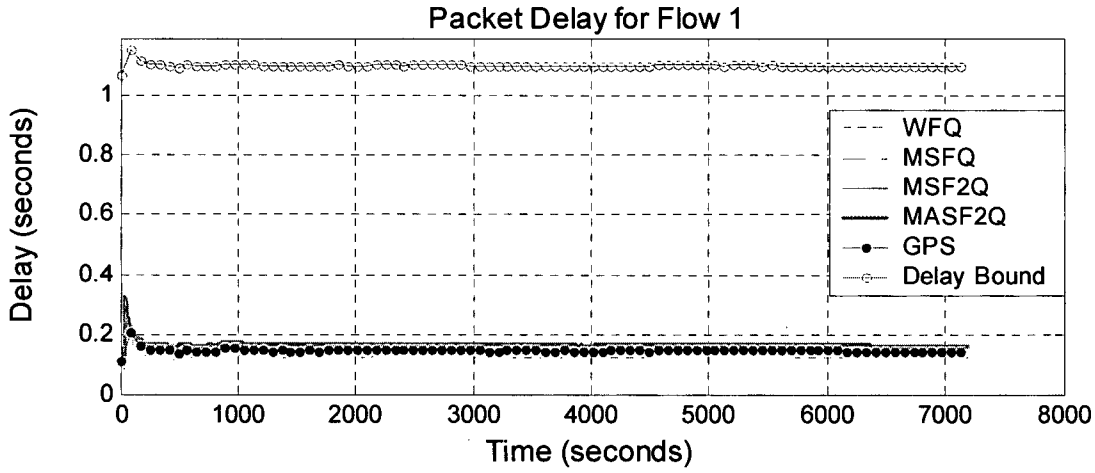


(b)

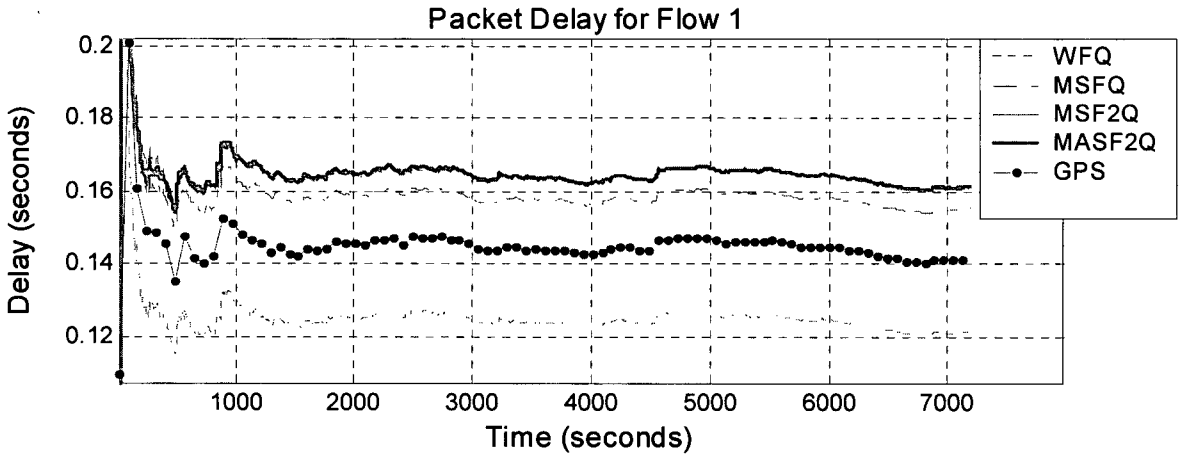


(c)

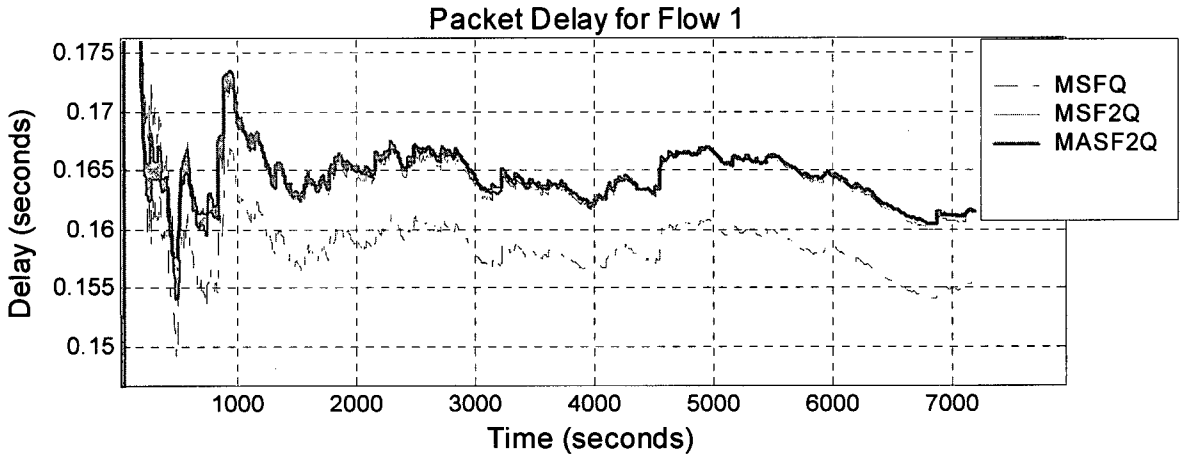
Figure 4.11: Average packet delay for flow 0: (a) overall view showing Delay Bound; (b), (c) detailed view



(a)

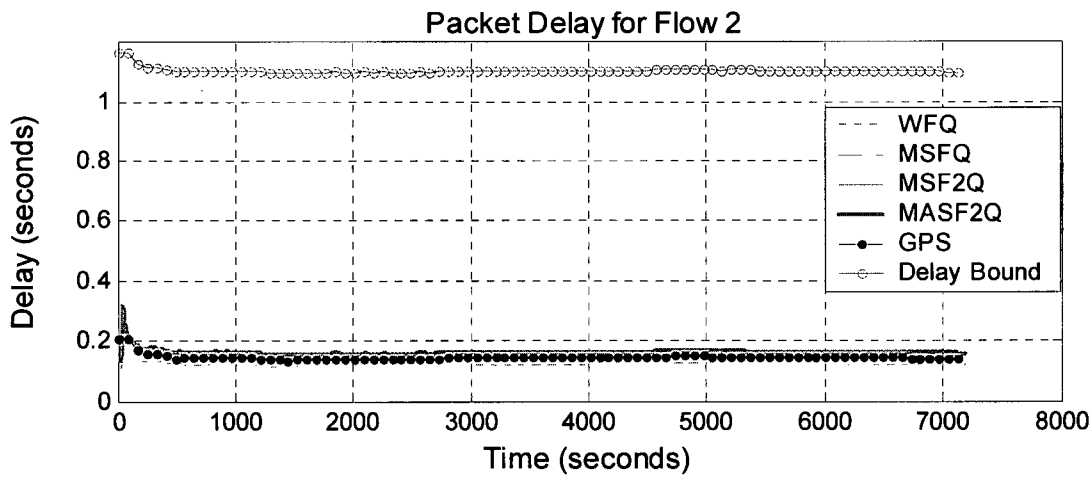


(b)

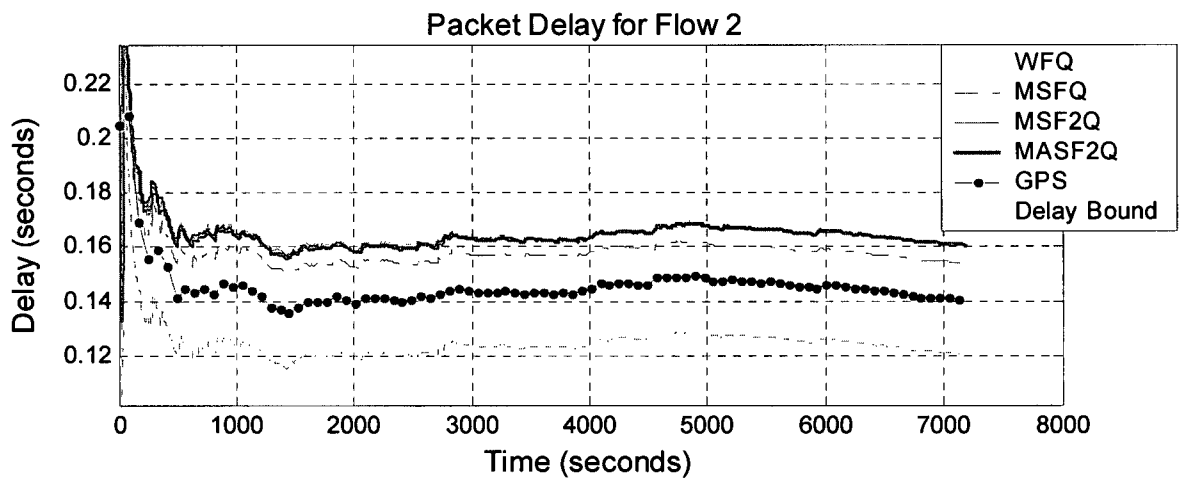


(c)

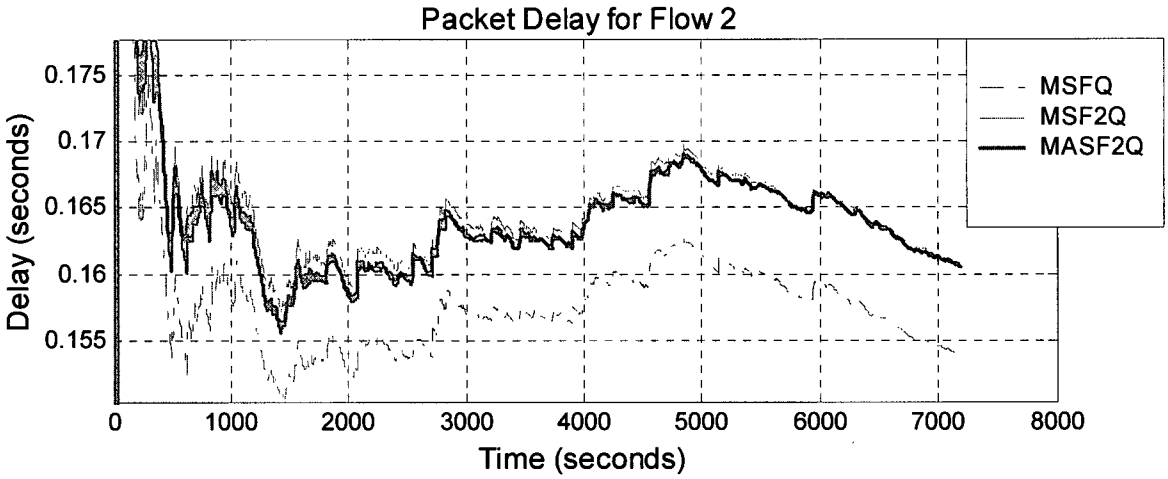
Figure 4.12: Average packet delay for flow 1: (a) overall view showing Delay Bound; (b), (c) detailed view



(a)

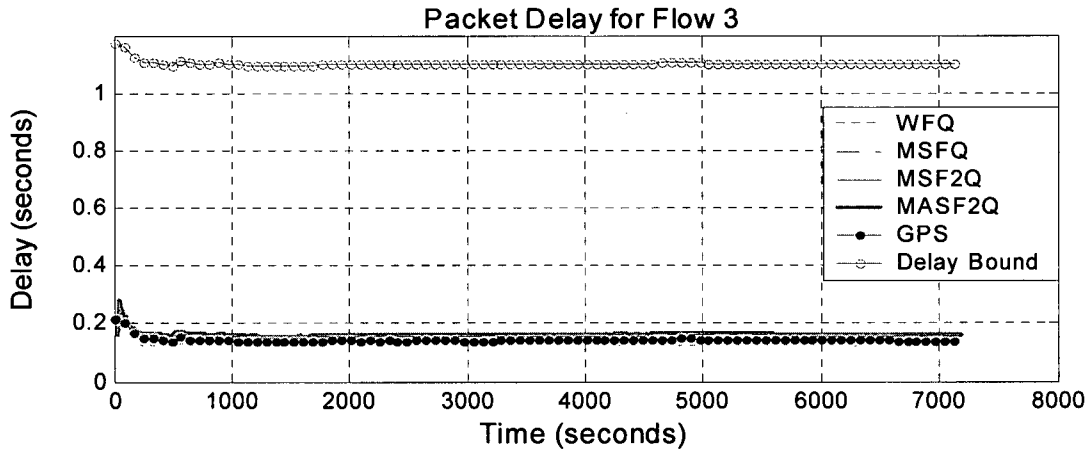


(b)

