# A Cell Burst Scheduling for ATM Networking Part I: Theory

C. Tang, A. T. Chronopoulos, Senior Member, IEEE
Computer Science Department
Wayne State University
email:ctang, chronos@cs.wayne.edu

E. Yaprak, Member, IEEE
Engineering Technology Division
Wayne State University
yaprak@et1.eng.wayne.edu

## Abstract

*Fair queueing is a useful queueing discipline for packet switching systems. It was developed in last decade and was aimed at the general packet switching systems with varying packet length. However, it is not suitable for use in the ATM networking, because the ATM cell length is very small and fixed, and so the scheduling scheme on a per cell basis isn't practical. Here we introduce the burst and quality unit concepts in the scheduling algorithm and we make some significant modification on the fair queueing and adapt it to ATM networking to meet QoS requirements. Under the* Work-Conserving *assumption, we show that the burst based non-preemptive and preemptive algorithms provide throughput and fairness guarantees.*

**Key Words** : *Fair Queueing, Cell Burst, Quality of Service (QoS), non-preemptive, preemptive, scheduling.*

## 1 Introduction

In this section, we give a preliminary description of a packet scheduling algorithm, its requirements, the general model, the terminology and notation used in the algorithm. There are several scheduling algorithms proposed for packet switched networking, e.g. [1], [2], [3], [4], [5], [6], [7].

Here we present the Burst-Based Weighted Fair Queueing (BBWFQ) for ATM cell scheduling and we investigate related QoS's guarantees under this algorithm. We study the algorithm performance through simulation in Part II.

### 1.1 Basic Requirements

We next state the basic requirements for a packet scheduling algorithm:

**Liveness Property:** Any packet will be scheduled eventually. In a switch node, some packets may be discarded due to buffer limitation and service regulation. As long as a packet arrives at the switch and gets buffered, it becomes eligible for scheduling. The scheduler should offer the service to the packet.

**Ordering Property:** All eligible packets will be scheduled in a proper order, for any sound scheduling algorithm. The order is well-defined based on the traffic models of the sessions.

**Safety Property:** A packet will be scheduled based on the information the switch has so far. Although this information can't contain any future arrival information to the switch, the decision made to schedule one packet is based on the scheduling scheme given.

**QoS guarantee:** an ATM network main advantage over other networks is the quality of networking characteristics. It can support different kinds of QoS. From the scheduling algorithm point of view, the delay, throughput and fairness are the major concerns here.

### 1.2 The General Model

To focus our research topic, we simplify the model of the system as follows: An ATM switch services a lot of 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. Thus regardless of their origin, the sessions are all termed incoming sessions to the scheduler. Figure 1 shows this simplified model. We will design the algorithms based on this model.

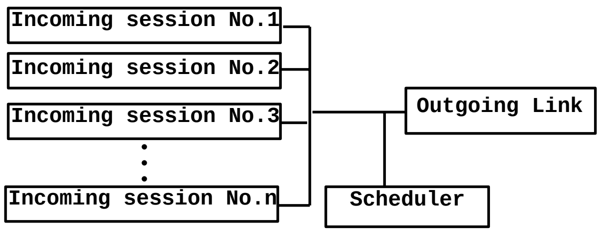


Figure 1: The algorithm based model

We make some assumptions for our model:
i) All sessions are independent. ii) There is a single scheduler for each output link in every switch. iii) Inside each session, we enforce a FIFO queue. iv) The throughput of the system bus and I/O bus is greater than the output link. So it's possible there are some cells from different sessions arriving simultaneously, i.e. in same cell slot. v) Our model uses output queueing.

