An Efficient ATM Network Switch Scheduling

Anthony T. Chronopoulos, *Senior Member, IEEE*, Caimu Tang, *Student Member, IEEE*, and Ece Yaprak, *Member, IEEE*

***Abstract*—The problem of allocating network resources to application sessions backlogged at an individual switch has a great impact on the end-to-end delay and throughput guarantees offered by the network. There exists a class of algorithms based on Weighted Fair Queueing (WFQ) for scheduling packets which are work-conserving and they guarantee fairness to the backlogged sessions. These algorithms also apply to ATM networks with packet equal to a single cell or an ATM block of (fixed size).**

Bursts are groups of varying numbers of cells. We generalize WFQ to schedule bursts. Our motivation is to derive an adaptive algorithm which generalizes the (fixed size) packet level to a varying size packet level. The new algorithm enhances the performance of the switch service for many important applications. The proposed scheme maintains the work-conserving property, and also provides throughput and fairness guarantees. The worse-case delay bound is also given. We use simulation to study the performance characteristics of our algorithm. Our results demonstrate the efficiency of the new algorithm.

***Index Terms*—ATM networks, cell burst, fair queueing, quality of service (QoS), quality measurement unit.**

I. Introduction

ATM is a high-speed connection-oriented switching multiplexing technology that allows various kinds of applications to run under a uniform infrastructure. Quality of Service (QoS) guaranteed networking is a prominent characteristic of ATM technology, which makes it possible for the multimedia and other mission-critical applications to run in this network environment. The QoS guarantees are usually in terms of bandwidth, packet delay, delay jitter, and packet loss. The (client application) source submits the traffic specifications. The network provides QoS guarantees by making suitable scheduling of the network resources. The traffic request and the QoS guarantees form a 'contract' between the source and the network. The network provides the QoS guarantees in the traffic contract as long as the source conforms with the traffic specifications. In order to support various kinds of B-ISDN services, ATM employs a layered-protocol, and it uses fixed-length packets (cells or blocks of cells) as the switching data unit.

The sequence of cells of an application (called session) are transmitted by a source and reach their destination via a path of intermediate switches. On each switch a server schedules and forward packets (from different sessions) along the path from their source to their destination in the network. Several scheduling algorithms which provide QoS guarantees have been proposed and analyzed. Examples of such scheduling algorithms can be found in [5], [7], [9], [15], [17] and references therein.

There are several algorithms proposed for packet switched networks, including Stop-and-go Queueing [8], VirtualClock Multiplexing [16] and One-level Generalized Processor Sharing (GPS) [6] and Packetized Generalized Processor Sharing (PGPS) [13]. Other results aim to improve these algorithms in terms of computational efficiency in a gigabit network model [1], [10]. For ATM networking, the basic traffic unit is cell which differs from all previous studied scenarios, and this network also has very high bandwidth. Therefore, performance speed is crucial. The advantage is two-fold to schedule in a burst level rather than a cell level, first of all, it helps to speedup the queueing computation, instead of determining scheduling on a cell level, a chunk of cells (burst) only needs one scheduling computation, for example, if the average cell chunk size is 10, it means the computation is reduced by 10 folds, secondly, burst scheduling can easily provide QoS guarantees at session level rather than at cell level. From end user point of view, cell level QoS guarantees have only indirect impact on the applications while burst level QoS guarantees can directly benefit end users, as a simple example on video playback, a end user does not care if every cell from the next frame is delivered on time for a smooth playback, it is crucial that the cells belong to the next frame are delivered on-time. The algorithm developed in this work has been focused on these aspects from the ATM networking point of view.

In this paper we study scheduling the output queue for a single node in an ATM network. We consider store and forward network switching. We are interested in scheduling bursts (of cells) instead of individual cells. Although the precise definition is given later, a burst is a group of cells of a session which have arrived at a single switch node. This is different from the definition of a burst in [15], [17]. There a burst (or block) has fixed size and its size is contained in the block header. In our definition the burst size varies, and it may change depending on the traffic conditions at a certain switch.

We consider the problem of scheduling bursts in a single node. The algorithm extend the WFQ (or PGPS) algorithm [13]. We will use the abbreviations WFQ or PGPS (GPS) to denote the same discipline in the rest of the paper. We use the virtual time function and we compute the timestamp of the arriving burst based on the burst already in service. This is similar to SCFQ for packet scheduling [7], [9]. This algorithm will be called Burst (-based) Weighted Fair Queueing (BWFQ). The algorithm is a generalization of WFQ and maintains the *work-conserving* (i.e., the server is busy as long as there are backlogged packets in the server) and the *fairness* (at burst-level) properties. We then give bounds on the deviation of the guarantees BWFQ from WFQ. These bounds are in terms of parameters of

Manuscript received June 10, 2002; revised May 23, 2003.

A. T. Chronopoulos is with the Computer Science Department, University of Texas, San Antonio, TX 78249 USA (e-mail: atc@cs.utsa.edu).

E. Yaprak is with the College of Engineering, Wayne State University, Detroit, MI 48206 USA (e-mail: yaprak@et1.eng.wayne.edu).

Digital Object Identifier 10.1109/TBC.2003.817079

the bursts. Assuming that the buffer space of a switch does not grow we can expect that BWFQ will yield superior performance over WFQ especially when the number of sessions (backlogged) is small and the application data units are large. However, in the other cases it is expected to perform no worse than (cell) WFQ, because it is a generalization of it. Our simulations confirm our theoretical expectations.

BWFQ has a few salient advantages compared to regular scheduling algorithms besides the above mentioned ones. One is the relaxed deadline in terms of end-to-end delay. The other is the computation efficiency. Since the high-level packet is segmented according to ATM cell size, one high-level packet usually will cause a cell burst in the underlying network, and sometimes the high-level packet is the unit of quality of performance measurement. For example, an FTP packet will generate a TCP packet, and the TCP packet is further encapsulated in an IP packet and is sent out. Also, many current multimedia applications use UDP as the transport layer protocol. One application level packet will need several UDP packets to be sent. These UDP packets will generate a cell burst in the underlying ATM network. Since the high-level information is totally encapsulated in these packets, from the lower level network point of view it is almost impossible to get extact burst information (except for other mechanisms that pass this information down to the network node, such as through setup signaling). However, we can assume an one-to-one correspondence between high level packets to a cell burst in the network.

In Section 2, we present the basic requirements, some model features and the terminology and notations. In Section 3, we present BWFQ for ATM scheduling and we investigate related QoS's guarantees under this algorithm. In Section 4, the fairness property is presented. In Section 5, we present simulation results. In Section 6, we draw conclusions.

II. Background and Terminology

A. Basic Requirements and Model Features

The basic requirements must ensure that traffic arriving at a switch is scheduled and forwarded respecting the QoS guarantees in the traffic contract. The most important guarantees are in terms of delay, throughput, and fairness. In order to support these guarantees the following basic requirements are needed for scheduling (session) packets which are backlogged in the switch. (a) A packet will be scheduled based on the information the switch has received. This information cannot contain any future arrival information to be received at the switch. (b) The scheduling algorithm sets an order in serving packets according to some fair discipline. This also implies that starvation (i.e., service deprivation) does not occur for any session. That is each session will be given service in a finite and reasonable amount of time.

B. Model Features

We simplify the system model as follows. An ATM switch services many sessions, some of which may come from local applications. From the scheduler point of view, it is necessary to treat these sessions and the local sessions in the same manner.

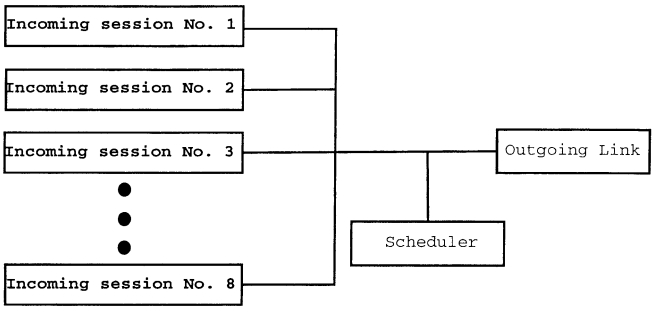

Fig. 1. The Algorithm Based Model.

Thus, regardless of their origin, the sessions are all termed incoming. Fig. 1 shows the simplified model upon which our algorithm is based.

We assume that our network traffic model has the following features.

1) All sessions are independent.
2) The store and forward mode is used.
3) There is a single scheduler for each output link in the switch.
4) Inside each session a FIFO queue is enforced.
5) The switch processing rate (i.e., CPU rate and buffer access rate) is greater than the the output link transmission rate.
6) Our model uses output queueing.

C. Terminology and Notation

We introduce terminology and notation in order to describe the BWFQ discipline and prove the guarantees that it offers. The terminology is similar to the algorithms for scheduling packets (e.g., see [13], [9], [14]).

Definition 1: (a) *System Busy Period* is the longest time interval during which the server is always busy. All backlogged cells in a system period are transmitted before the end of the system busy period. (b) *Backlogged Session Period* is the time interval during which there are cells of the session continuously backlogged. All cells backlogged during a backlogged session period are served before the end of the backlogged session period. Note that a system busy period may consist of backlogged session periods of different sessions.

Definition 2: (a) *Cell Burst* is a group of cells which belong to a single application data unit of a session. We will use *Burst* for brevity. When a burst arrives it is buffered in the input queue allocated to a session. (b) *Burst Arrival Time* (BAT) is the time when the burst arrives at the switch and gets buffered. A burst arrives at the switch when the last cell of the burst arrives.

Definition 3: (a) *Burst Departure Time* (BDT) is the time when the burst is placed in the output queue. (b) *Burst Waiting Time* (BWT) is the BDT minus the BAT.