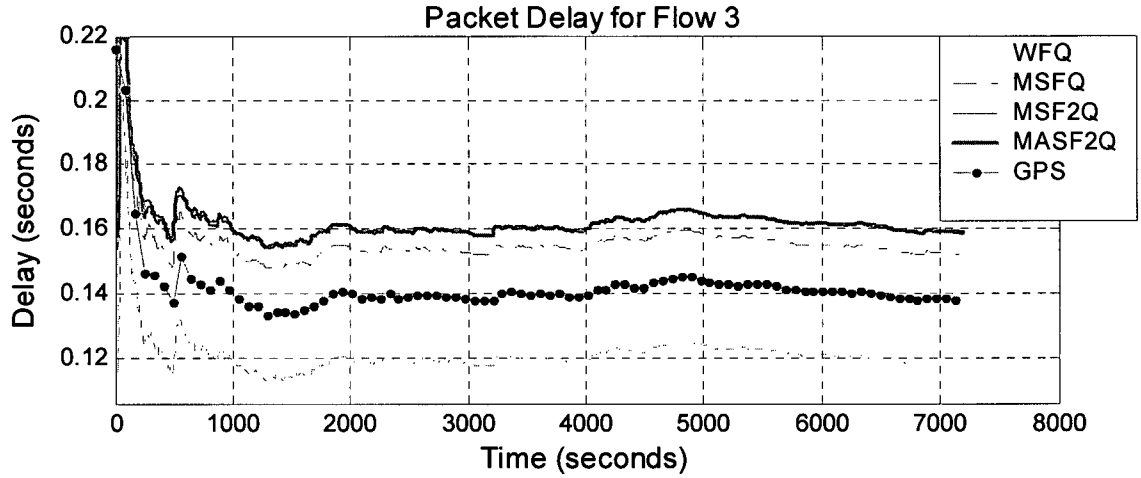


(c)

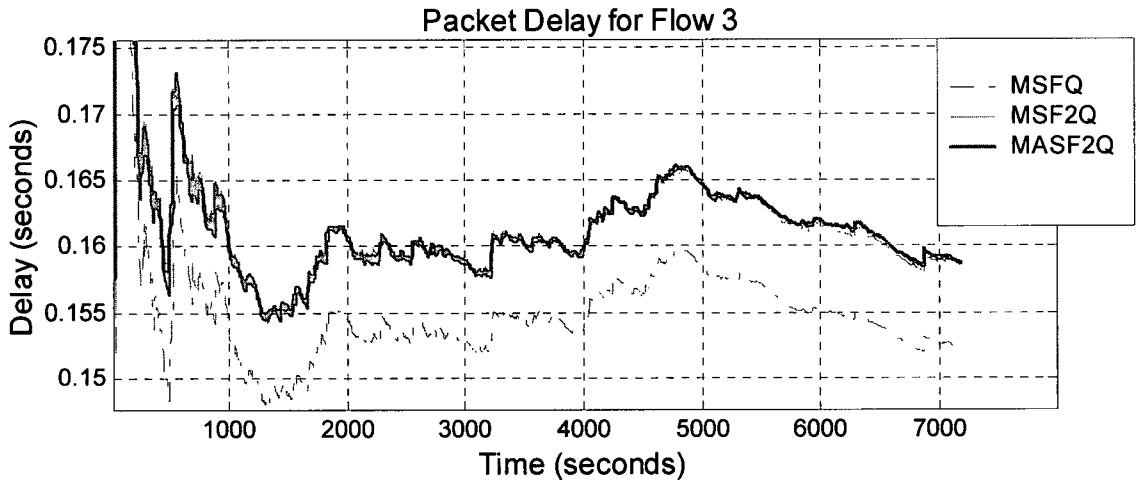
Figure 4.13: Average packet delay for flow 2: (a) overall view showing Delay Bound; (b), (c) detailed view



(a)

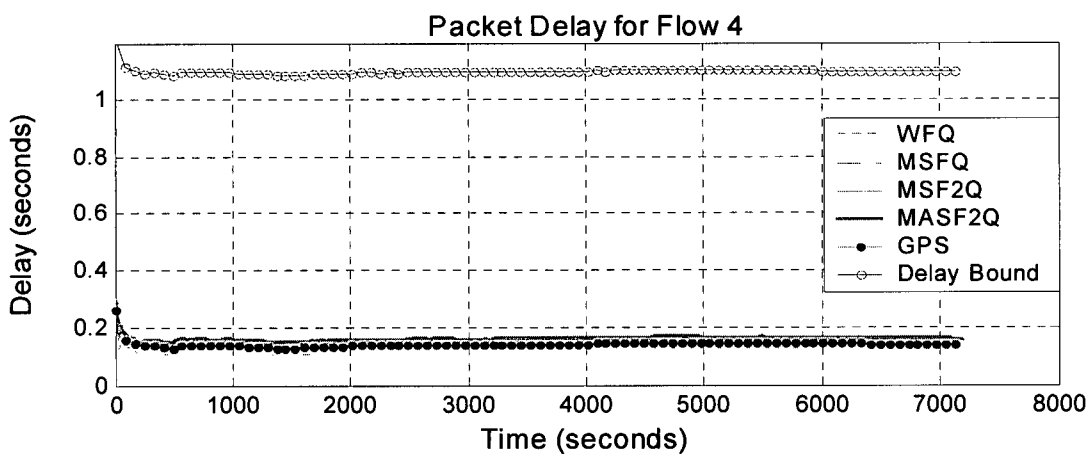


(b)

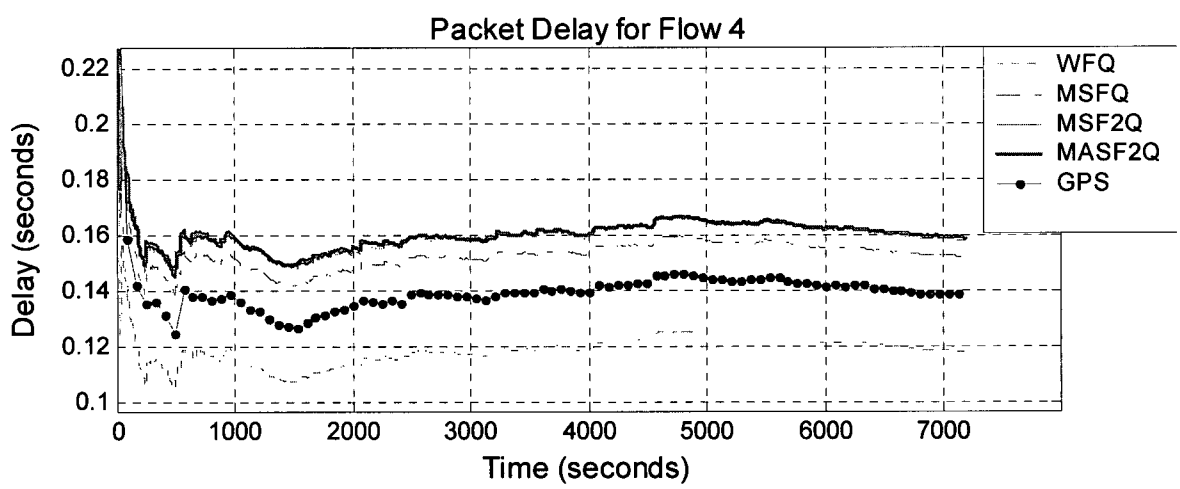


(c)

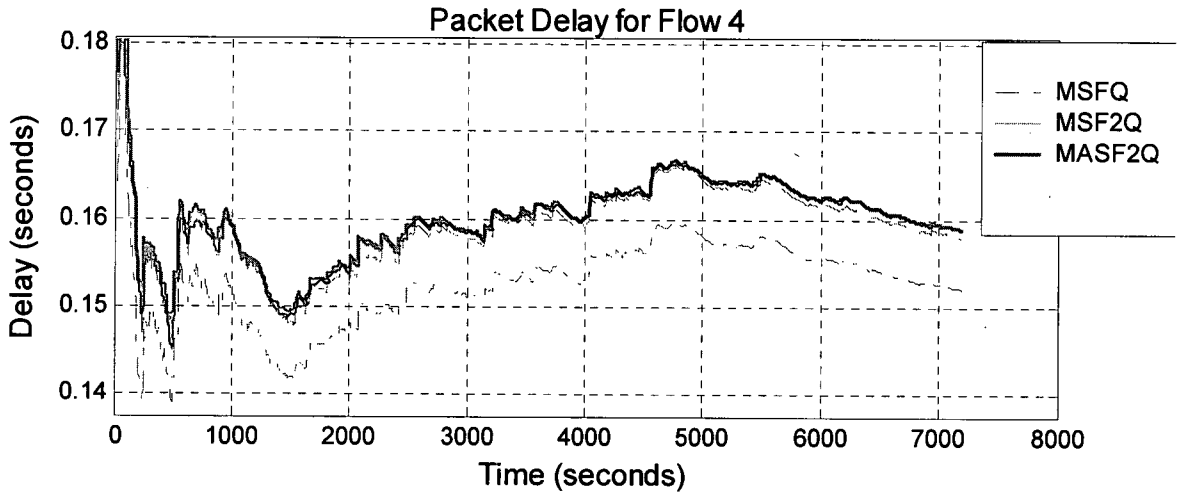
Figure 4.14: Average packet delay for flow 3: (a) overall view showing Delay Bound; (b), (c) detailed view



(a)



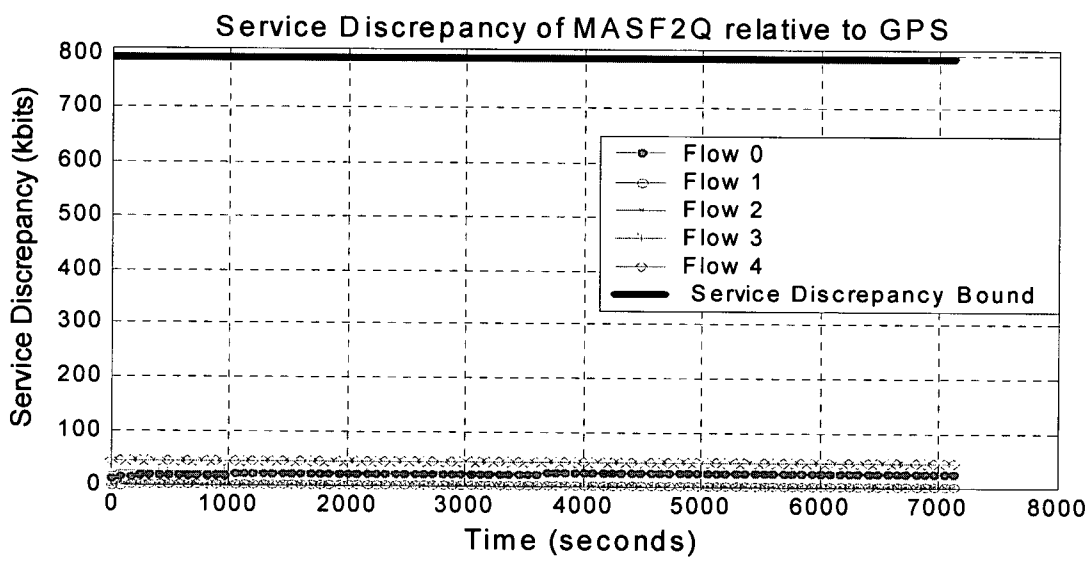
(b)



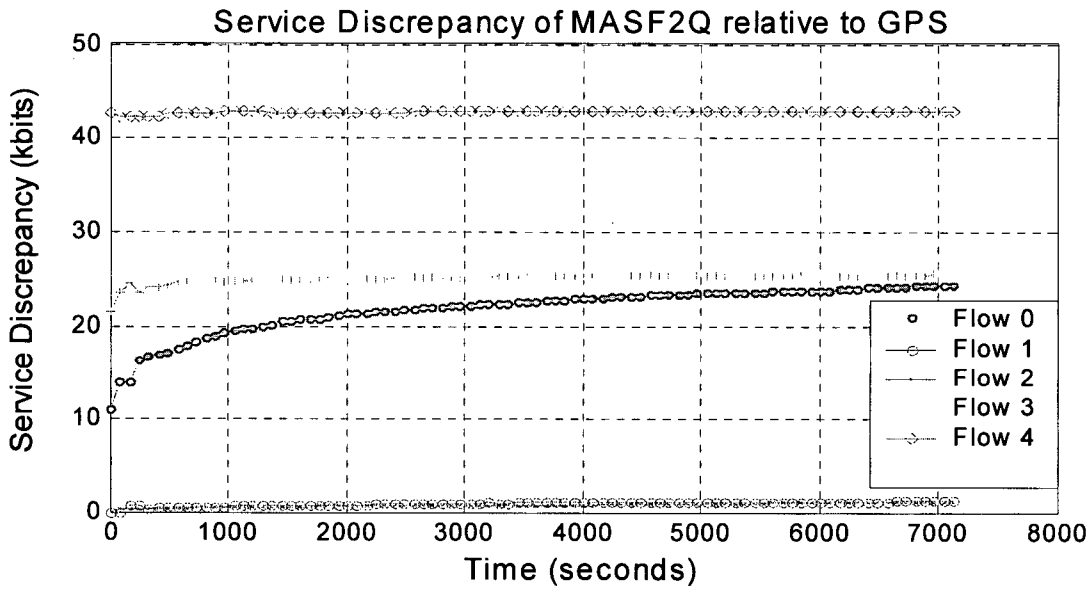
(c)

Figure 4.15: Average packet delay for flow 4: (a) overall view showing Delay Bound; (b), (c) detailed view

As shown in Figure 4.16, the service discrepancy for all flows reached stable levels and fell well within the worst-case bound as derived in Theorem 3 in Section 3.2. Despite having the same bandwidth requirements, flows 1 to 4 seemed to display different service discrepancy results / behaviour (Figure 4.16 (b)). This could be due to the order in which these flows were served in MASF^{2}Q system. Flows were served in order of increasing flow numbers. Thus, flows 3 and 4 tend to always get served later than flows 1 and 2. This could cause the service of flows 3 and 4 to always fall more behind that of their GPS counterparts.