## 1.3 Terms and Notations

**QoS Measurement Unit:** The data unit whose QoS is interesting to the end-user. For example, an I-frame in MPEG or one HTTP packet in Internet.
**Cell Burst:** A group of cells which come from one QoS measurement unit. These cells usually will cause a cell burst arrival in the incoming queues corresponding to this session. We sometimes call it **Burst** for simplicity.
**Burst Scheduling Eligible Time:** The time that the burst arrives at the switch and gets buffered.
**Burst Turnaround Time:** The time the burst finished its service in the switching unit, denoted by BTT.
**Burst Waiting Time:** The burst turnaround time minus the scheduling eligible time of the burst and the total service time of the burst, denoted by BWT.
**System Busy Period:** The time interval in which there is at least one active session, i.e. All cells are backlogged in a system period are served before the end of the system busy period.
**Session Busy Period:** The time interval in which this session is continuously backlogged, i.e. All cells are backlogged in a session busy period are served before the end of the session busy period. Note that a system busy period may consist of several session busy periods of different sessions.
**Work-Conserving Scheme:** A scheme in which the server isn't idle whenever there is cell backlogged in the server.

We will use following notation to explain our algorithm:
$\underline{(i,j)}$**:** The j-th burst in session i. $\underline{c(i,j)}$**:** The burst size of (i,j) in terms of cells. $\underline{a(i,j)}$**:** The arrival time of (i,j). $\underline{f(i,j)}$**:** The finish-time of (i,j) under GPS. $\underline{\hat{f}(i,j)}$**:** The finish-time of (i,j) under BBWFQ. $\underline{F(i,j)}$**:** The virtual finish-time of (i,j) under BBWFQ. $\underline{S(i,j)}$**:** The virtual start-time of (i,j) under BBWFQ. $\underline{NS(t_1,t_2)}$**:** The number of served cells during time interval $(t_1,t_2)$ in GPS server. $\underline{\widehat{NS}(t_1,t_2)}$**:** The number of served cells during time interval $(t_1,t_2)$ in BBWFQ server. $\underline{\tau_0}$**:** One cell slot, a constant that depends on the link capacity and processor of the switch.

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

We use an example to further illustrate some of these notations and terms. The example is described by giving the burst size and burst arrival/turnaround/waiting times in Table 1-4:

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

Table 1: The burst arrival time

| SN. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| BN. 1 | 3 | 10 | 5 | 3 | 10 | 5 | 20 | 10 |
| BN. 2 | 2 | 8 | 5 | 2 | 8 | 20 | 20 | 10 |
| BN. 3 | 3 | 8 | 5 | 2 | 8 | 20 | 20 | 10 |

Table 2: The burst size

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

Table 3: The burst turnaround time(BTT)

| SN. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| BN. 1 | 0 | 22 | 6 | 1 | 25 | 30 | 61 | 15 |
| BN. 2 | 1 | 45 | 10 | 5 | 43 | 97 | 97 | 59 |
| BN. 3 | 0 | 71 | 27 | 6 | 79 | 127 | 107 | 87 |

Table 4: The burst waiting time(BWT)

where, SN stands for Session Number and BN stands for Burst NUmber.

In this example, all the sessions have the same bandwidth portion. Notice that, in this example, the waiting time from session 1 and session 4 is relative shorter than the others, because these two sessions have smaller burst size and larger interarrival interval. So, the interarrival time distribution is also important for shorter delay besides the bandwidth portion. Here, BTT 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 session busy period. For example, if the source is a MPEG-2 codec, and there is an I-frame generated and sent out, that's the source generates a burst, all the intermediate nodes along the path will have a busy period.

For this example, we can see that the longer the BTT in each node, the longer the end-to-end delay the burst will experience. The delay jitter will also change. It doesn't make much sense to lower the delay of individual cell and the intercell jitter. Also, in packet switching systems other than ATM, like Frame Relay or SMDS, because of the packet size being much larger, it will be important to make the QoS guarantee on the packet level. This is the main difference between packet switching system and cell switching system in terms of the QoS analysis. It is also the reason we need to treat them differently. In order to control the delay and delay jitter, we need to control the BTT, there are several factors which affect the time. (i) The backlogged sessions at the beginning of scheduling cycle. (ii) The burst size and the total backlogged cells in this session queue. (iii) The bandwidth ratio available to the session if fair queue is used.