We will use following notation in explaining the algorithm.

- (i, j): The j-th burst in session i.
- $c(i, j)$: The burst size of (i, j) equals the number of cells of a burst.
- $a(i, j)$: The arrival time of (i, j), that is, the arrival time of the last bit of the last cell in the burst.
- $f(i, j)$ or f_{GPS}: the finish time of (i, j) under GPS.

TABLE I
THE BURST ARRIVAL TIME

BN \ SN	1	2	3	4	5	6	7	8
1	0	3	0	2	10	20	30	40
2	10	20	10	8	30	30	60	50
3	15	30	18	12	40	50	90	60

TABLE II
THE BURST SIZE

BN \ SN	1	2	3	4	5	6	7	8
1	3	10	5	3	10	5	20	10
2	2	8	5	2	8	20	20	10
3	3	8	5	2	8	20	20	10

TABLE III
THE BURST DEPARTURE TIME

BN \ SN	1	2	3	4	5	6	7	8
1	3	35	11	6	45	55	101	65
2	13	73	25	15	81	147	177	119
3	18	109	50	20	127	197	217	157

TABLE IV
THE BURST WAITING TIME

BN \ SN	1	2	3	4	5	6	7	8
1	0	22	6	1	25	30	51	15
2	1	45	10	5	43	97	97	59
3	0	71	27	6	79	127	107	87

- $\hat{f}(i,j)$ or f_{BWFQ}: the finish time of (i,j) under BWFQ.
- $F(i,j)$: the virtual finish time of (i,j) under BWFQ.
- $S(i,j)$: the virtual start time of (i,j) under BWFQ.
- $\mathrm{NS}(t_1,t_2)$: the number of served cells during time interval (t_1,t_2) in the GPS server.
- $\widehat{\mathrm{NS}}(t_1,t_2)$: the number of served cells during time interval (t_1,t_2) in the BWFQ server.
- τ_0: one cell slot, a constant that depends on the link capacity and processor of the switch.

Note that the BDT of (i,j) under GPS is $f(i,j)$, and the BDT of (i,j) under BWFQ is $\hat{f}(i,j)$.

We provide an example to illustrate the notation and terminology.

Example: We use PGPS to schedule bursts instead of cells. The input parameters are given in Tables I and II, and the output is shown in Tables III and IV. Let SN stand for Session Number and let BN stand for Burst Number.

In this example, all the sessions have the same bandwidth weight. Notice that the waiting time for Session 1 and Session 4 is relatively shorter than for the others; these two sessions have a small burst size and a large interarrival interval. The session s' interarrival time distribution is an important factor not only for allocating bandwidth weight but also for shorter scheduling delay.

Here, BDT is more meaningful for the application in terms of delay and delay jitter (in video applications). Usually, a system busy period starts at the beginning of one backlogged session period. For example, if the source is an MPEG-2 codec, and there is an I-frame generated and sent out to all the intermediate nodes.

From this example we can see that the longer the BDT (at an intermediate node) is, the longer the end-to-end delay and the larger the jitter the burst will be for the application data units. Thus it makes more sense to focus on reducing the delay/jitter at burst-level instead of cell-level. In order to control the delay and delay jitter, we need to control the BDT. Several factors affect BDT: (i) the backlogged sessions at the beginning of the scheduling cycle; (ii) the burst size and the number of backlogged cells in the session queue; and (iii) the bandwidth weight available to the session if FQ is used.

III. BURST-BASED WEIGHTED FAIR QUEUEING

In this section, we present the BWFQ algorithm. We then prove a discrepancy bound for the GPS server, and we also prove delay and fairness guarantees. The simulation results (in Section 5) further demonstrate the performance advantages of BWFQ over the nonburst version.

We consider the virtual time function used in WFQ [13]. Let $V(t)$ be the virtual time function, which is defined to be zero when the server is idle. Let ϕ_i be the bandwidth weight (i.e., service rate) for the backlogged session i. Then the rate change (for session i) is controlled by $(\phi_i)/(\sum_{j\in B_t}\phi_j)$, where B_t is the set of all backlogged sessions at time t. We will apply the virtual time function to a burst instead of a single cell. Let i and k be the session index and burst number, respectively and let $c(i,k)$ be the burst size.

1) The BWFQ Algorithm: Starting system busy period (at physical time t_{start}):

$$S(i,0)=F(i,0)=0 \text{ for arbitrary session index i,}$$

and

$$V(t_{\mathrm{start}})=0.$$

Burst arrival:

1) Burst-start: $c(i,k) = 0, S(i,k) = \max\{F(i,k-1), V(a(i,k))\}$.
2) For each new cell arrival the burst size is incremented: $c(i,k)=c(i,k)+1$.
3) End of a burst: $F(i,k)=S(i,k)+c(i,k)(\tau_0/\phi_i)$, where τ_0 is the cell slot.

Burst departure:

For a given session i, the scheduler serves the bursts according to the ascending order of the $F(i,k)$ backlogged in the server so far.

Fig. 2 gives an illustration based on 3 backlogged sessions with various sizes of bursts denoted by rectangle with different length. The dotted line denotes the computation of the virtual time of a burst upon arrival and the solid line denotes the computation of the virtual finishing time. The scheduler will use virtual finishing times to schedule bursts.

We next give several results about the discrepancy bound between the BWFQ algorithm and the GPS algorithm.

By using the following lemma, we only need to consider the behavior of one system busy period in order to study the scheduling of the system.

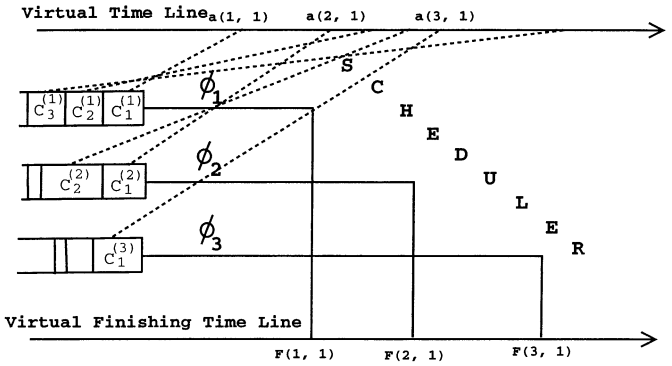

Fig. 2. Illustration of BWFQ with three backlogged sessions.

Lemma 1: If the scheduling algorithm is work-conserving, then scheduling orders of different system busy periods are unrelated.

Proof: We use s_α and e_α to denote the start-time and ending time of the system busy period α. Let j be a session index and α be a system busy period and let N be the total number of sessions in the server.

Let $A_{\alpha j}$ be all cells of the session j within period α ($A_{\alpha j} = 0$, if the session is idle in the period α or the period length is zero).

Then, we have

$$\frac{e_\alpha - s_\alpha}{\tau_0} = \sum_{j=1}^{N} A_{\alpha j}.$$

Thus, we get

$$e_\alpha = \left(\sum_{j=1}^{N} A_{\alpha j} \right) \tau_0 + s_\alpha.$$

Since $s_{\alpha+1} \geq e_\alpha$, we have, $s_{\alpha+1} \geq (\sum_{j=1} A_{\alpha j})\tau_0 + s_\alpha$.

By iteration, we get

$$s_{\alpha+1} \geq \left(\sum_{k=1}^{i} \sum_{j=1}^{N} A_{kj} \right) \tau_0.$$

This means that all the cell arrivals before the start of the period $\alpha + 1$ are served by the time $s_{\alpha+1}$. So the scheduling order is independent of the scheduling history. In other words, the scheduling order in the period $(\alpha + 1)$ is independent of the periods: $1, \ldots, \alpha$. □

Claim 1: BWFQ is a work-conserving scheme.

Explanation: Case 1: At least one session is backlogged, whose end has arrived. The burst is eligible for service. Case 2: No session is eligible for service. If there are bursts already backlogged, the burst in the head (of the priority queue) will be selected to schedule after it is truncated to determine its end. □

In order to give the main result, we need the following lemma.

Lemma 2: Assume that under BWFQ there are two bursts $(i_1, j_1), (i_2, j_2)$ at time τ, and also assume that (i_1, j_1) finishes before (i_2, j_2) when there is no arrival after time τ. Then burst (i_1, j_1) will always finish before burst (i_2, j_2), regardless of arrival patterns after time τ.

Proof: Let's assume burst (i_1, j_1) finishes at time t_1, and burst (i_2, j_2) finishes at time t_2, then $t_1 < t_2$ when there is no arrival after time τ. Let Δ be a nonempty arrival pattern after time τ, and t_1' be the new finish-time of the (i_1, j_1), t_2' be the new finish-time of (i_2, j_2). Then $t_1 \leq t_1', t_2 \leq t_2'$. Because the server is GPS and session i_1 is continuously backlogged between t_1 and t_1' and session i_2 is continuously backlogged between t_2 and t_2'.

Therefore,

$$t_2' \geq (t_1' - t_1) + \frac{\phi_1}{\phi_2}(t_2 - t_1) + t_1 = \frac{\phi_1}{\phi_2}(t_2 - t_1) + t_1',$$

where ϕ_1 and ϕ_2 are the bandwidth portions associated with the two sessions i, j according to the definition of the GPS server. Then, we have

$$t_2' \geq t_1'.$$

This means that (i_2, j_2) finishes before (i_1, j_1) under arrival pattern Δ. Since, pattern Δ is arbitrary, we proved our claim. □

The following two theorems state how good the approximation is.

Notation: $C_{\max} = \max_{(p,k)} c(p, k)$, for session bursts (p, k) backlogged at the switch.