(a)



(b)

Figure 4.16: Service discrepancy and its bound for all flows in MASF^{2}Q: (a) overall view; (b) detailed view.

4.2 Simulation Scenario 2

In this scenario, the traffic generation pattern was bursty – meaning that packets of all flows could be generated only during specific time intervals within the simulation period. The simulation period started at the 10-second mark and ended at the 2-hour mark. The bursty pattern was generated using OPNET's ON-OFF traffic pattern generator. This scenario used ON-periods of 15 seconds each and OFF-periods of 30 seconds each. Starting from the 10-second mark, the ON-OFF periods occurred repeatedly one after another in a fixed interval pattern for all the flows. Packets could only arrive during the ON periods, but packets were served continuously. The statistical packet generation parameters included specifications of packet lengths and flow rates. We assumed that each flow generated Poisson-type traffic arrival. All flows had average inter-arrival times of 0.05 seconds with exponential distributions. Packets of all flows had fixed lengths of 10 kb. This scenario was to study whether the same packet-server pairing method would perform better in a burstier environment. It was speculated that burstier traffic pattern would create more instances where two or more servers would be free at the same time. This would allow the packet-server pairing algorithm to choose how to efficiently allocate servers to serve waiting packets.

However, based on the simulation results, we were unable to conclude that the algorithm worked better under bursty traffic. Averaged flow rate difference plots showed that slight improvements in service fairness for some of the flows (in this case flows 1 and 4) actually came at the expense of other flows. The desired result was all the flows received better fairness. Although using a bursty pattern, still only one server might have become idle at any one time most of the time when packets were backlogged and ready to be served. In effect, this limited the desired efficiency of the packet-server pairing that needed two or more idle servers to choose from.

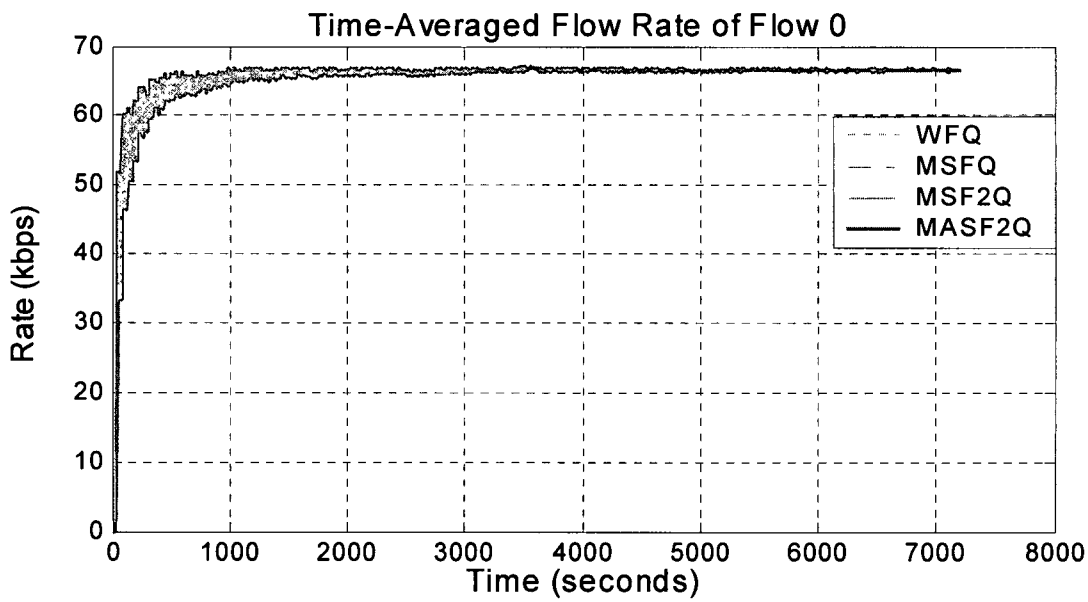


Figure 4.17: Time-averaged flow rate of flow 0

The first step was to verify that all schemes (MASF^{2}Q, MSF^{2}Q, MSFQ, and WFQ) indeed allowed the flows to achieve their desired average rates in bps. We presented the average flow rate results for all flows. In Figures 4.17 to 4.21, flows 0 to 4 achieved their desired average rate of approximately 67 kbps within seconds of the start of simulation. Note that all flows had the same shares of the total system bandwidth, and that the flows were ON only 1/3 of the time. Thus, 1 Mbps divided by 5 and multiplied by 1/3 resulted in 67 kbps.