## 2 Burst Based Weighted Fair Queueing

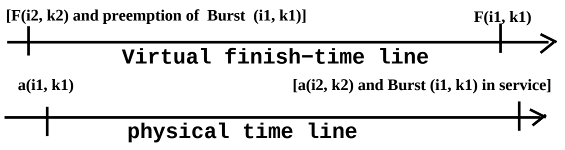
In this section, we will give two BBWFQ cell scheduling algorithms (nonpreemptive and preemptive). we then prove a discrepancy bound to the PS server and we also prove the delay and fairness guarantees. The simulation results will further demonstrate the advantages over the non-burst version which will appear in the next sections.

### 2.1 The Algorithms

Let's first give the description of the algorithm. Let $V(t)$ be a virtual time function [5]. Let i, k be the session-index and burst-number respectively, and $c(i,k)$ be the burst size. The algorithm steps are as follows:
Starting system busy period(at physical time $t_{start}$):
$S(i,0) = F(i,0) = 0$; for arbitrary session index i and $V(t_{start}) = 0$;
Burst arrival: 1) Burst-start:

$$c(i,k) = 0, S(i,k) = max\{F(i,k-1), V(a(i,k))\}.$$

2) Cell arrival: $c(i,k) = c(i,k) + 1$. 3) End of a burst $F(i,k) = S(i,k) + c(i,k) * \frac{\tau_0}{\phi_i}$ , where $\tau_0$ is the cell slot and $\phi_i$ is the bandwidth portion available for session i.
Burst departure(Non preemptive): The scheduler serves the bursts in sessions according to the ascending order of the $F(i,k)$ which are backlogged in the server so far.
Burst departure(Preemptive): The scheduler chooses the burst with smallest $F(i,k)$ among the backlogged sessions in the server. In case a burst with smaller virtual finish-time arrives, then the scheduler will preempt the currently served burst at the closest boundary of a cell and start to schedule this burst. The following figure gives a direct explanation to the preemptive scheme.


[F(i2, k2) and preemption of Burst (i1, k1)]
F(i1, k1)
Virtual finish-time line
a(i1, k1)
[a(i2, k2) and Burst (i1, k1) in service]
physical time line
Figure 2: Illustration of preemption scheme

Note that in the preemptive scheme for burst departure, starvation won't happen because the virtual finishing time of a burst is a kind of dynamic priority. For, the preempted burst will definitely be scheduled at time $t_{current} + C(i,k) * \tau_0/\phi_i$, where the burst $(i,k)$ is the preempted burst from session i. This also implies that the session's throughput is guaranteed. Following is an example to illustrate the difference between preemptive and non-preemptive algorithms. Let AT = arrival time, BS = burst time, ST = virtual start time, FT = virtual finish time and ST = scheduled time of the burst. In Table 5, we show the non-preemptive scheme scheduling results for a system with three sessions.

| | *session-1*($\frac{1}{4}$) | | | *session-2*($\frac{1}{4}$) | | | *session-3*($\frac{1}{2}$) | | |
|---|---|---|---|---|---|---|---|---|---|
| AT | 1 | 8 | 17 | 0 | 9 | 20 | 6 | 10 | 15 |
| BS | 7 | 5 | 4 | 5 | 4 | 5 | 4 | 4 | 4 |
| ST | 2 | 30 | 50 | 0 | 20 | 36 | 9 | 17 | 33 |
| FT | 30 | 50 | 66 | 20 | 36 | 56 | 17 | 33 | 49 |
| ST | 12 | 33 | 42 | 5 | 24 | 38 | 16 | 20 | 28 |

Table 5:The preemptive scheme example
The sessions portions of the total bandwidth are $(\frac{1}{4})$, $(\frac{1}{4})$ and $(\frac{1}{2})$ respectively. We notice that the first burst in the third session is finished at time slot 16 instead of at time slot 9, because it isn't eligible to schedule at time 5 even though it has a smaller virtual finish-time in the this example. But for the preemptive scheme, the first burst from session-1 is scheduled at time slot 5, and after it has one slot service, its service is preempted by the first burst from session-3 because its virtual finish-time is less than the currently served burst.