Theorem 1: For all bursts (i, j) in BWFQ we have:

$$\hat{f}(i, j) - f(i, j) \leq \tau_0 C_{\max}. \tag{1}$$

Proof: We consider a fixed burst (i, j) and we prove the result. Since this burst is chosen arbitrarily, this suffices to prove the theorem. By Lemma 1, we only need to prove the inequality holds for one system busy period. Without loss of generality, assume the start-time of the busy period is zero. Because the BWFQ and GPS are both work-conserving disciplines, the system busy periods of these two are identical. So the start-time of system busy period under GPS is also zero.

Define a partial order: $(i, j) \leq (\tilde{i}, \tilde{j})$ iff $a(i, j) \leq a(\tilde{i}, \tilde{j})$, where $a(\cdot, \cdot)$ is the arrival time of burst $(\cdot, \cdot)$. For burst (i, j), there are two cases:

1) All bursts (p, k) which satisfy $(p, k) < (i, j)$ leave the GPS server before burst (i, j) does, then,

$$f(i, j) \geq \hat{f}(i, j) \tag{2}$$

where equation holds when only session i is continuously backlogged during $[a(i, j), f(i, j)]$.

2) There exists burst (p, k), such that $(p, k) < (i, j)$ and $f(p, k) > f(i, j)$.

Therefore the set $Q(i, j) = \{(p, k) \,|\, (p, k) < (i, j)$ and $f(p, k) > f(i, j)\} \neq \phi$.

The set Q is finite because the total number of bursts arrived before $a(i, j)$ is finite. Thus, there exists a burst $(\tilde{i}, \tilde{j}) \in Q(i, j)$, such that

$$(\tilde{i}, \tilde{j}) \geq (p, k) \tag{3}$$

for any burst $(p,k) \in Q$. Burst $(\tilde{i},\tilde{j})$ begins transmission at $\hat{f}(\tilde{i},\tilde{j})\tau_0$ under BWFQ. And by Lemma 2,

$$\min_{\tilde{i},\tilde{j})<(p,k)<(i,j)} a(p,k) > \hat{f}(\tilde{i},\tilde{j}) - c(\tilde{i},\tilde{j})\tau_0. \tag{4}$$

This means that all the bursts in $\{(p,k)\,|\,(\tilde{i},\tilde{j}) < (p,k) \leq (i,j)\}$ arrive after $\hat{f}(\tilde{i},\tilde{j}) - c(\tilde{i},\tilde{j})\tau_0$ (under GPS) and depart before burst (i,j) departs (under GPS). So we have

$$f(i,j) \geq \hat{f}(\tilde{i},\tilde{j}) - c(\tilde{i},\tilde{j})\tau_0 + \sum_{(\tilde{i},\tilde{j})<(p,k)\leq(i,j)} c(p,k)\tau_0,$$

and

$$\hat{f}(\tilde{i},\tilde{j}) + \sum_{(\tilde{i},\tilde{j})<(p,k)\leq(i,j)} c(p,k)\tau_0 = \hat{f}(i,j) \tag{5}$$

Because BWFQ is work-conserving, so by (2) and (3),

$$f(i,j) \geq \hat{f}(i,j) - c(\tilde{i},\tilde{j})\tau_0. \tag{6}$$

Therefore, $\hat{f}(i,j) - f(i,j) \leq \tau_0 \max_{(p,k)} c(p,k)$. This is the inequality (1). □

Note that, in a finish time FQ, two cases may occur. Case 1: The finish time under GPS is greater than the finish time under BWFQ. Case 2: The finish time under GPS is less than the finish time under BWFQ. In case 1, the right-hand side of inequality (1) is negative, which means that WFQ scheduled burst will finish earlier than the GPS finish time. This theorem states that the finish time under BWFQ is either earlier than GPS (case 1) or the finish time under BWFQ is later than GPS finish time but their difference is uniformly bounded (case 2).

Theorem 2: Let $C_{\max} = \max_{(p,k)} c(p,k)$. For any time τ and session i, let $\mathrm{NS}_i(\tau,t)$ and $\widehat{\mathrm{NS}}_i(\tau,t)$ be the number of cells of session i served under GPS and BWFQ, in the interval $[\tau,t]$, respectively. Then we have

$$\mathrm{NS}_i(0,\tau) - \widehat{\mathrm{NS}}_i(0,\tau) \leq C_{\max},$$

Proof: Notice that the service order of each session is FIFO under both disciplines.

Let t be the departure time of last burst, say (i,j), before τ in session i (under GPS).

For GPS:

$$\mathrm{NS}_i(0,\tau) = \sum_{k=1}^{j} c(i,k) + \int_t^{\tau} \delta(t)\,dt, \tag{7}$$

where $\delta(t)$ is the serve rate of burst $(i,j+1)$ which is a step function. The first term of (7) is the served bursts and the second term is the fraction of burst $c(i,j+1)$ already served. It is obvious that

$$\int_t^{\tau} \delta(t)\,dt < c(i,j+1).$$

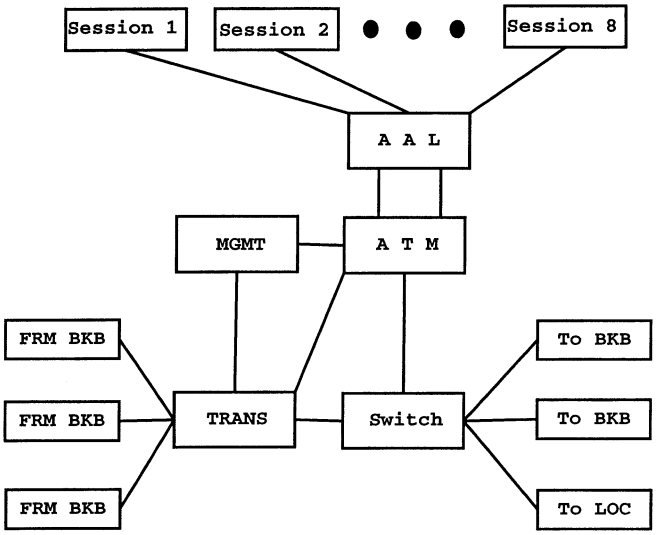

Fig. 3. Switch Model.

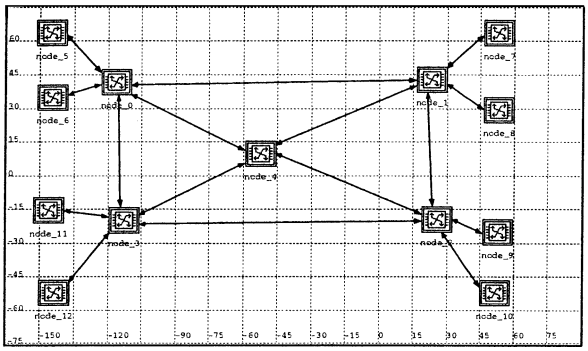

Fig. 4. Network simulation topology.

By Theorem 1, the burst (i,j) finishes under BWFQ at $\tilde{t}$, no later than $t + \tau_0 C_{\max}$, i.e.,

$$\tilde{t} \leq t + \tau_0 C_{\max}.$$

Also, $\tilde{t} < \tau$, because of the FIFO property of session i under both disciplines, we have

$$\mathrm{NS}_i(0,t) = \widehat{\mathrm{NS}}_i(\widehat{\mathrm{NS}}_i(0,\tilde{t})).$$

And a fraction of $(i,j+1)$ are finished at time τ. So there are two cases:

1) $t + C_{\max}\tau_0 > \tau > \tilde{t}$. Since $\tau > \tilde{t}$, we have

$$\widehat{\mathrm{NS}}_i(0,\tau) \geq \sum_{k=1}^{j} c(i,k).$$

2) $\tau \geq t + C_{\max}\tau_0$. Since function $\widehat{\mathrm{NS}}_i(0,t)$ is nondecreasing, we have

$$\widehat{\mathrm{NS}}_i(0,\tau) \geq \widehat{\mathrm{NS}}_i(0,t + C_{\max}\tau_0) \geq \sum_{k=1}^{j} c(i,k).$$

TABLE V
THE SOURCE TRAFFIC PARAMETERS

Session No.	1	2	3	4	5	6	7	8
Bandwidth Portion	0.1	0.1	0.1	0.1	0.2	0.5	0.5	1.0
Pk Size Args	8988	1581	1581	8988	1240	1240	2400	2400
Source Interarrival	200	200	200	200	100	100	100	100
Source Start Time	100.0	0.0	200.0	0.0	100.0	100.0	20.0	0.0
Source End Time	2000	1800	1700	1500	2000	2000	1800	1700

Therefore,

$$\widehat{\mathrm{NS}}_i(0,\tau) \geq \sum_{k=1}^{j} c(i,k). \tag{8}$$

By combining (7) and (8), we prove the theorem. □

From the previous theorem, we get the following throughput guarantee of BWFQ.

Corollary 1: For the BWFQ server, session i can have a throughput guarantee of

$$\widehat{\mathrm{NS}}_i(0,t) \geq \phi_i \frac{t}{\tau_0} - C_{\max},$$

where the ϕ_i is the bandwidth weight in a bandwidth allocation scheme.

Proof: Notice that $\mathrm{NS}_i(0,t) \geq \phi_i(t/\tau_0)$ is a minimum service guarantee by the definition of GPS [13]. The proof follows by applying Theorem 2. □

Note that the throughput guarantee is the most important requirement for multimedia applications. Without this guarantee, for example, the smooth playback of the video/audio clip is hard to obtain.

IV. FAIRNESS PROPERTY

In GPS, a burst obtains its service immediately after arrival, and the service rate depends on the currently backlogged sessions and on its own available bandwidth weight ϕ_i. This queueing scheme never overuses or underuses its service. Therefore, GPS assures ideally fair queueing. As an approximation of GPS, BWFQ can not guarantee the same level of fairness, because it never takes into account the future arrival of bursts. Nevertheless, BWFQ can still provide guaranteed fairness.