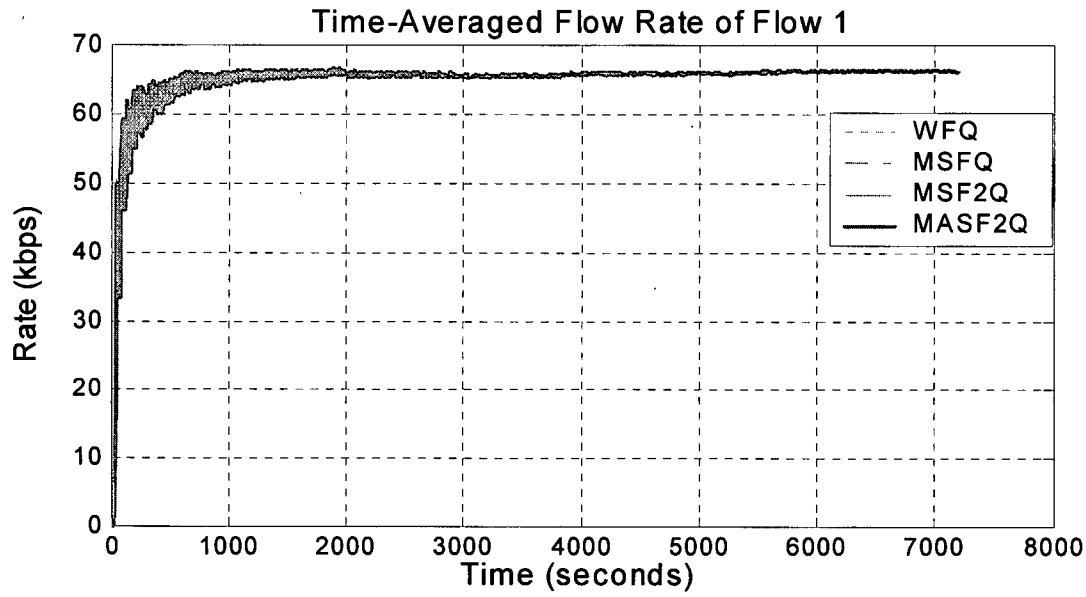


Figure 4.18: Time-averaged flow rate of flow 1

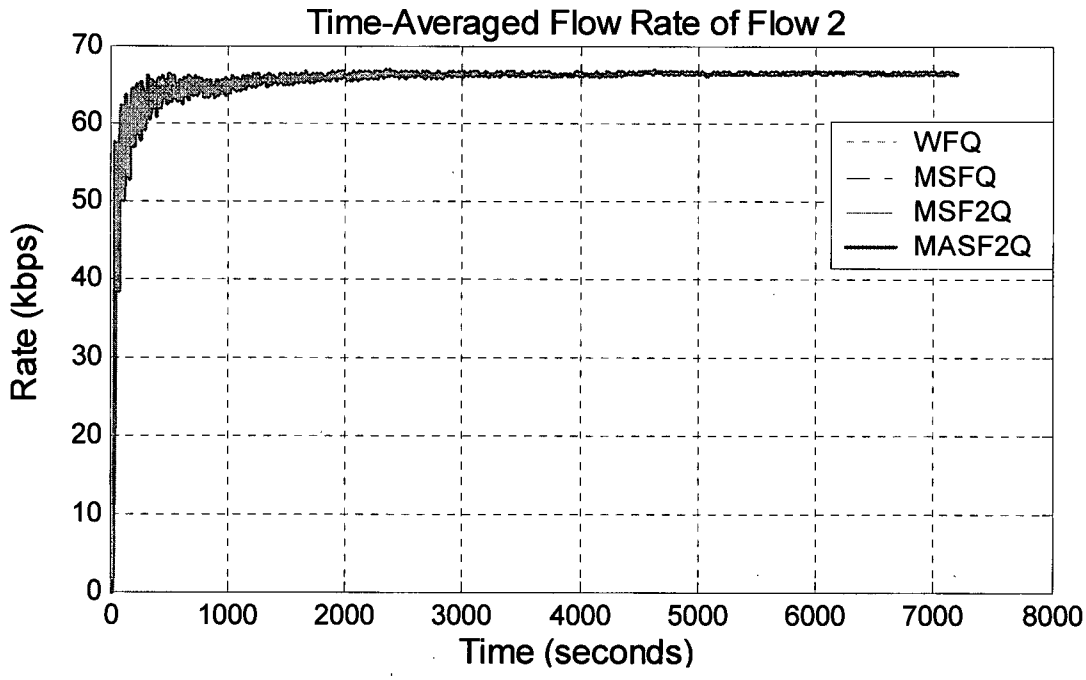


Figure 4.19: Time-averaged flow rate of flow 2

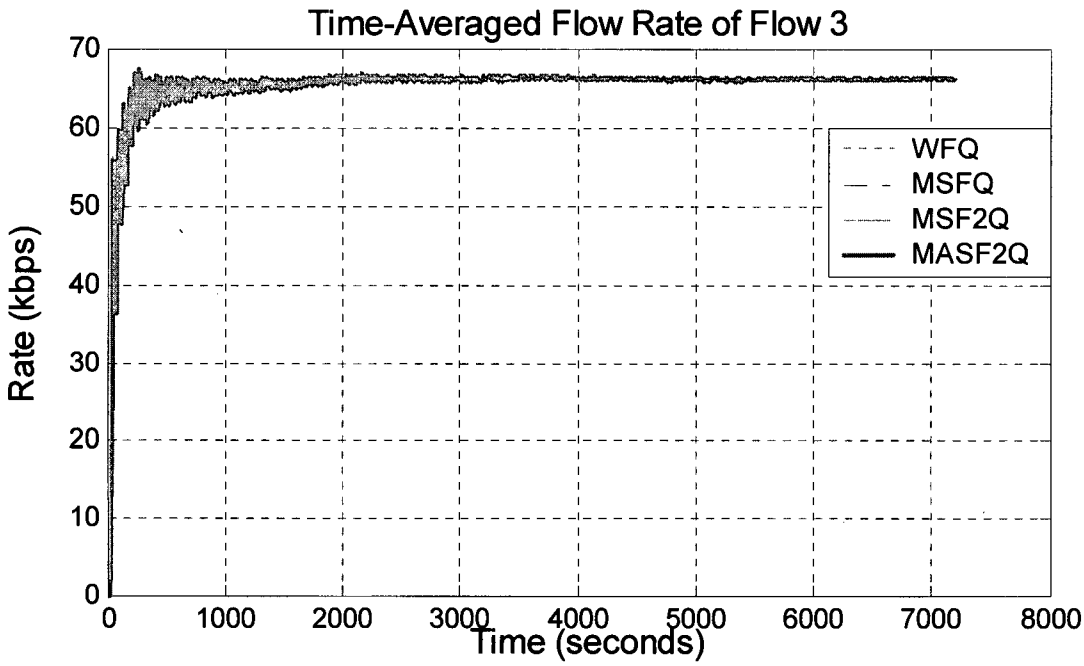


Figure 4.20: Time-averaged flow rate of flow 3

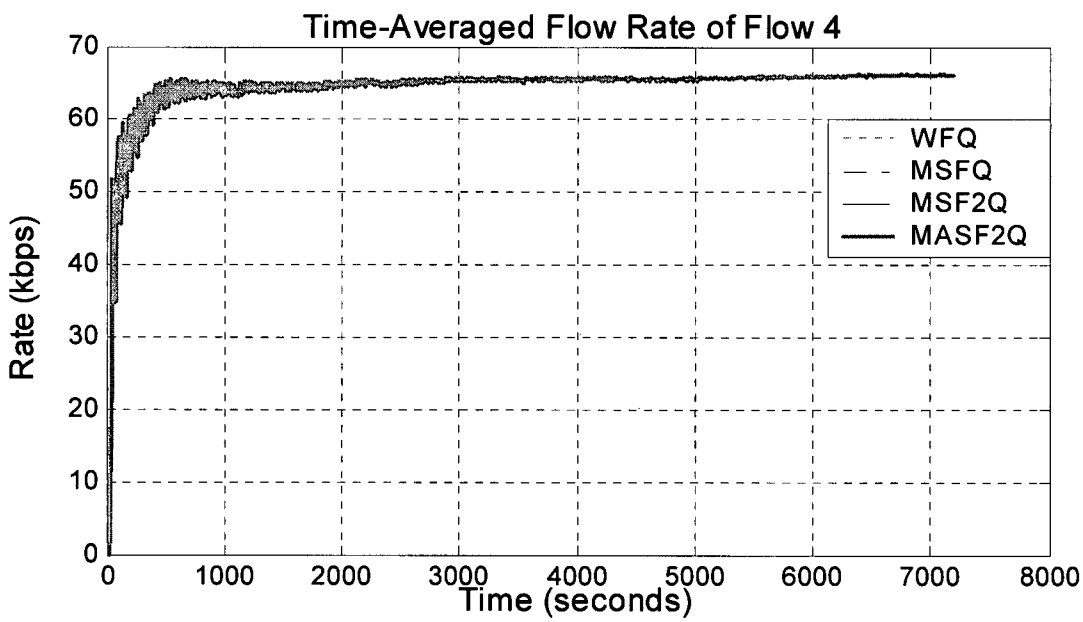
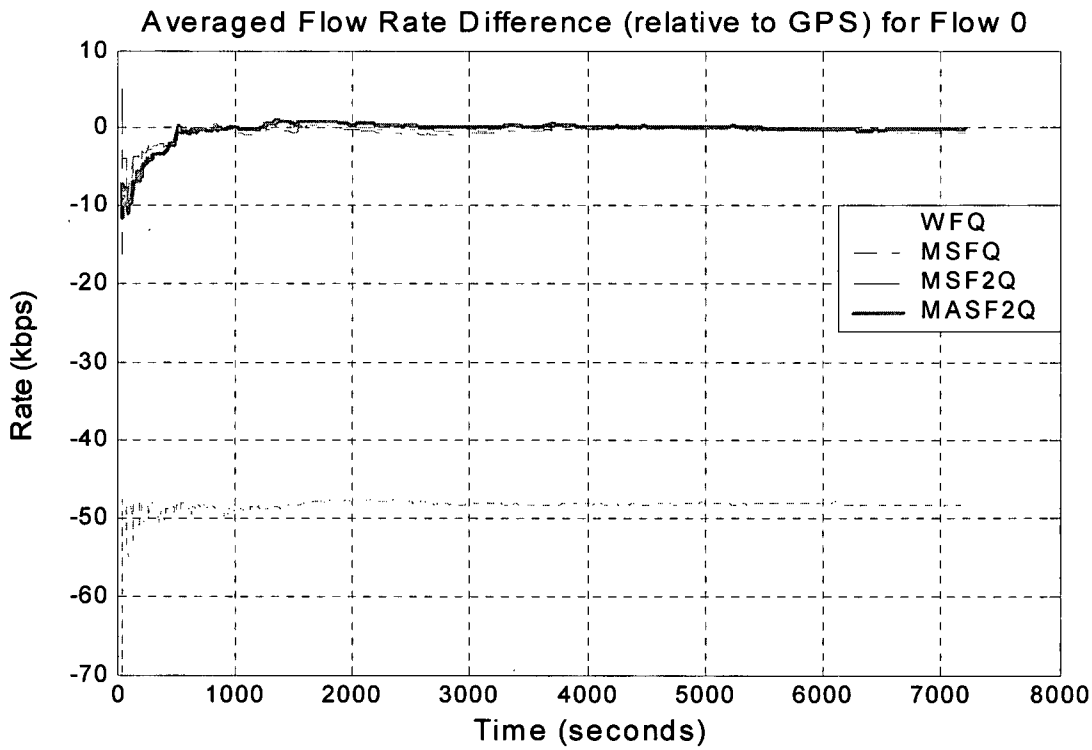


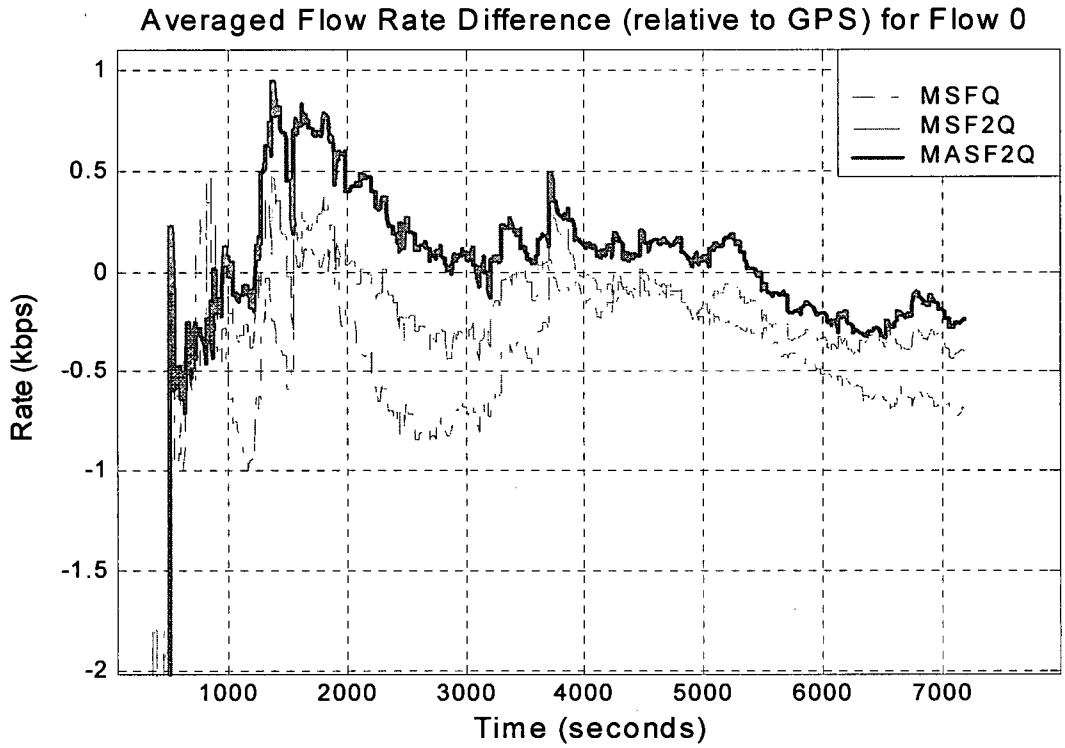
Figure 4.21: Time-averaged flow rate of flow 4

We measured again the bandwidth difference (in bps) between GPS flow rate and packetized flow rate, or $r_i(\tau) - \hat{r}_i\ (\tau)$, to give a snapshot of how fair each scheme was toward every flow compared to GPS. The closer the line plots of flow rate difference were to zero (or

time axis), the better the fairness achieved relative to GPS. The trends in flows 1 and 4 (Figures 4.23 and 4.26 below) showed that MASF^{2}Q was at least similar in fairness compared to MSF^{2}Q, and, for the most part, approached GPS more closely. These plots again suggested that a suitable packet-server pairing could allow MASF^{2}Q to achieve better fairness. However, the trend in flow 0 (Figure 4.22) showed that MASF^{2}Q was about equal in fairness compared to MSF^{2}Q. The line plots seemed to alternate (in closeness to the time axis) which suggested that flow 0 did not enjoy the same advantage as flows 1 and 4. The trends in flows 2 and 3 (Figures 4.24 and 4.25) showed that MASF^{2}Q was worse in fairness compared to MSF^{2}Q. This suggested that some flows suffered slightly in fairness while others improved. The packet-server pairing algorithm did not seem to benefit from the many bursty periods added to all flows at the same time points. This could have been caused by only one server becoming idle at any one time most of the time during an ON-period. This might have been enough to again limit the efficiency of packet-server pairing.

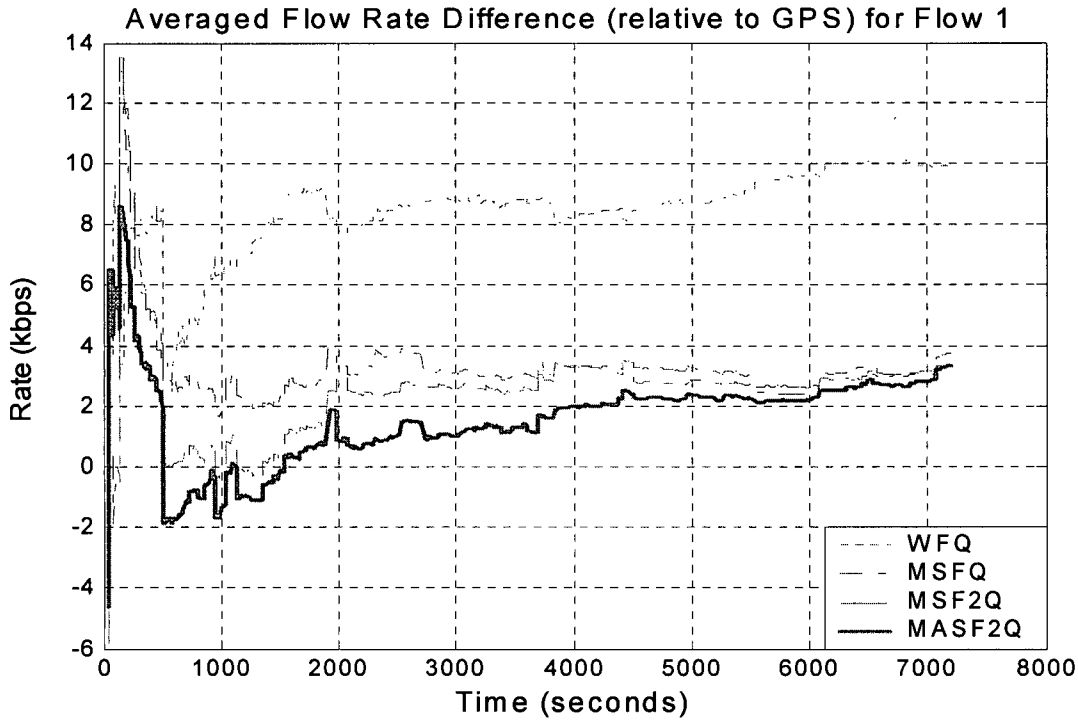


(a)

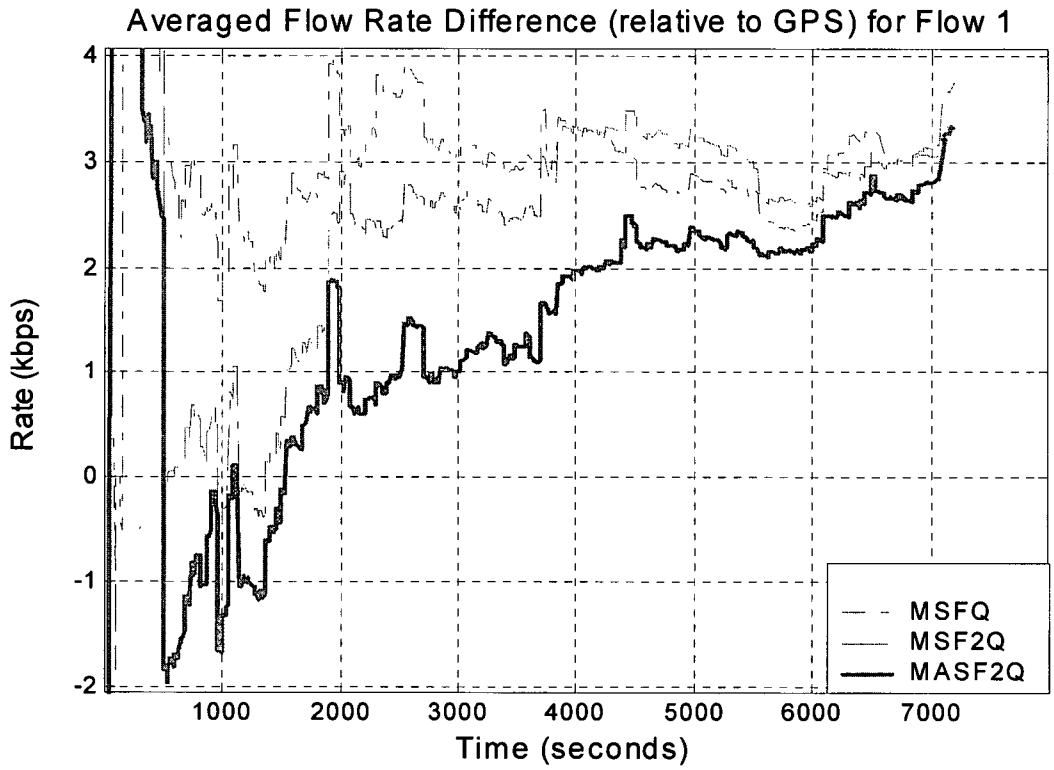


(b)

Figure 4.22: Flow rate difference of flow 0 relative to GPS system: (a) overall view showing WFQ, and (b) detailed view.