It's obvious that the preemptive scheme is a better approximation to GPS and it has better QoS properties than the non preemptive, but it is hard to give a strict mathematical proof as pointed out in [5]. So, simulation is a good way to demonstrate this.

We next give several results about the discrepancy bound between the algorithm and PS algorithm. Here, we assume the algorithm is provided with the burst boundary before it starts. The following two lemmas and two theorems are useful for these results. We don't give proofs here. Similar results are proved for PGPS in [5] and their extentions to BBWFQ are easy.

**Lemma 1** *Assume that the scheduling algorithm is work-conserving, then scheduling orders of different busy periods are unrelated.* □

By using this lemma, we only need to consider the behavior of one system busy period in order to study the scheduling of the system. We consider the burst GPS which is similar to the GPS defined in [5]. The only difference is that the server starts to serve one burst once it arrived instead of serving a single cell which arrived as in non-burst GPS. Furthermore, we assume that the burst boundary is known to the server. In order to give the main result, we need the following lemma as follows:

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

The following two theorems state how good the approximation is. Notice that the theorems hold for non-preemptive scheme. As for the preemptive scheme, there are reasons the approximation is better, we will present results later.

**Theorem 1** *If the scheduling scheme is nonpreemptive, then for all bursts $(i, j)$,*

$$\hat{f}(i,j) - f(i,j) \le \tau_0 * (\max_{(p,k)} c(p,k)). \quad (1)$$

**Proof:** We consider a fixed burst $(i, j)$. We prove the result for it. 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 BBWFQ 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: $(i,j) \le (\tilde{i},\tilde{j})\ iff\ a(i,j) \le a(\tilde{i},\tilde{j})$,

where $a(.,.)$ is the arrival time of burst $(.,.)$. 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

$$f(i,j) \ge \hat{f}(i,j)$$

   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 $f(p,k) > f(i,j)\} \ne \phi$.
The set Q is finite because the total number of bursts arrived before $a(i,j)$ is finite. So, there exists a burst $(\tilde{i},\tilde{j}) \in Q(i,j)$, such that

$$(\tilde{i},\tilde{j}) \ge (p,k)$$

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

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

This means that all the bursts in $\{(p,k)|(\tilde{i},\tilde{j}) < (p,k) \le (i,j)\}$ 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

$$f(i,j) \ge \hat{f}(\tilde{i},\tilde{j}) - c(\tilde{i},\tilde{j}) * \tau_0 + \sum_{(\tilde{i},\tilde{j})<(p,k)\le(i,j)} \{c(p,k)\tau_0\}, \quad (2)$$

and

$$\hat{f}(\tilde{i},\tilde{j}) + \sum_{(\tilde{i},\tilde{j})<(p,k)\le(i,j)} \{c(p,k)\tau_0\} = \hat{f}(i,j). \quad (3)$$

Because BBWFQ is work-conserving. So by( 2) and ( 3), $f(i,j) \ge \hat{f}(i,j) - c(\tilde{i},\tilde{j})\tau_0$.
Therefore, $\hat{f}(i,j) - f(i,j) \le \tau_0 * max_{(p,k)} c(p,k)$.
That is the inequality( 1). □
Note that, in a finish-time fair queueing, the two cases may occur: 1. the finish-time under GPS is greater than the finish- time under BBWFQ and 2. the finish-time under GPS is less than the finish time under BBWFQ. In case 1., the right-hand side of inequality in 1. is negative, which means that BBWFQ scheduled burst will finish earlier than GPS finish-time. So this theorem tells us that either the finish-time under BBWFQ is earlier than GPS (case 1.) or the finish-time under BBWFQ is later than GPS finish-time but their difference is uniformly bounded(case 2.).
**Notation**: $C_{max} = \max_{(p,k)} c(p,k)$.