Definition 1: Let $\mathrm{NS}_i(t_1,t_2)$ be the number of served cells during interval $[t_1,t_2]$ when session i is continuously backlogged under a scheduling algorithm. The fairness index of the server is defined as:

$$\alpha = \max \left| \frac{\mathrm{NS}_i(t_1,t_2)}{\phi_i(t_2-t_1)} - \frac{\mathrm{NS}_j(t_1,t_2)}{\phi_j(t_2-t_1)} \right|$$

where i,j are sessions in the server.

Note that for a very specific case, if there is no overlap on backlogged periods, it is meaningless to consider fairness. In this case, only one session is active during a system busy period.

This definition is algorithm based, not session based. Therefore, the fairness guarantee is also algorithm based. The following proposition shows that this definition is well defined.

Proposition: For the GPS server, $\alpha = 0$.

Proof: For any two sessions i,j, let $[t_1,t_2]$ be the common interval where both are backlogged. By the definition of a GPS server, for any backlogged session i, the inequality

$$\frac{\mathrm{NS}_i(\tau,t)}{\mathrm{NS}_k(\tau,t)} \geq \frac{\phi_i}{\phi_k}$$

holds for $k = 1,2,\ldots,N$. Since both sessions i and j are backlogged in $[t_1,t_2]$, we have $(\mathrm{NS}_i(t_1,t_2))/\phi_i = (\mathrm{NS}_j(t_1,t_2))/\phi_j$. This means: $|(\mathrm{NS}_i(t_1,t_2)/\phi_i) - (\mathrm{NS}_j(t_1,t_2)/\phi_j)| = 0$. So, $\alpha = 0$. □

The following theorem gives the fairness guarantee under BWFQ.

Theorem 3: For the BWFQ server the following fairness quarantee holds:

$$\alpha \leq \frac{1}{\tau_0} \frac{1}{\max_{i\in N}\{\phi_i\}},$$

where N is the set of sessions in the server, and ϕ_i is the rate available to the session i.

Proof: By the definition of the fairness index, for two arbitrary sessions i,j which are continuously backlogged in interval $[t_1,t_2]$, there are four cases:

i) only session i receives service in $[t_1,t_2]$,
ii) only session j receives service in $[t_1,t_2]$,
iii) neither session i nor session j receives service in $[t_1,t_2]$, and
iv) both sessions i,j receive service in interleaved order in interval $[t_1,t_2]$.

In case 1,

$$\left| \frac{\mathrm{NS}_i(t_1,t_2)}{\phi_i(t_2-t_1)} - \frac{\mathrm{NS}_j(t_1,t_2)}{\phi_j(t_2-t_1)} \right| = \left| \frac{\mathrm{NS}_i(t_1,t_2)}{\phi_i(t_2-t_1)} \right| = \frac{1}{\tau_0\phi_i}.$$

In case 2,

$$\left| \frac{\mathrm{NS}_i(t_1,t_2)}{\phi_i(t_2-t_1)} - \frac{\mathrm{NS}_j(t_1,t_2)}{\phi_j(t_2-t_1)} \right| = \left| \frac{\mathrm{NS}_j(t_1,t_2)}{\phi_j(t_2-t_1)} \right| = \frac{1}{\tau_0\phi_j}.$$

In case 3,

$$\left| \frac{\mathrm{NS}_i(t_1,t_2)}{\phi_i(t_1-t_2)} - \frac{\mathrm{NS}_j(t_1,t_2)}{\phi_j(t_2-t_1)} \right| = 0.$$

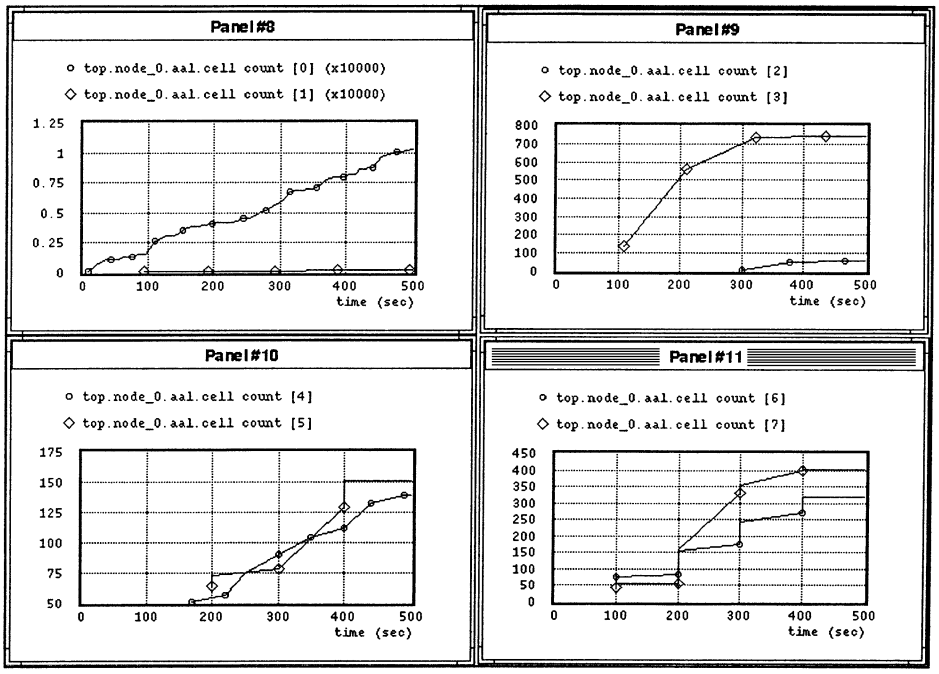

Fig. 5. Traffic arrival pattern of cell PGPS.

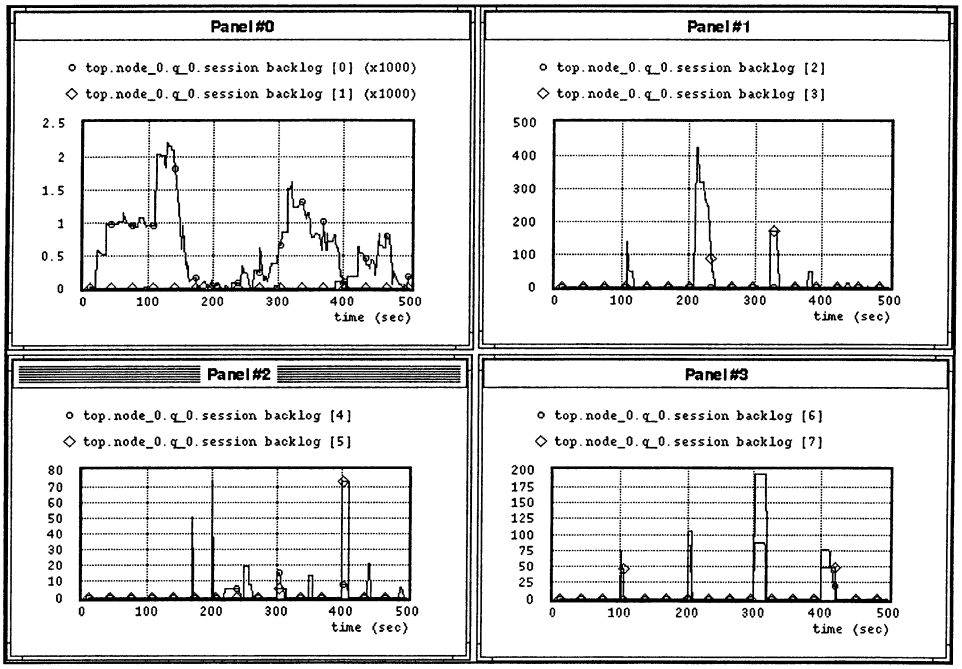

Fig. 6. Session backlog of BWFQ.

In case 4,

$$\left|\frac{\mathrm{NS}_i(t_1,t_2)}{\phi_i(t_2-t_1)}-\frac{\mathrm{NS}_j(t_1,t_2)}{\phi_j(t_2-t_1)}\right|\le\frac{1}{\tau_0}\min\left(\frac{1}{\phi_i},\frac{1}{\phi_j}\right),$$

because, for $k=i$ or $k=j$,

$$\left|\frac{\mathrm{NS}_k(t_1,t_2)}{\phi_k(t_2-t_1)}\right|\le\frac{1}{\phi_k}\frac{1}{\tau_0}.$$

Therefore, we have proved our claim. □

With Theorem 3, we know that BWFQ server can guarantee that best effort sessions do not suffer from service starvation under any condition which is one of salient features of BWFQ. This is also shown by the simulation study that follows.