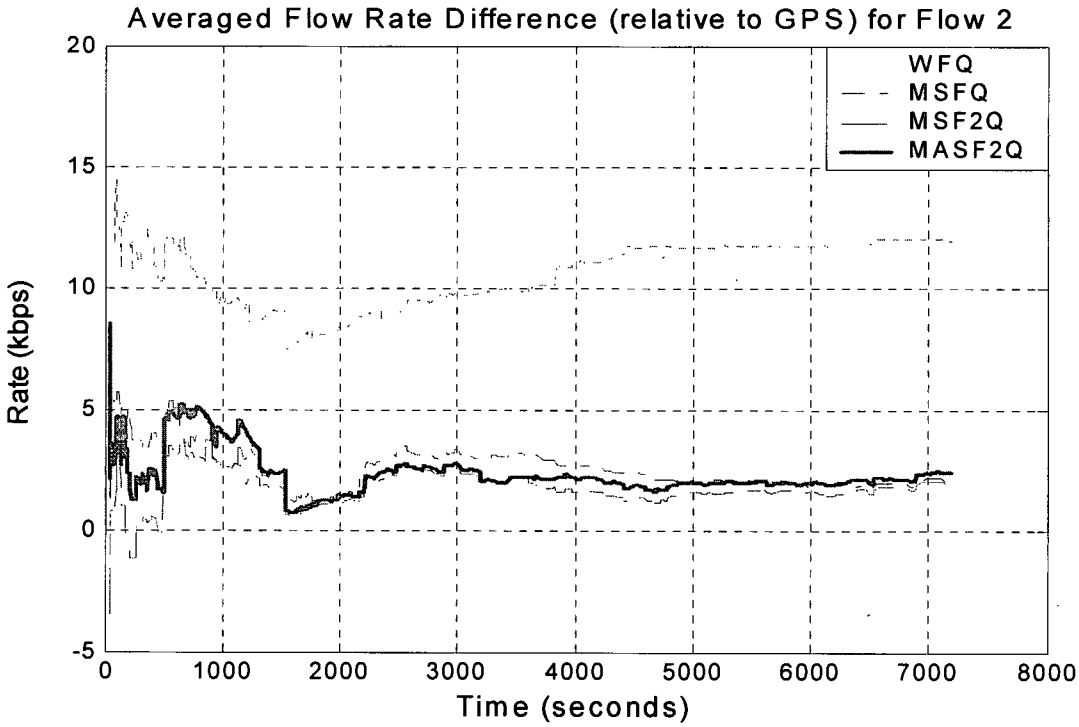


(a)

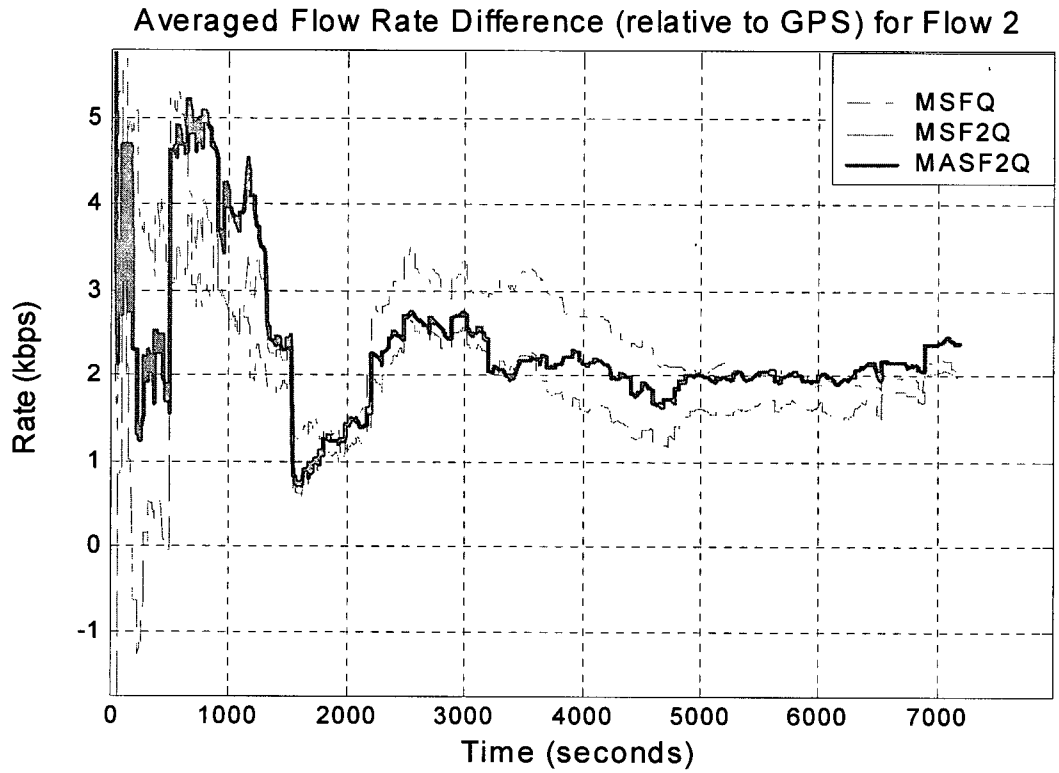


(b)

Figure 4.23: Flow rate difference of flow 1 relative to GPS system: (a) overall view showing WFQ, and (b) detailed view.

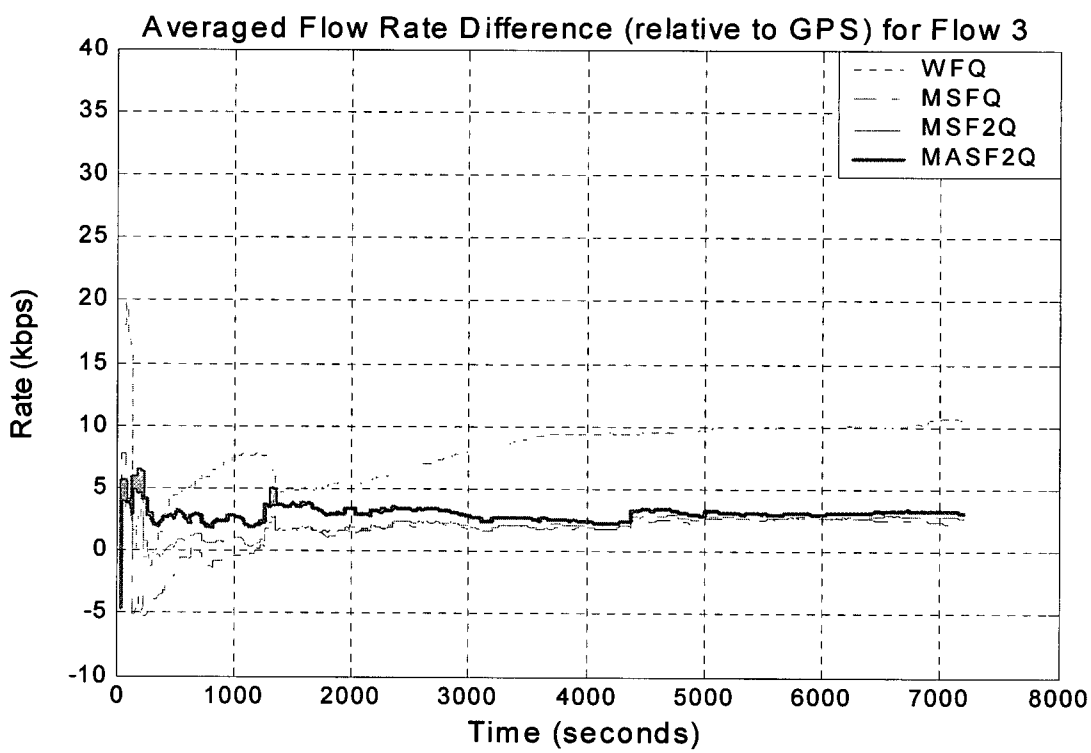


(a)

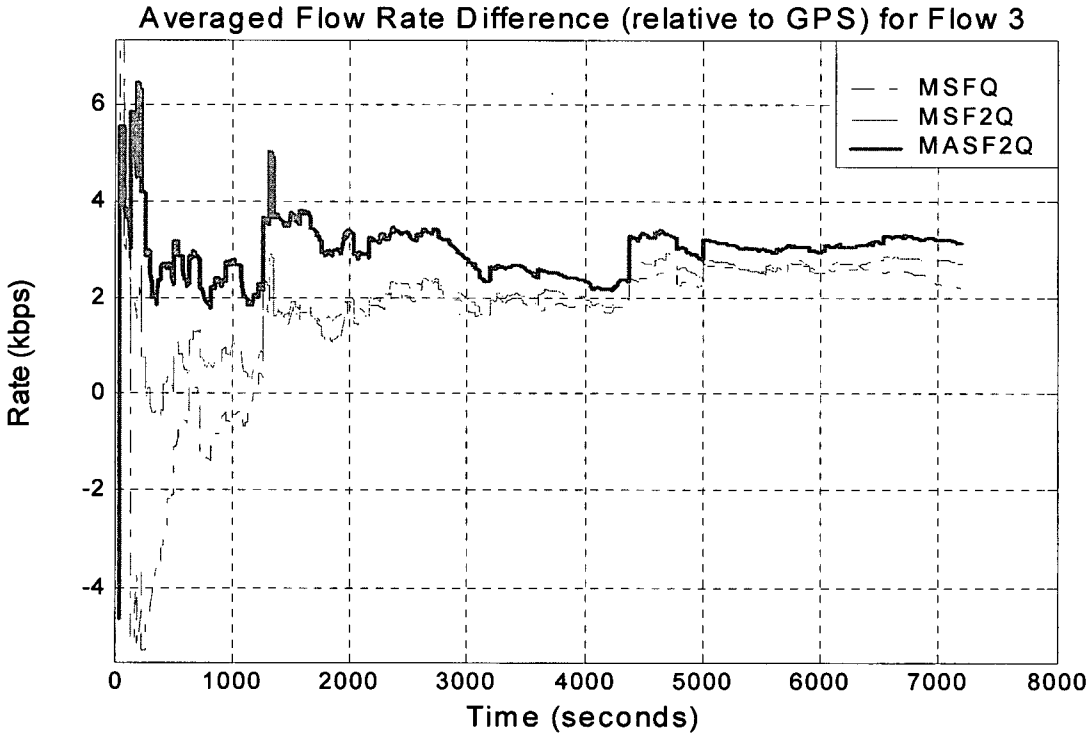


(b)

Figure 4.24: Flow rate difference of flow 2 relative to GPS system: (a) overall view showing WFQ, and (b) detailed view.

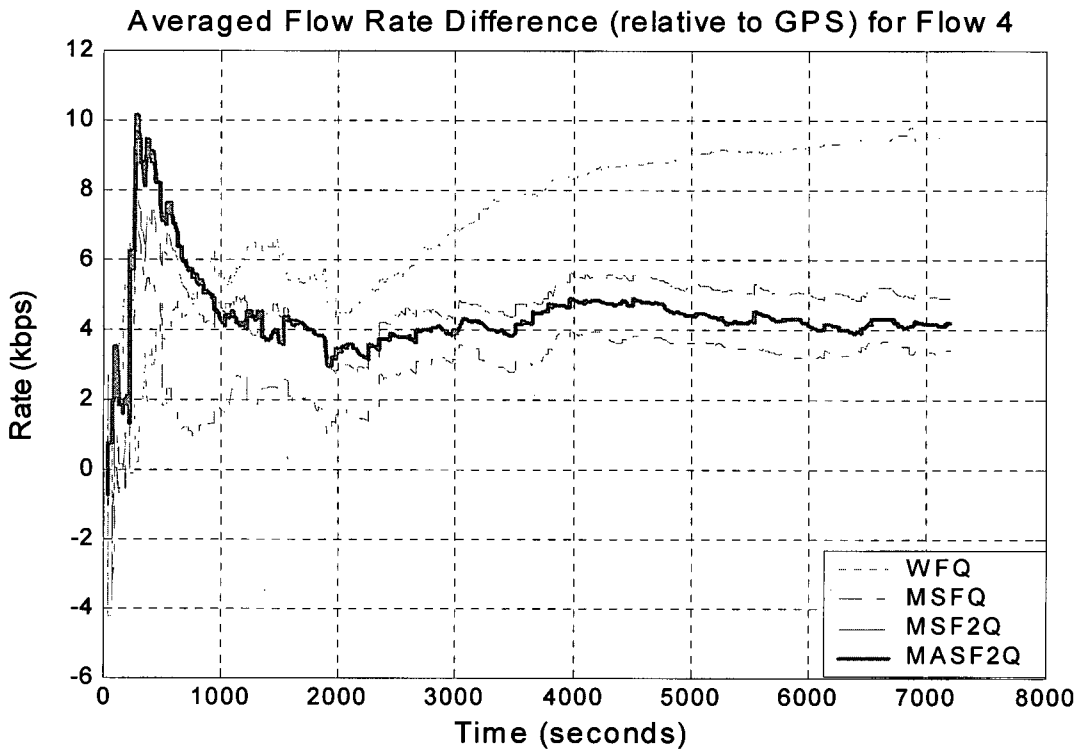


(a)