**Theorem 2** *If the scheduling scheme is nonpreemptive, for any time $\tau$, and session i, we have*

$$NS_i(0,\tau) - \widehat{NS}_i(0,\tau) \le C_{max},$$

*where $NS_i(\tau,t)$ and $\widehat{NS}_i(\tau,t)$ are the number of cells of session i served under GPS and BBWFQ, in the interval $[\tau,t]$, respectively.* □

The proof of Theorem 2 is similar to an analogous theorem proved in [5]. From the previous theorem, we get the following throughput guarantee of BBWFQ:

**Corollary 1** *If the scheduling scheme is nonpreemptive, in the BBWFQ server, the session i can have a throughput guarantee as:*

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

*where the $\phi_i$ is the bandwidth portion in a normalized bandwidth allocation scheme.*

**Proof:** Notice that $NS_i(0,t) \ge \phi_i * \frac{t}{\tau_0}$ as a minimum service guarantee by the definition of GPS [5]. The Corollary follows by applying Theorem 2. □

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

The following theorem is quite straightforward, and it demonstrates the advantage of the preemptive scheme over the non-preemptive scheme. But the bound isn't as tight as our next result.

**Theorem 3** *If the scheduling scheme is preemptive, then for all bursts $(i, j)$, we have*

$$\hat{f}(i,j) - f(i,j) \le C_{max} * \tau_0,$$

*Furthermore, the burst service order is same in both GPS and BBWFQ.*

**Proof:** We first show that an equivalent non-preemptive priority queue system can be constructed for the preemptive queue system by splitting preempted bursts into two parts with first part being scheduled and second part being preempted.

Throughout this proof, we denote, for the n-th preemption, the burst in service as $(i_n, j_n)$, and the corresponding new arrival burst which has smaller virtual finishing time than burst $(i_n, j_n)$ as $(\tilde{i_n}, \tilde{j_n})$, (i.e. $\hat{F}(\tilde{i_n}, \tilde{j_n}) < \hat{F}(i_n, j_n)$). We also denote by $t(i_n, j_n)$ the service-starting time of burst $(i_n, j_n)$. For $n = 1$, the first preemption in the queue system, we prove there is no preemption after the splitting. It is easy to see (under our assumption) that the following inequalities hold:

$$t(i_1, j_1) < a(\tilde{i_1}, \tilde{j_1}) < \hat{f}(i_1, j_1),$$

under our assumption. When the preemption occurs, we split burst $(i_1, j_1)$ into two parts $(i_1, j_1^a)$, $(i_1, j_1^b)$ with lengths equal to $\frac{a(\tilde{i_1}, \tilde{j_1}) - t(t_1, j_1)}{\tau_0} * B$ cells and the remaining cells of the burst $(i_1, j_1)$, respectively. This new priority queue system will have no preemption upon and inclusive $a(\tilde{i_1}, \tilde{j_1})$.

Assume before the k-th preemption, there is no preemption after our rearrangement of bursts. There are two cases:

1. At times $a(\tilde{i_k}, \tilde{j_k})$ and $a(\tilde{i}_{k+1}, \tilde{j}_{k+1})$, the burst in service originated from the same burst $(i_k, j_k)$.
2. The bursts in service at times $a(\tilde{i_k}, \tilde{j_k})$ and $a(\tilde{i}_{k+1}, \tilde{j}_{k+1})$ are from different bursts.

For case 1., the burst $(i_k, j_k)$ will turn into three smaller bursts with lengths: $\frac{a(\tilde{i_k}, \tilde{j_k} - t(i_k, j_k))}{\tau_0} * B$, $\frac{a(i_{\tilde{k+1}}, j_{\tilde{k+1}} - t(i_k, j_k^b)}{\tau_0} * B$, and the remaining cells, respectively. For case 2., using the same treatment as in the base step of the induction, we can change the preempted bursts occured between $a(\tilde{i_k}, \tilde{j_k})$ and $a(\tilde{i}_{k+1}, \tilde{j}_{k+1})$ to non preempted bursts. So, after the $(k+1) - th$ preemption, there is still an equivalent non-preemption priority queue. We proved the claim by induction. By applying to Theorem 2, we finished the first part of the theorem.