V. Simulation Study

In this section, we simulate the algorithm and test the related parameters. We first model an ATM switch and implement BWFQ and cell PGPS servers. We select a group of typical sessions with different properties, e.g., best effort, real-time

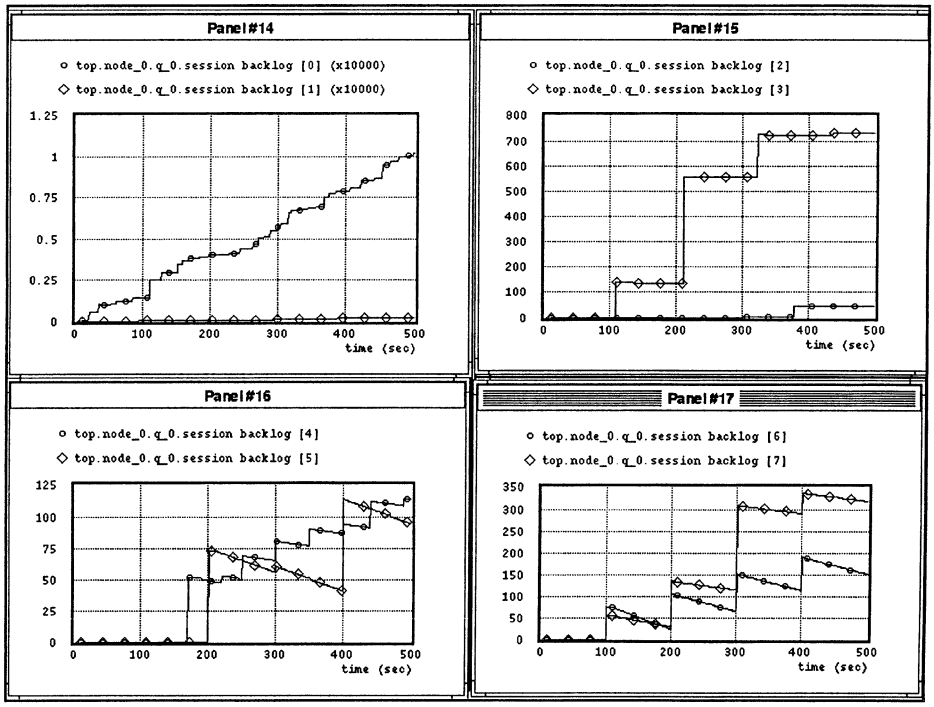

Fig. 7. Session backlog of cell PGPS.

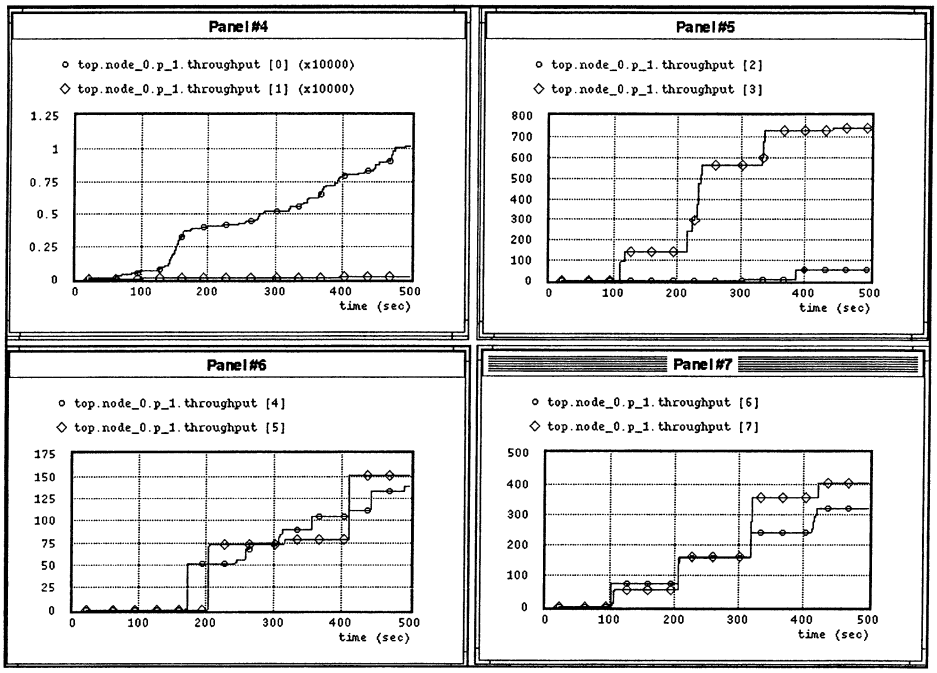

Fig. 8. Session throughput of BWFQ.

stream-like sessions. The performances of BWFQ have been measured and we also verify the properties of BWFQ (i.e., various service guarantee and fairness properties) as claimed in previous section. The simulation has been conducted based on both short-term and relatively long-term (for steady state behavior). In the end of the simulation, we alse give the computational efficiency comparison under the BWFQ and cell PGPS servers.

In order to make BWFQ practical it is necessary to distinguish the bursts. In [15] (and references therein) one uses a constant burst size for each session to approximate the real burst size. In [2] one passes the burst size information explicitly in the header cell of the burst or uses a marker cell to end a burst. In [7] and [9] one uses *flow-regulation* which makes the burst boundary distinguishable by the underlying network switches. We have used the flow-regulation method in our implementation. The details can be found in [4].

Our simulation model is based on a two-level ATM network environment consisting of backbone switches and access

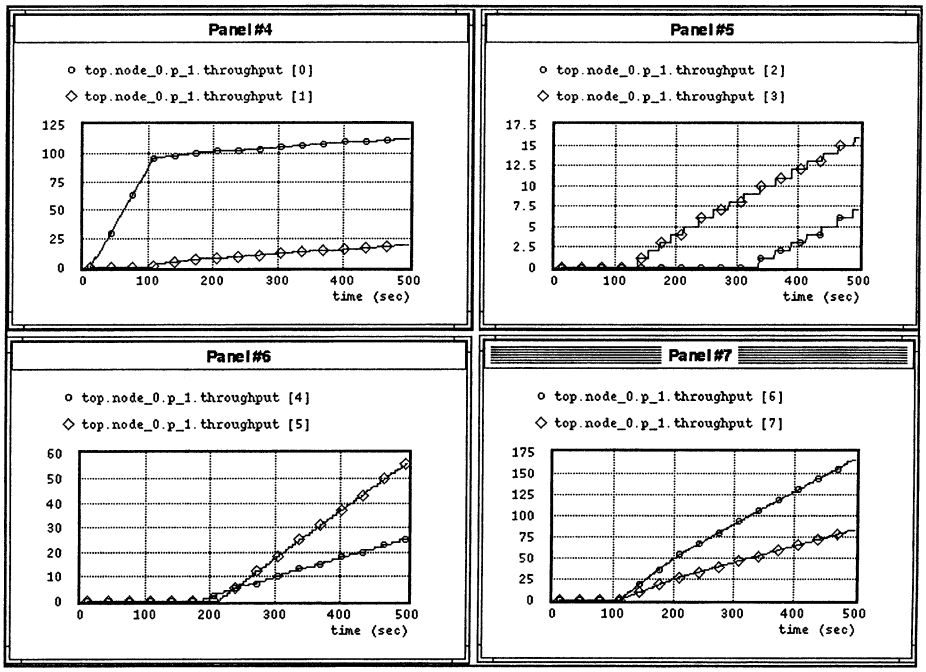

Fig. 9. Session throughput of cell PGPS.

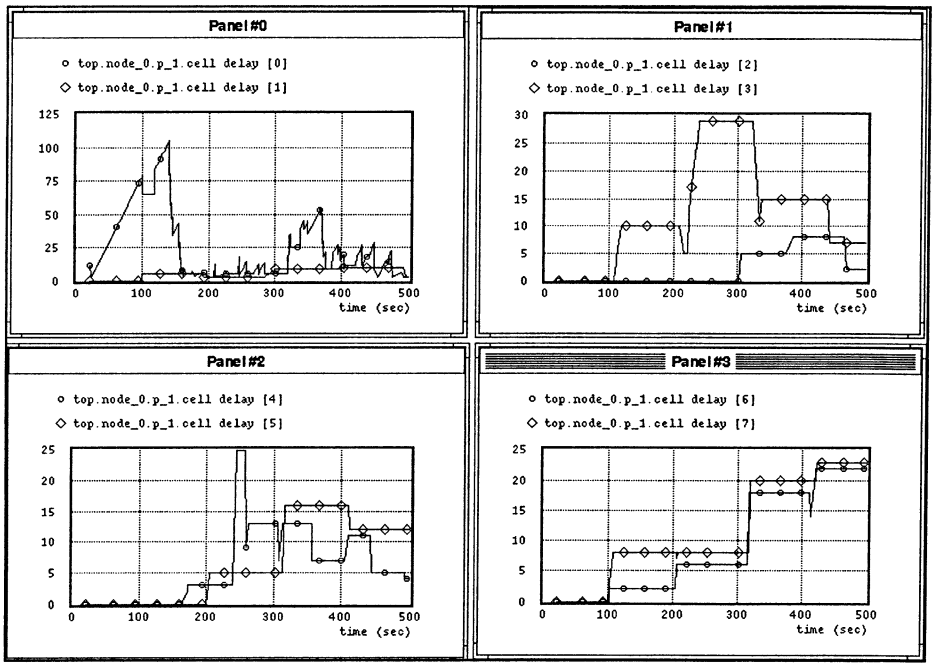

Fig. 10. Session delay of BWFQ.

switches. Our simulation package is OPNET modeler 3.0 B from MIL 3, Inc. It has a prominent advantage over other simulation packages, for it supplies adequate in-built C functions to support accurate algorithm implementation using a discrete-event mechanism, and it also supplies many flexible analysis tools [11], [12]. To help build a model representing a real-world network, OPNET allows a model specification complete with network, node, process, and parameter definitions to capture the characteristics of a modeled system's behavior at different modeling hierarchies. Geographically distributed sites are referred to as nodes or subnetworks. These nodes are connected through point-to-point links, bus links, or radio links in the OPNET network editor.