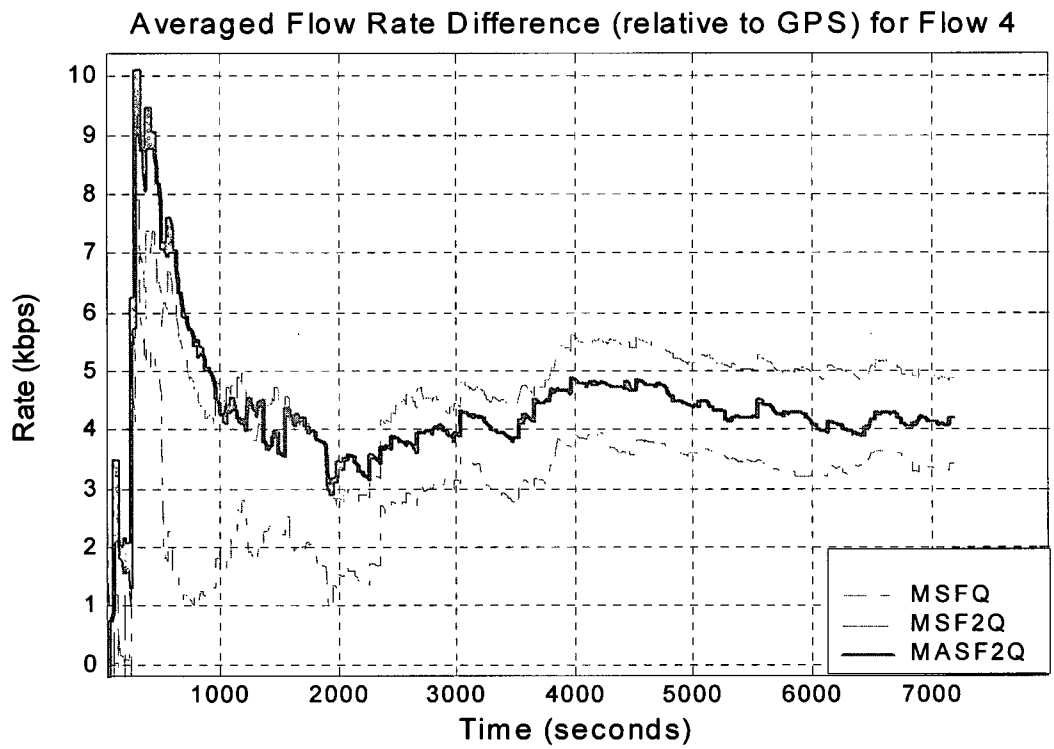


(b)

Figure 4.25: Flow rate difference of flow 3 relative to GPS system: (a) overall view showing WFQ, and (b) detailed view.



(a)

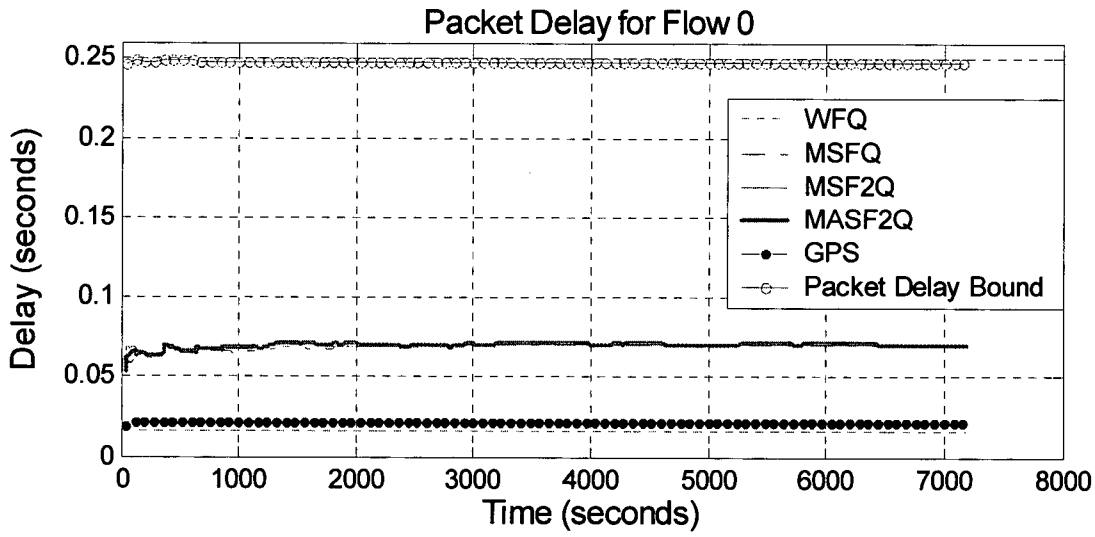


(b)

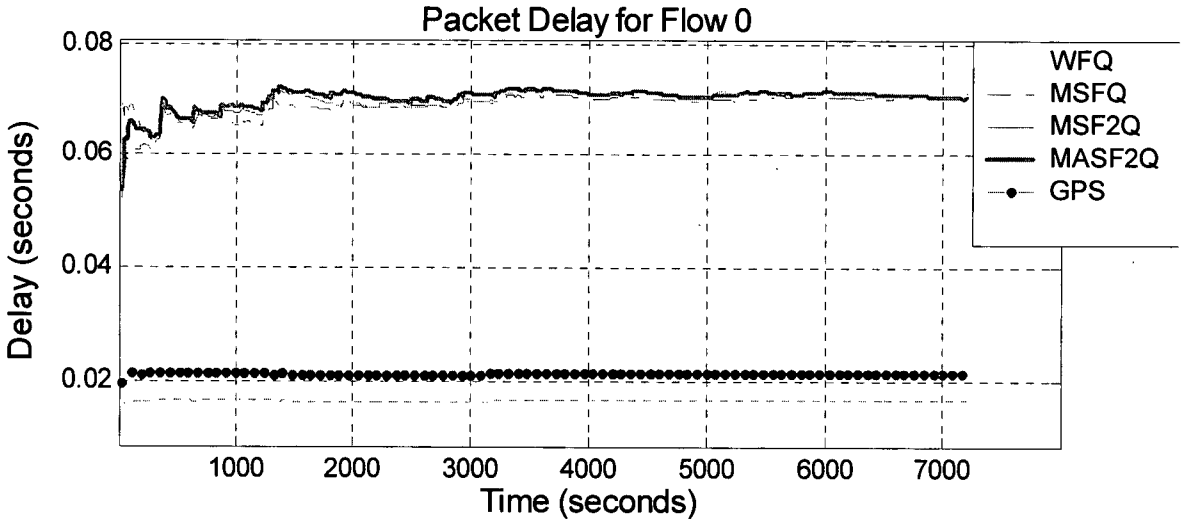
Figure 4.26: Flow rate difference of flow 4 relative to GPS system: (a) overall view showing WFQ, and (b) detailed view.

For bursty traffic pattern, MASF^{2}Q maintained very similar packet delay performance for all flows compared to the original MSF^{2}Q scheme (Figures 4.27 to 4.31). MASF^{2}Q packet delay was again well within the worst-case packet delay bound as derived in Theorem 2 of Section 3.2. The packet delay bound was 0.22667 seconds larger than the GPS packet delay. The same explanation given in Simulation Scenario 1 on the relationships of average packet delays between MSFQ, MSF^{2}Q, and MASF^{2}Q holds. MSFQ tends to have slightly better average packet delays because it was a work-conserving scheme that allowed packets to start and finish service sooner. The eligibility constraints of MSF^{2}Q and the rules of MASF^{2}Q sacrificed this work-conservation and prevented packets of backlogged flows to always be scheduled starting from the fastest idle servers available. This resulted in increased average packet delay. Compared to those of single-server schemes, all the multi-server schemes here had greater overall delays. This was caused by packets in WFQ and GPS finishing much earlier because the packets tend to occupy up to full bandwidth of the system. This was not true for the multi-server schemes because total bandwidth was fragmented into a number of

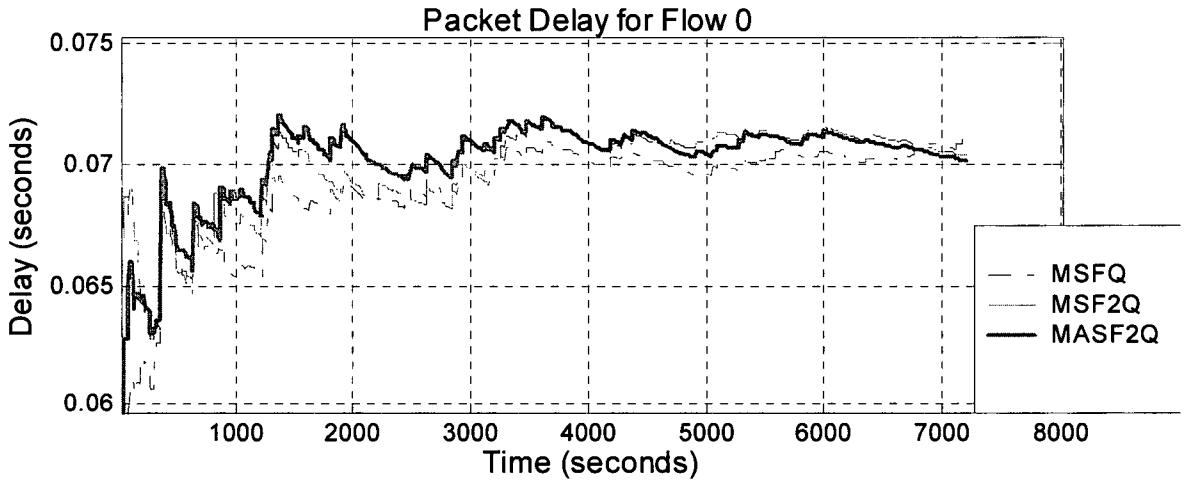
servers with fixed rates. Unless there were more packets backlogged than servers available, the full bandwidth would not be utilized.



(a)

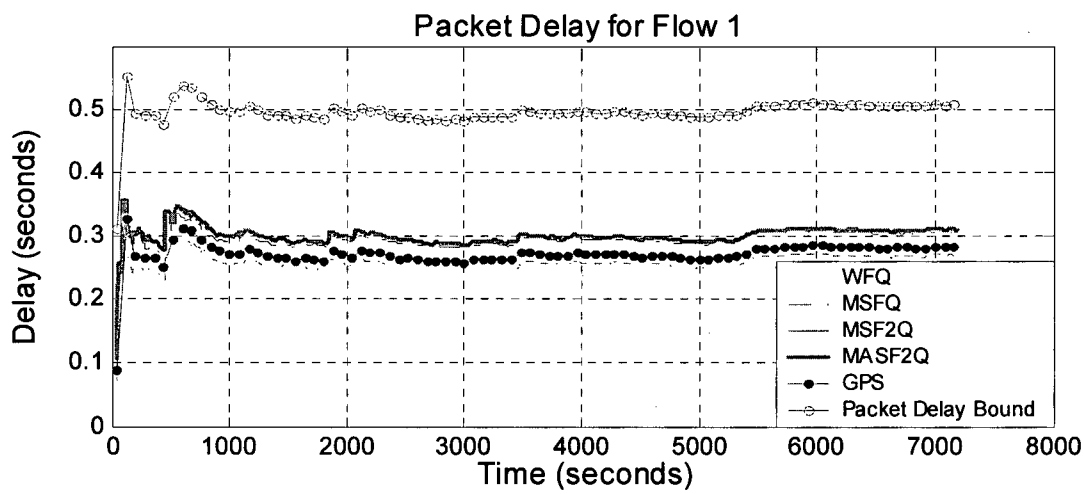


(b)

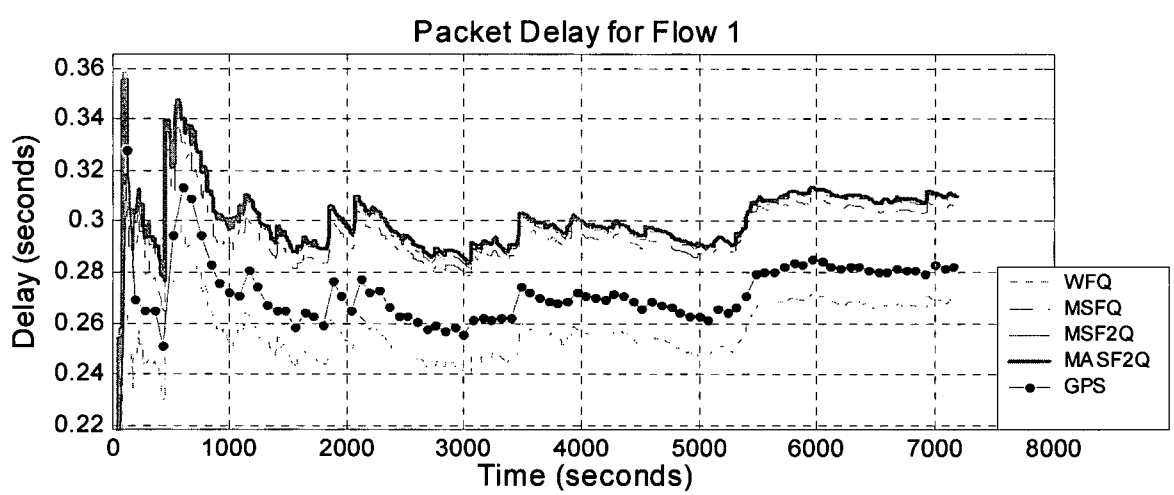


(c)