Assume two bursts $(n, i)$, $(m, j)$ and $f_{GPS}(n, j) \geq f_{GPS}(m, j)$, we have two cases to consider:

1. $a(n, i) < a(m, j)$: If $(n, i)$ is scheduled in $[a(n, i), a(m, j)]$. After its arrival, $(m, j)$ will preempt the $(n, i)$ because the finish-time of $(m, j)$ under GPS is earlier than that of $(n, j)$. If the scheduling decision is made after arrival of $(m, j)$, the scheduler will choose $(m, j)$. Here we assume the scheduler will give preference to the burst which hasn't received service for the longest period of time if a tie exists. Note that a tie may occur even if there is no tie in the non-preemptive server.
2. $a(n, i) \geq a(m, j)$: No preemption occurs between $(n, i)$ and $(m, j)$ at any time. Thus burst $(m, j)$ always gets service before burst $(n, i)$ does, because our assumption $f_{GPS}(n, j) \geq f_{GPS}(m, j)$.
   Therefore burst $(m, j)$ finishes before $(n, i)$ does under GPS.

So, $f_{BBWFQ}(n, i) \geq f_{BBWFQ}(m, j)$. $\square$

From this theorem, we know not only the discrepancy is bounded between GPS and BBWFQ under the maximum length of the bursts in the server but also the serving order is preserved. Obviously, the approximation is much better than non-preemptive case.

The next theorem gives another discrepancy bound for the preemptive scheme.

**Theorem 4** *Assume the scheduling scheme is preemptive and assume burst A from session i arrives at time $a(A)$ with size $c(A)$. If A finishes at $f_{GPS}(A)$ under GPS, and at time $f_{BBWFQ}(A)$ under BBWFQ, then*

$$f_{GPS}(A) - f_{BBWFQ}(A) \leq (\frac{c(A)}{\phi_i} - \triangle) * \tau_0,$$

*where $\triangle$ is the summation of lengths of bursts, which arrive after the arrival of A and depart before the departure of A under GPS (i.e. all the bursts which preempt A under BBWFQ) and $\phi_i$ is the bandwidth portion for session i.*

**Proof:** Since BBWFQ is work-conserving and the scheme is preemptive,

$$f_{BBWFQ}(A) \geq a(A) + (\triangle + c(A))\tau_0, \tag{4}$$

Let set $P = \{p_i : i = 1, \cdots, m,$ with bandwidth proportion $\phi_i\}$ be the burst set which arrives before arrival of burst A and departs after arrival of A, and let set $Q = \{q_i : i = 1, \cdots, n,$ with bandwidth proportion $\hat{\phi_i}\}$ be the burst set which arrive after the arrival of A and departs after departure of A. Notice that the intersection of P and Q is empty and the bursts in set P are preempted bursts, and the bursts in set Q can't preempt burst A. Since $[\frac{1}{\tau_0} * \frac{\phi_i}{\phi_i + \sum_{j=1}^m \phi_j}]$ is the rate for burst A during time interval $[a(q_1), a(A)]$, and by the definition of GPS,

$$\begin{aligned} c(A) = NS_i(a(A), f_{GPS(A)}) \geq \frac{1}{\tau_0} * [ \\ \frac{\phi_i}{\phi_i + \sum_{j=1}^m \phi_j}(a(q_1) - a(A)) + \\ \frac{\phi_i}{\phi_i + \hat{\phi_1} + \sum_{j=1}^m \phi_j}(a(q_2) - a(q_1)) + \\ \vdots \\ \frac{\phi_i}{\phi_i + \sum_{j=1}^{n-1} \hat{\phi_j} + \sum_{j=1}^m \phi_j}(a(q_n) - a(q_{n-1})) + \\ \frac{\phi_i}{\phi_i + \sum_{j=1}^{n} \hat{\phi_j} + \sum_{j=1}^m \phi_j}(f_{GPS}(A) - a(q_n))]. \end{aligned} \tag{5}$$