A simulation is executed by feeding the program a set of data files representing the model's parameters to dynamically model the behavior of the actual system. Predefined statistics of interest can be collected on sample simulation runs by using the Probe Editor to specify which built-in statistics should be recorded by the Simulation Kernel. Each simulation run can be viewed as an experiment on a certain group of parameters

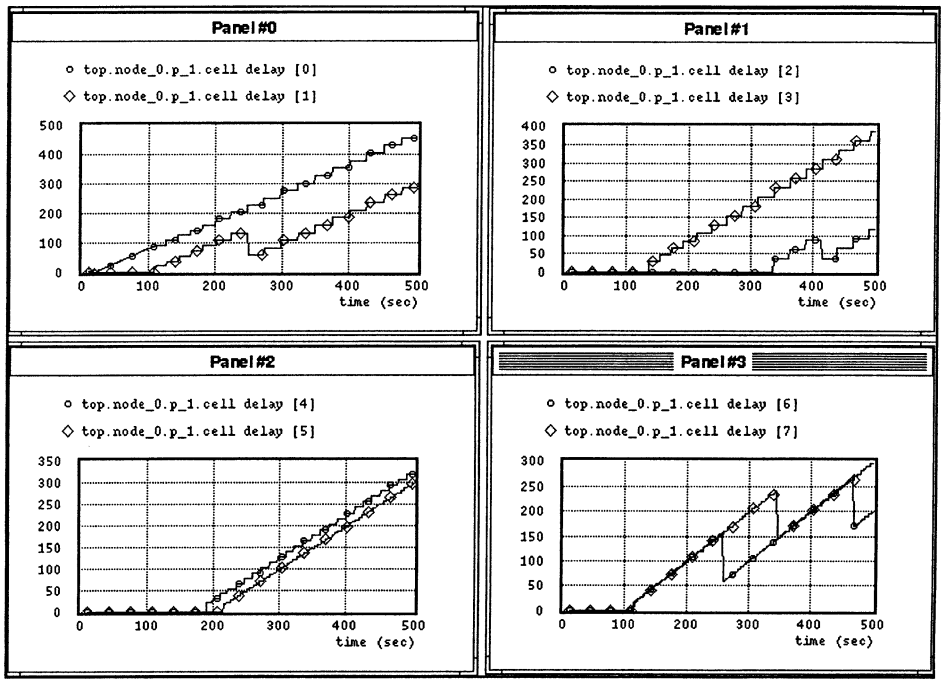

Fig. 11. Session delay of cell PGPS.

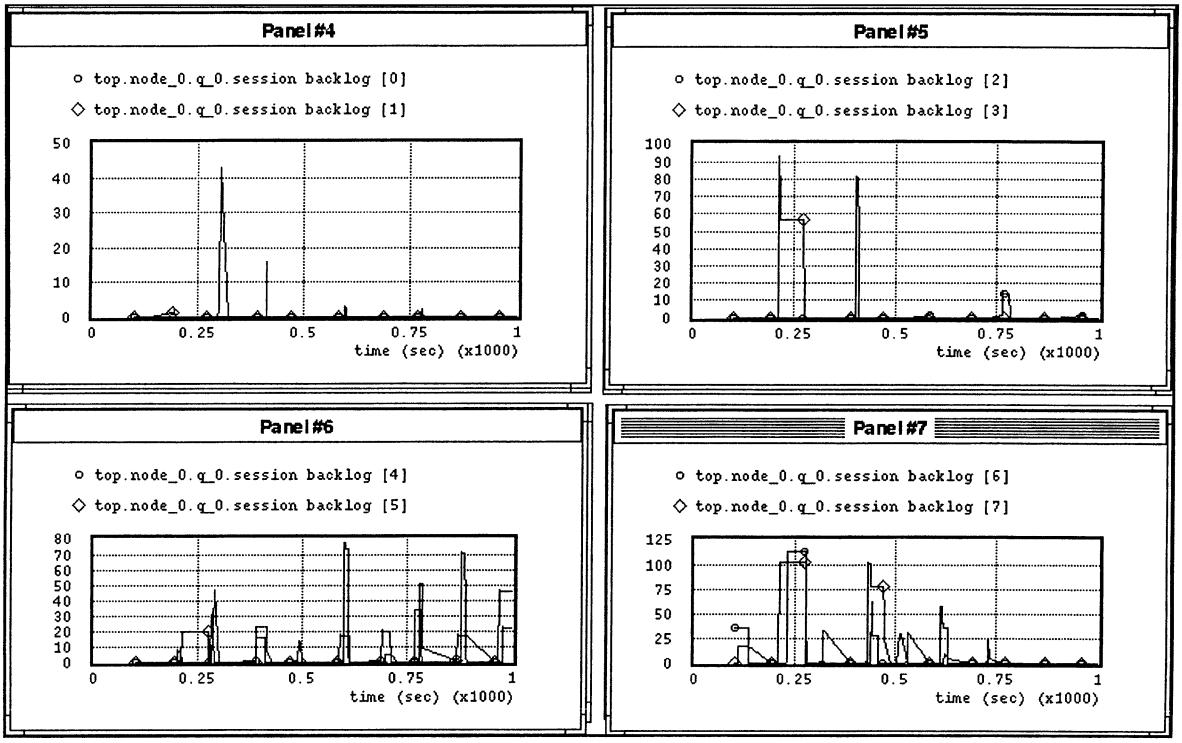

Fig. 12. Traffic arrivals of long last sessions.

of the system. It exhibits a repeatable behavior each time without changing the execution environment. This property is useful for isolating problems and analyzing or demonstrating interesting behavior. For simulating stochastic elements (for example, a packet generator module) however, different seed (one of the parameters) values should be used to produce distinct sequences so that each particular simulation run can be thought of as representing one possible scenario of events for the modeled system. Statistics of interest can be collected as output vector files during the simulation runs. Our simulation plots are generated this way. The ATM switch model design is shown in Fig. 3. Note: **AAL** = ATM adaptation layer; **MGMT** = traffic management; **ATM** = ATM layer; **FRM BKB** = from backbone switch; **To BKB** = to backbone switch; **To LOC** =

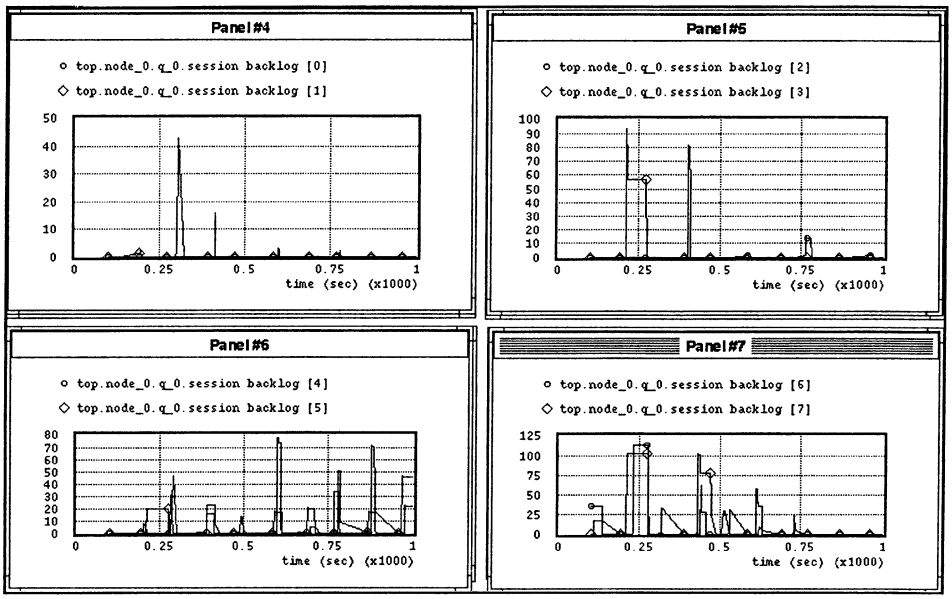

Fig. 13. Session backlog of BWFQ in steady state.

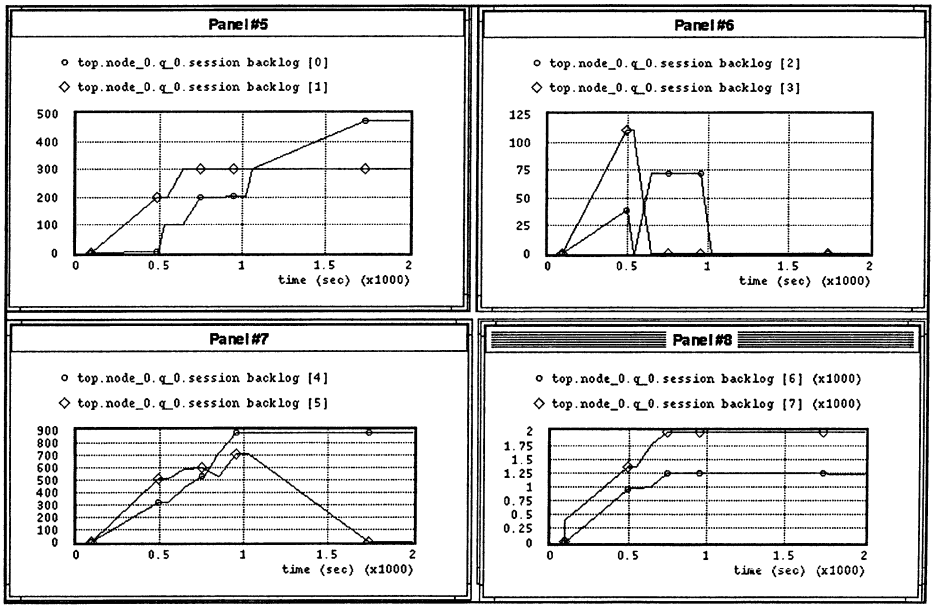

Fig. 14. Session backlog of cell PGPS in steady state.

to local switch; **Switch** = ATM switch fabric; **TRANS** = ATM cell transmission. The ATM layers are **Session #'s** (application layer), **AAL**, **MGMT/ATM**, and **TRANS/SWITCH** (physical layer). Our scheduling server is part of **AAL**.

The network topology for this simulation is shown in Fig. 4. Table V gives the source traffic parameters in our experiment with eight sessions.