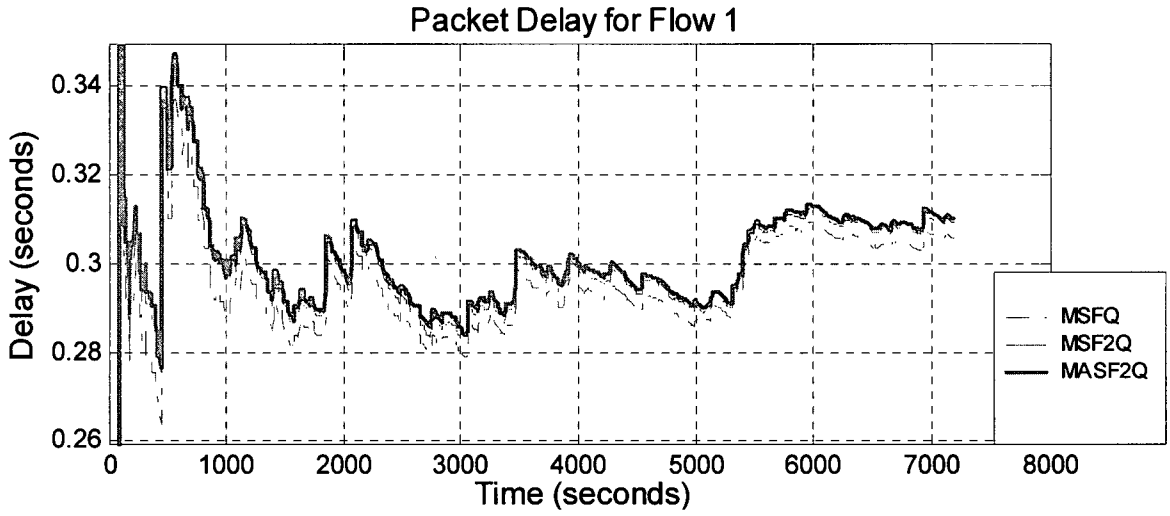
Figure 4.27: Average packet delay for flow 0: (a) overall view showing Delay Bound; (b), (c) detailed view



(a)

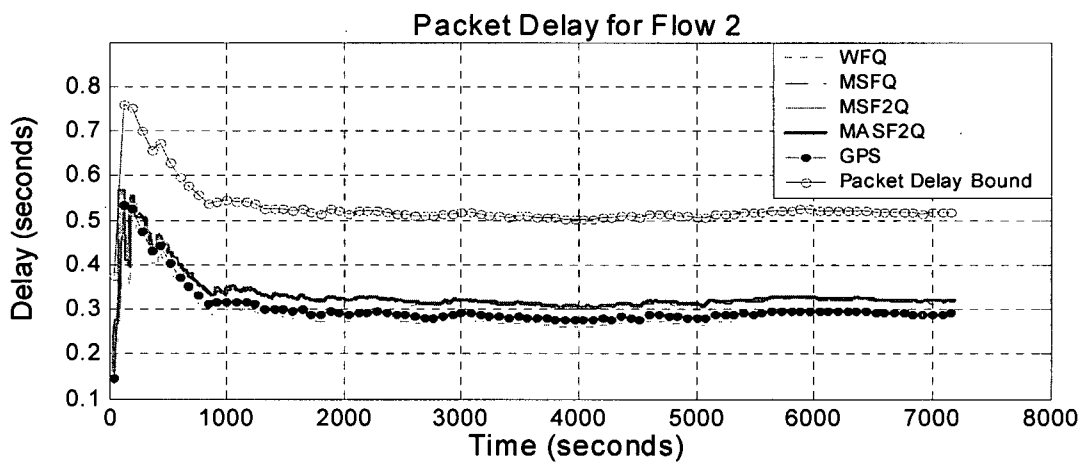


(b)

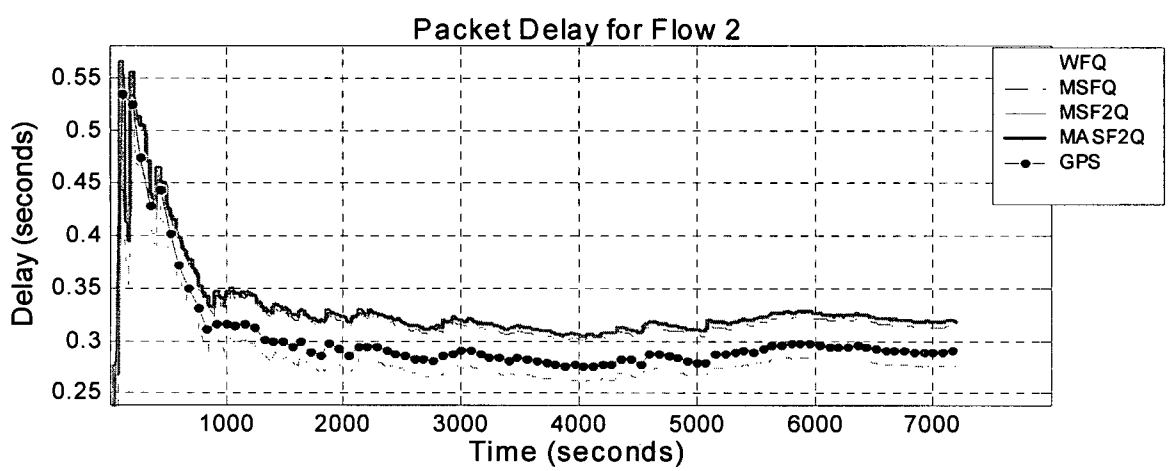


(c)

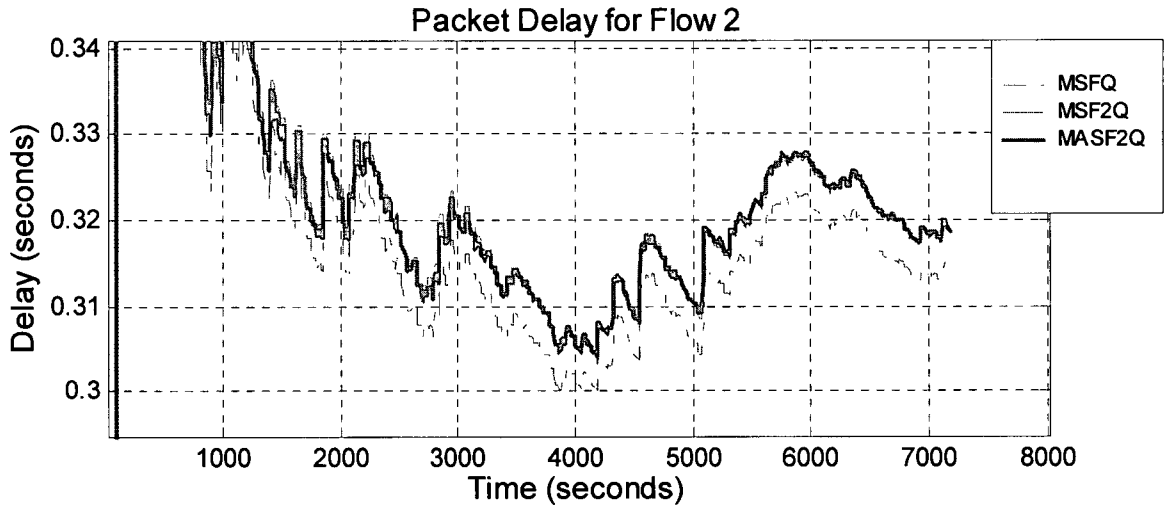
Figure 4.28: Average packet delay for flow 1: (a) overall view showing Delay Bound; (b), (c) detailed view



(a)

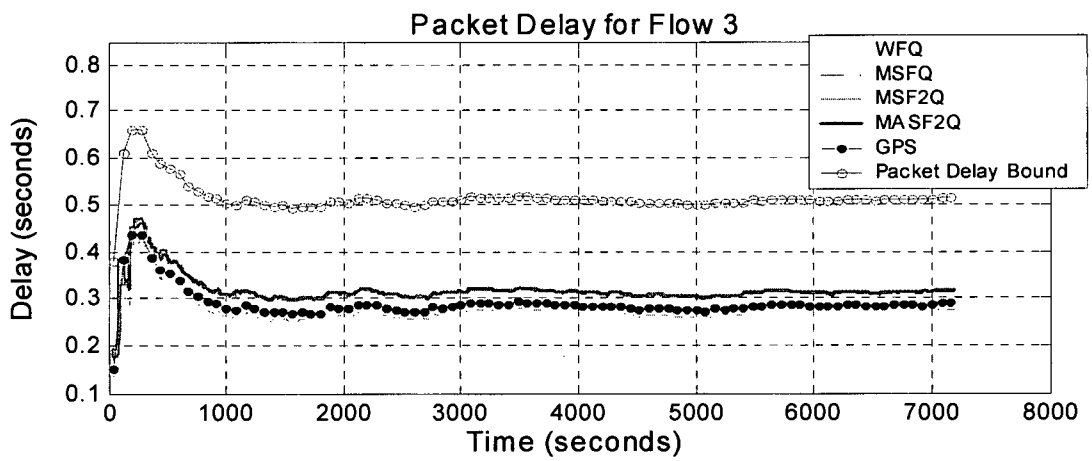


(b)

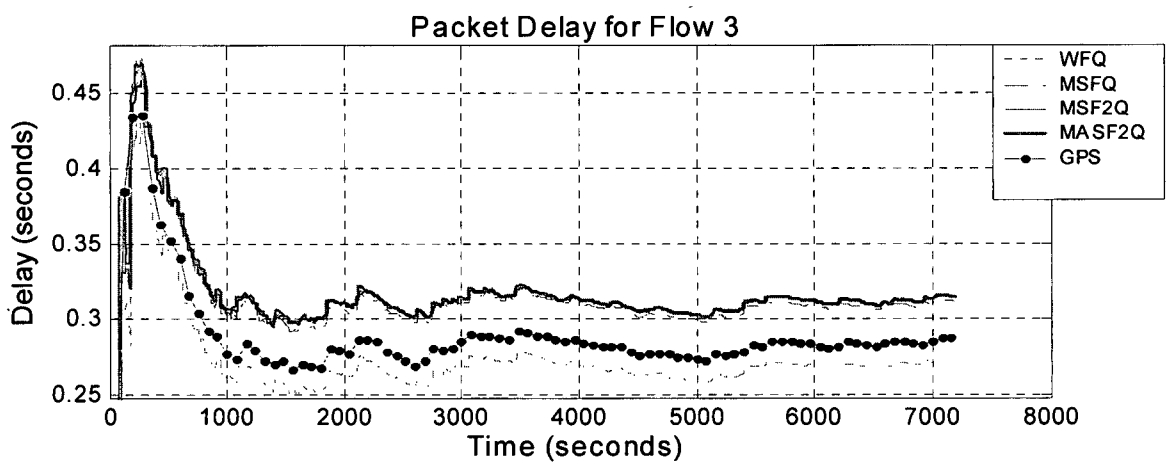


(c)

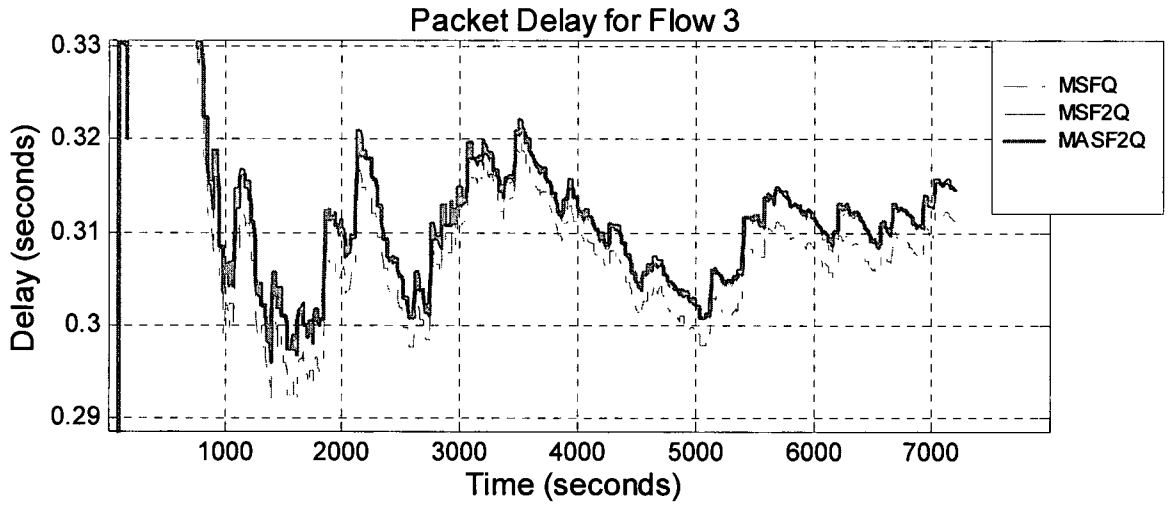
Figure 4.29: Average packet delay for flow 2: (a) overall view showing Delay Bound; (b), (c) detailed view



(a)

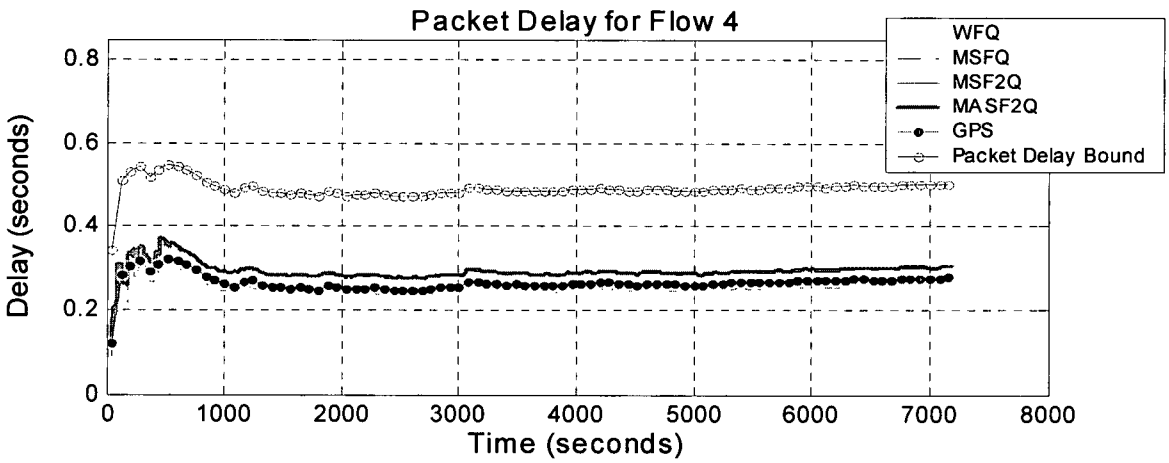


(b)

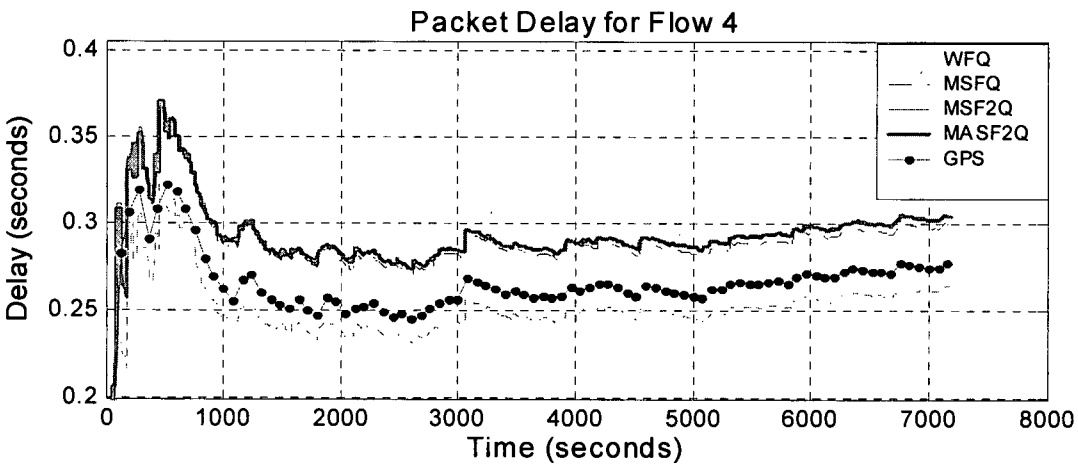


(c)

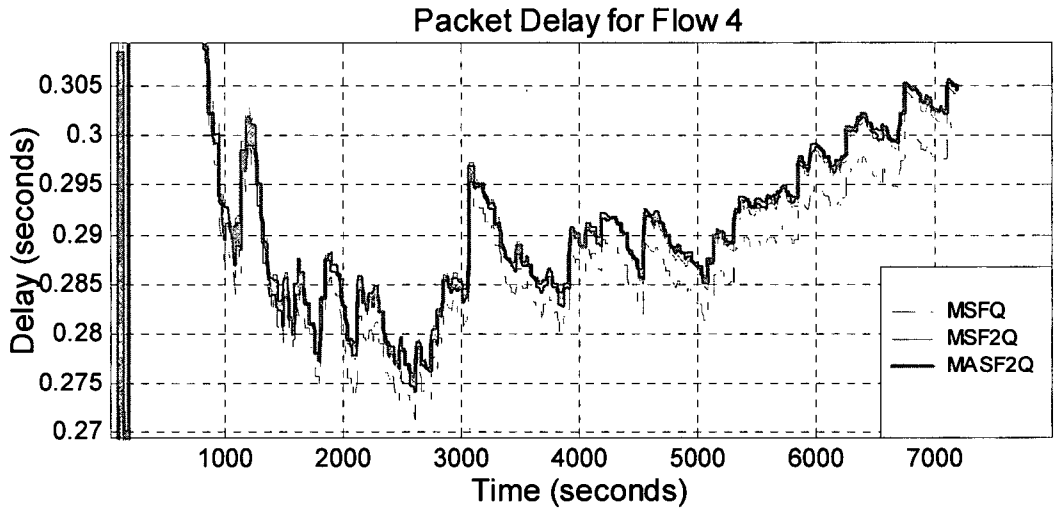
Figure 4.30: Average packet delay for flow 3: (a) overall view showing Delay Bound; (b), (c) detailed view



(a)



(b)



(c)

Figure 4.31: Average packet delay for flow 4: (a) overall view showing Delay Bound; (b), (c) detailed view

4.3 Summary

In this chapter, we presented 2 simulation scenarios to provide performance comparison among the various service disciplines when applied to MAS system. MASF^{2}Q and other packetized schemes were implemented. The first scenario had a continuous traffic pattern, while the second a bursty one. As was shown in figures, MASF^{2}Q had very little effect on packet delays. Simulation results suggested that the packet-server pairing rules of MASF^{2}Q in this thesis improved the fairness of only some of the flows, but perhaps at the expense of other flows. This was true on both simulation scenarios regardless of the traffic pattern. The reason might have been because only one server became idle at any one time most of the time when packets were backlogged and ready to be served. This limited the efficiency of the packet-server pairing that needed two or more idle servers to choose from. Although the theoretical worst-case bounds on packet delay and per-flow service discrepancy were correct, however, the expected outcome in the form of improvement in service fairness for *all* flows was not shown by the simulation results. The simulation results presented were unable to verify that MASF^{2}Q achieved its desired goal in terms of better fairness compared to MSF^{2}Q.

CHAPTER 5 CONCLUSIONS AND FUTURE WORK

We revisited the idea of fair queuing by starting with the ideal reference GPS scheduling discipline that provides perfect isolation among flows. A lot of research on scheduling has involved a single packetized server that approximates GPS in terms of delay, service discrepancy and throughput fairness. The resulting scheduling disciplines, such as PGPS, WF^2Q, etc. do not provide the same performance bounds when applied directly to multiple server systems. Unlike single-server systems, multiple server systems allow packet re-ordering among packets of the same flow that can lead to large delay bound. Due to bandwidth division, multiple server systems allow many flows to be served at the same time and may lead to poor throughput fairness.

The MSF^2Q service discipline takes advantage of even division of bandwidth to achieve better fairness. Its scheduling eligibility constraints allow flows that fall behind in service (compared to GPS) to get more bandwidth to catch up. At the same time, flows that are ahead in service are forced to wait for GPS. On average, this allows flow rates in MSF^2Q to approximate GPS more closely.

With multiple asymmetrical server systems, MSF^2Q does not have rules on how to pair up a packet with a server of suitable rate. In this thesis, we adapted the MSF^2Q for multiple asymmetrical server systems by adding rules for determining the appropriate packet-server pairings. Just like MSF^2Q taking advantage of bandwidth division, our new scheme, called $MASF^2Q$, tries to take advantage of asymmetrical server rates to improve throughput fairness. With more choice of rates, on average, $MASF^2Q$ should be able to reduce any overshoot or undershoot of packetized flow rate in its attempt to match GPS flow rate. A theoretical example was given in Section 3.4 to illustrate that, in principle, the new scheme could produce better throughput fairness by more evenly distributing the traffic of a flow throughout a busy period. This flow tracked its minimum guaranteed bandwidth while allowing other flows to utilize larger pieces of the remaining available system bandwidth. Simulation results were also provided in Chapter 4 to show comparison among $MASF^2Q$ and other service disciplines. The results suggested that the packet-server pairing rules of $MASF^2Q$ in this thesis improved the fairness of only some of the flows, perhaps at the expense of other flows. The reason might have been because only one server became idle at any one time most of the time when packets were backlogged and ready to be served. This limited the

efficiency of the packet-server pairing that needed two or more idle servers to choose from. Based on the simulation results, we were unable to conclude that the algorithm performed better than the MSF^2Q. The desired result was all the flows received better fairness.

Future work to study and improve throughput fairness includes running additional simulations where bandwidth is subdivided into smaller pieces. This should allow us to study how the level of bandwidth fragmentation affects throughput fairness. Another improvement is to devise packet-server pairing algorithms that account for cases when a single server, rather than two or more, becomes available. This new algorithm may actually be intelligent enough to wait until at least two servers become idle before pairing packets and servers. This may require additional computation to decide whether pausing scheduling until another server is idle is worth the wait. This method, if successful in improving throughput fairness, may involve more sacrifice on work-conservation. The packet-server pairing idea could be combined with any single-server service disciplines, such as Deficit Round Robin, to handle multiple server systems. Research can also be done on the possibility of integrating packet-server pairing algorithms with the flow eligibility condition of $MASF^2Q$ as one unit rather than two separate steps.

BIBLIOGRAPHY

[1] J. M. Blanquer and B. Ozden, "Fair Queuing for Aggregated Multiple Links," *Proceedings of the ACM SIGCOMM*, San Diego, California, pages 189 – 197, August 2001.

[2] A. Demers, S. Keshav, and S. Shenkar, "Analysis and Simulation of a Fair Queuing Algorithm," *Journal of Internetworking: Research And Expererience,* vol. 1, no. 1, pages 3 – 26, Dec 1990.

[3] A. K. Parekh and R. G. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks – the Single Node Case," *IEEE/ACM Transactions on Networking*, vol. 1, no. 3, pages 344 – 357, June 1993.

[4] J. C. R. Bennet and H. Zhang, "WF^2Q: Worst-case Fair Weighted Fair Queueing," *Proceedings of the IEEE INFOCOM*, San Francisco, California, pages 120 – 128, March 1996.

[5] S. J. Golestani, "A Self-Clocked Fair Queuing Scheme for Broadband Applications," *Proceedings of the IEEE INFOCOM 1994 Conference*, Toronto, Canada, pages 636 – 646, June 1994.

[6] L. Zhang, "Virtual Clock: A New Traffic Control Algorithm for Packet-Switched Networks," *ACM Transactions on Computer Systems*, vol. 9, no. 2, pages 101 – 124, May 1991.

[7] M. Shreedhar and G. Varghese, "Efficient Fair Queuing Using Deficit Round Robin," *IEEE/ACM Transactions on Networking*, vol. 4, no.3, pages 375 – 385, June 1996.

[8] HP™ System Interconnect Solutions Lab, "Using Auto-Port Aggregation to Build a Screaming Fast Network Server Connection," *Hewlett Packard Networking and Communications Technical Documentation,* September 2000.

[9] "Sun™ Trunking 1.3 Installation and User's Guide," *Sun Trunking Software Documentation.* Part No. 817-3374-10, Rev. A, July 2003

[10] "3Com's Dynamic Access," *EtherLink® Server 10/100 PCI NIC Family User Guide*, October 2000.

[11] "Intel's Adaptive Load Balancing," *Intel EtherExpress™ PRO/100+ TX PCI Model B LAN Adapter Installation Guide*, 1998.

[12] J. A. Cobb and M. Lin, "A Theory of Multi-Channel Schedulers for Quality of Service," *Journal of High Speed Networks*, vol. 12, no. 1 – 2, pages 61 – 86, September 2003.

[13] J. Nagle, "On Packet Switches with Infinite Storage," *IEEE Transactions on Communications*, vol. 35, no. 4, pages 435 – 438, April 1987.

[14] K. N. Gopinath, J. Blanquer, and B. Ozden, "An Empirical Analysis of Packet Reordering Link Aggregation Schemes and their Impact on TCP Performance," *Proceedings of the 2004 Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, pages 43 – 53, 2004.

[15] "Linux Channel Bonding," *SourceForge.net Project*

[16] OPNET Modeler ver. 9.1.A, *OPNET Technologies Inc.*