In the figures (generated by OPNET), the ordinate unit is the *number of cells* for throughput (or for backlog) and *number of cell slots* for delay. The abscissa unit shows "time(sec)". This is not actual seconds, but the *simulation time* measured in *cell slots*. The number after the legend in each figure patch is the session number.

We compare our burst version BWFQ with the cell version. Through the simulation, we find that performance on queue delay, session throughput, and session backlog is almost the same. This means that under the new algorithm the buffer and source traffic usage parameter control (UPC) parameters need not change, and we can still obtain the required performance. Further work on the detail correlation between these algorithms (such as the cell loss) is needed. Here we only focus on the performance indices mentioned above.

We tested the performance of the two algorithms under the same source traffic pattern. The source traffic arrivals are measured in cells and shown in Fig. 5. Figs. 6 and 7 show the session backlogs under BWFQ and cell PGPS, respectively. The

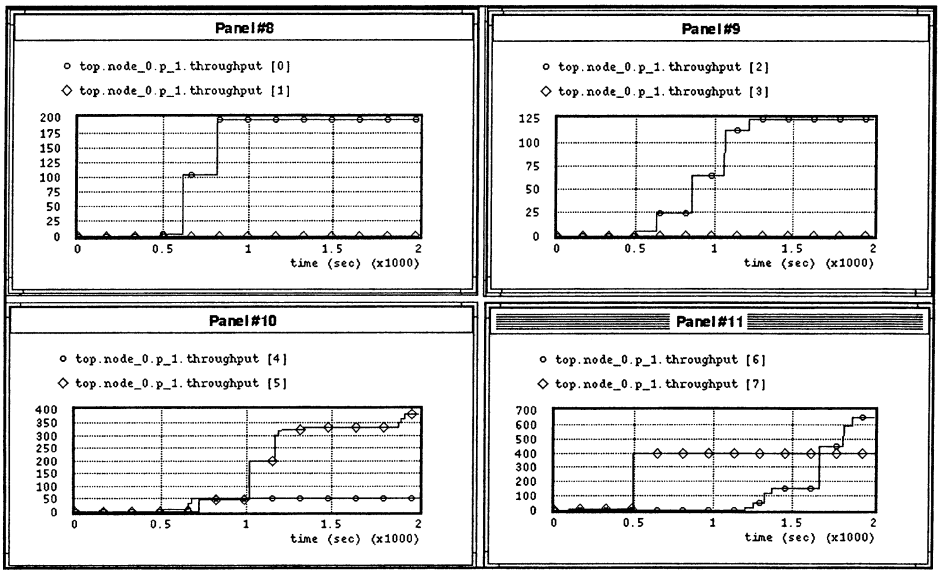

Fig. 15. Session throughput of BWFQ in steady state.

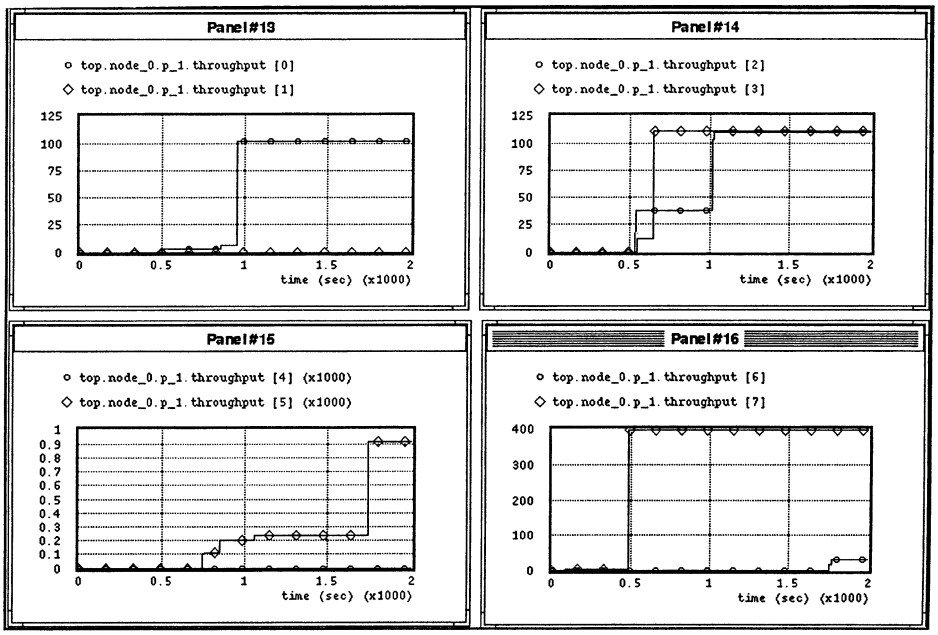

Fig. 16. Session throughput of cell PGPS in steady state.

backlog variation in the BWFQ servers is relatively larger than that in the cell PGPS server. This is reasonable because we schedule the cells in a burst (i.e., a group of cells in one time from one session, not one cell at one time) which may delay other bursts from other session. One noticeable advantage of BWFQ on session backlog is that the server has a shorter busy period in average than that of cell PGPS server. Figs. 8 and 9 show the session throughputs under BWFQ and cell PGPS, respectively. From the two figures, we can see the throughput increases in average for all the sessions, and for the best effort traffic sessions, session 2 and session 3 do not suffer from unfair comptition from other more aggressive sessions and they have minimal gauranteed service which indicates the fairness property of BWFQ. From this comparison, we also notice the throughputs of the real-time sessions are very consistent in a BWFQ server. In contrast, the throughputs in a cell PGPS server varies too dramatically, which could negatively affect a smooth playback of multimedia applications. Figs. 10 and 11 show the session delays under BWFQ and cell PGPS measured on a cell level, respectively. The delays from the eight sessions are all reduced on average under BWFQ when compared to a cell PGPS sever. Both cell PGPS and BWFQ servers experience delays roughly according to the session cell arrival pattern, however BWFQ can reduce the average delays according to the allocated bandwidth portions of each session from the session level.

In order to see the steady state behavior of BWFQ, we also have simulation runs which last a long period of time with different traffic mix (we also select the same 8 sessions for

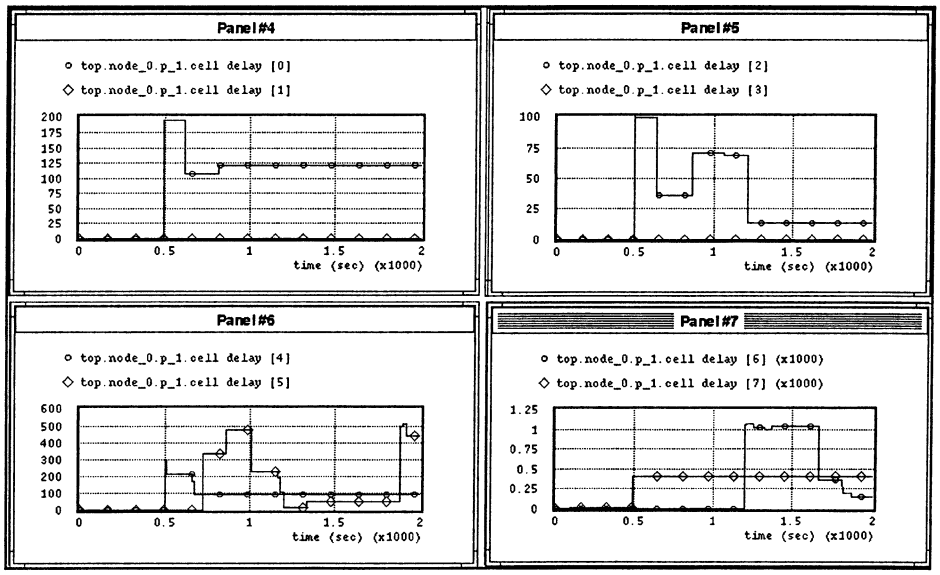

Fig. 17. Session delay of BWFQ in steady state.

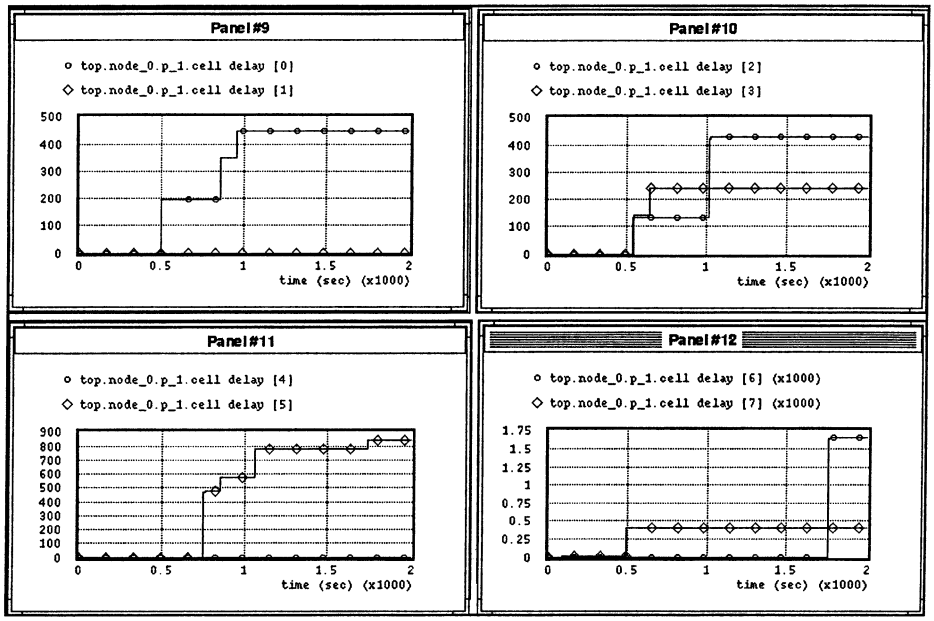

Fig. 18. Session delay of cell PGPS in steady state.

backlog, thoughput, and delay comparisons, the traffic arrival is shown in Fig. 12). Figs. 13 and 14 show the backlogs under BWFQ and cell PGPS servers, respectively. We can see from these two figures that the backlogs are reduced under BWFQ server. Backlog accumulates for some sessions after sometime and stay this for a long period of time under cell PGPS server. However, this never happens under BWFQ server. Figs. 15 and 16 show the thoughputs under BWFQ and cell PGPS servers, respectively. From these figures, the throughput performances increase under BWFQ compared to under cell PGPS and also the throughputs conform to the traffic arrivals and the allocated bandwidths to each sessions, i.e., bandwidth are more fairly shared among the competing sessions. Figs. 17 and 18 show the delays under BWFQ and cell PGPS servers, respectively. From these figures, the delays of the eight sessions are all reduced under BWFQ when compared to these under cell PGPS.

In the remainder of this simulation study, we shall compare the computation efficiency of BWFQ vs. cell PGPS. Figs. 18 and 19 show the virtual time progress and the number of invocations under BWFQ and cell PGPS servers, respectively. From the figures, the invocation of the cell PGPS algorithm is much larger than that of the BWFQ, note that one invocation under two servers has almost the same computation overhead and less invocation of the algorithms can shorten the switching latency of cells.

VI. Conclusion

A scheduling discipline plays a critical role in QoS. We have demonstrated that BWFQ is superior to the cell PGPS. It offers

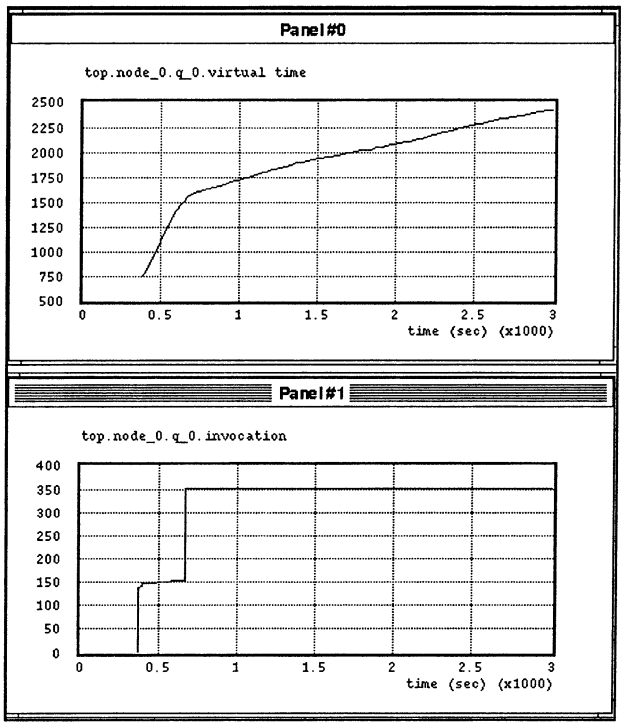

Fig. 19. Virtual time and number of invocations of BWFQ.

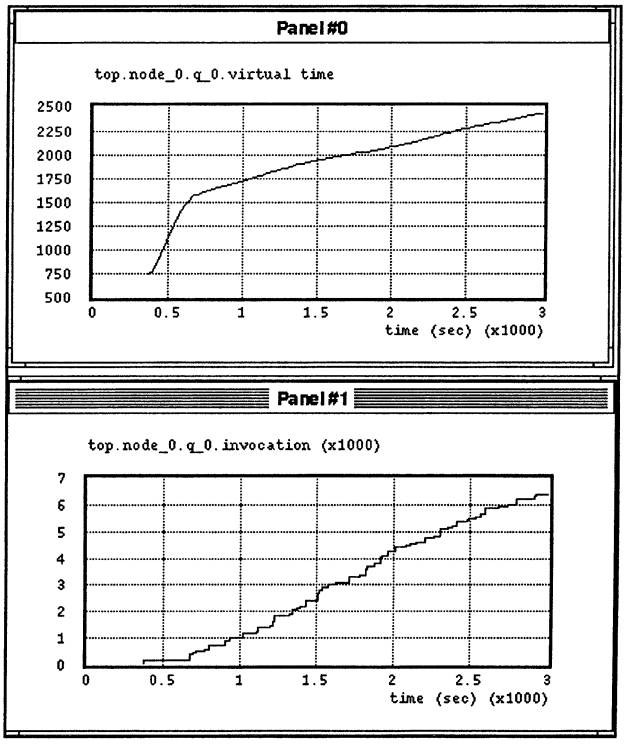

Fig. 20. Virtual time and number of invocations of cell PGPS.

an improved fairness and less delay for delay-sensitive applications. Since an ATM switch uses a small fixed-size cell as a switching unit, it expedites the cell switching process and reduces the switching delay, and it also makes communication synchronization easier. However, BWFQ needs a more complex queuing discipline than the standard FIFO and so the processing overhead is no longer negligible. The simulation study demonstrates that BWFQ is highly efficient in terms of the computation complexity and the major QoS indices.

REFERENCES

[1] J. Bennett and H. Zhang, "Hierarchical packet fair queueing algorithms," *IEEE/ACM Trans. Networking*, vol. 5, no. 5, pp. 675–689, Oct. 1997.

[2] P. E. Boyer and D. P. Tranchier, "A reservation principle with application to the ATM traffic," *Computer Networks and ISDN Systems*, vol. 24, no. 4, pp. 321–334, 1992.

[3] F. Chiussi, A. Francini, and J. Kneuer, "Implementing fair queueing in ATM switches—Part 2: The logarithmic calendar queue," *Proc. IEEE GLOBECOM'97*, pp. 519–525, 1997.

[4] A. T. Chronopoulos and C. Tang, "An efficient implementation of burst fair queuing for ATM networking," in *Proc. 10th IASTED International Conference on Distributed Systems*, Las Vegas, Nevada, Oct. 23–31, 1998, pp. 7–13. pp. 326–333.

[5] R. L. Cruz, "A calculus for network delay, part I: Network elements in isolation," *IEEE Trans. Information Theory*, vol. 37, no. 1, pp. 114–131, Jan. 1991.

[6] A. Demers, S. Keshav, and S. Shenkar, "Analysis and simulation of a fair queueing algorithm," *Internet Research and Experiment*, vol. 1, 1990.

[7] S. J. Golestani, "Network delay analysis of a class of fair queueing algorithms," *IEEE Trans. Selected Areas in Communications*, vol. 13, pp. 1057–1070, Aug. 1995.

[8] ——, "Congestion-free transmission of real-time traffic in packet networks," *Proc. IEEE INFOCOM 1990*, June 1990.

[9] P. Goyal and H. M. Vin, "Generalized guaranteed rate scheduling algorithms: A framework," *IEEE/ACM Trans. Networking*, vol. 4, no. 4, pp. 561–571, Aug. 1997.

[10] P. Goyal, H. M. Vin, and H. Cheng, "Start-time fair queueing: A scheduling algorithm for integrated services packet switching networks," presented at the SIGCOMM'96, Aug. 1996.

[11] *OPNET Modeler: Modeling/Vol. 1,/Vol. 2*.

[12] *OPNET Modeler: Simulation Kernel/Anim-Pk, /Prg-Topo*.

[13] 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 Trans. Networking*, vol. 1, no. 3, pp. 344–357, June 1993.

[14] D. Styliadis and A. Varma, "Efficient fair queueing algorithms for packet-switched networks," *IEEE/ACM Trans. Networking*, vol. 6, no. 2, pp. 175–185, June 1998.

[15] G. Xie and S. Lam, "Real-time block transfer under a link-sharing hierarchy," *IEEE/ACM Trans. Networking*, vol. 6, no. 1, pp. 30–41, Feb. 1998.

[16] L. Zhang, "VirtualClock: A new traffic control algorithm for packet-switched networks," *ACM Trans. Computer Systems*, vol. 9, no. 2, pp. 101–124, May 1991.

[17] T. Mizuike, Y. Ito, D. J. Kennedy, and L. N. Nguyen, "Burst scheduling algorithms for SS/TDMA systems," *IEEE Trans. Communications*, vol. 39, no. 4, Apr. 1991.

Anthony T. Chronopoulos received the Ph.D. at the University of Illinois in Urbana-Champaign in 1987. Presently, he is an Associate Professor at University of Texas in San Antonio. He is a senior member of IEEE. He has published 31 journal and 35 refereed conference proceedings publications in the areas of parallel and distributed computing and computer networks. He has been awarded twelve federal/state government research grants. His work is cited in more than 140 non-coauthors' research articles.

Caimu Tang received the B.S. from Xi'an Jiaotong University, Xi'an, China and M.S. from Wayne State University, Detroit, Michigan, in Applied Mathematics and Computer Science, respectively. Mr. Tang's current research interests are on algorithmic aspects of wireless sensor network and distributed collaborative signal processing.

Ece Yaprak is an Associate Professor in the Division of Engineering Technology at Wayne State University. She received the B.S. in Electrical Engineering in 1980 from the University of Michigan-Dearborn; the M.S. in Computer Engineering in 1984 and the Ph.D. in 1989 from Wayne State University. Dr. Yaprak joined the Division in August 1993. Her specialty areas are Computer Networking and Communications. Prior to joining WSU, she worked for Western Michigan University, General Electric, Ford Motor Company. During the last nine years she has held faculty fellowships at NASA Lewis Research Center, Jet Propulsion Laboratory of California Institute of Technology, NAVY/ASEE Summer Research Faculty Fellowship at the U.S. NAVY Space and Warfare Systems Center (SPAWAR) and NASA Ames Research Center of Stanford University.