By ( 5), we have

$$f_{GPS}(A) \le a(q_n) + \frac{(c(A) - \frac{1}{\tau_0} * \oplus)}{\frac{1}{\tau_0} * \frac{\phi_i}{\phi_i + \sum_{j=1}^{m} \phi_j + \sum_{j=1}^{n} \hat{\phi_j}}}, \quad (6)$$

where $\oplus = \sum_{k=1}^{n} (\frac{\phi_i}{\phi_i + \sum_{j=1}^{m} \phi_j + \sum_{j=1}^{k} \hat{\phi_j}} *$

$$(a(q_k) - a(q_{k-1}))).$$

in the discussion above, we assume $q_0 = A$. Since $\sum_{k=1}^{n} \frac{\phi_i + \sum_{j=1}^{m} \phi_j + \sum_{j=1}^{n} \hat{\phi_j}}{\phi_i + \sum_{j=1}^{m} \phi_j + \sum_{j=1}^{k} \hat{\phi_j}} * (a(q_k) - a(q_{k-1}))$

$$\ge \sum_{k=1}^{n} (a(q_k) - a(q_{k-1})) \ge a(q_n) - a(A). \quad (7)$$

we have:

$$f_{GPS}(A) \le \tau_0 * (\frac{\phi_i + \sum_{j=1}^{m} \phi_i + \sum_{j=1}^{n} \hat{\phi_j}}{\phi_i}) * c(A) + a(A). \quad (8)$$

By ( 4) and ( 8), we have:

$$f_{GPS} - f_{BBWFQ} \le (\frac{c(A)}{\phi_i} - \triangle) * \tau_0 \ ,$$

because $\phi_i + \sum_{i=1}^{m} \phi_i + \sum_{j=1}^{n} \phi_j \le 1$ , if we use a normalized bandwidth portions. However if we don't use the normalized bandwidth portion, we need replace the $\phi_i$ by the bandwidth percentage available to the session i. □

Note that we give another bound under the preemptive scheme for case 1. as we mentioned in the comments after Theorem 1. This means that if BBWFQ scheduled burst finishes earlier than GPS finish-time, the difference of the two finish-times is bounded. From the theorem above, we find that all the bursts which contribute to the $\triangle$ term have a shorter burst size than the burst A, the total size is hard to compare with the size of burst A. But by Theorem 2, we know the service order is kept between BBWFQ and GPS under preemptive scheme. So the summation length $\triangle$ is bounded at least by the maximum size of bursts in this session.

## 2.2 Fairness Property

It is easy to see that in GPS, a burst obtains its service immediately after its arrival, and the service rate depends on the currently backlogged sessions and on its own available bandwidth portion $\phi_i$. This queueing scheme never overuses/underuses its service. Therefore, GPS is the absolutely fair queueing. As a approximation of GPS, BBWFQ can't guarantee the same level of fairness as GPS, because BBWFQ never take into account the future arrival of bursts. However, BBWFQ can still provide the guaranteed fairness under our fairness definition.

**Definition 1** *Let $NS_i(t_1, t_2)$ be the number of served cells during interval $[t_1, t_2]$ when session i is continuously backlogged under a scheduling algorithm, we define the fairness index of the server as:*

$$\alpha = \max\{\left| \frac{NS_i(t_1,t_2)}{r_i(t_2 - t_1)} - \frac{NS_j(t_1,t_2)}{r_j(t_2 - t_1)} \right|\},$$

*where i, j are arbitrary sessions, which are continuously backlogged in $[t_1, t_2]$}*

Note that for a very specific case, if there is no overlap on backlogged periods, it's meaningless to consider fairness. In the 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 actually shows that this definition is well-defined.

**Proposition 1** *For the GPS server, $\alpha$=0.*

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

$$\frac{NS_i(\tau, t)}{NS_k(\tau, t)} \ge \frac{\phi_i}{\phi_k},$$

holds for $k = 1, 2, \cdots, N$.

Since sessions i, j are backlogged in $[t_1, t_2]$,we have $\frac{NS_j(t_1,t_2)}{NS_i(t_1,t_2)} \ge \frac{\phi_j}{\phi_i}$ .

So, $\frac{NS_i(t_1,t_2)}{\phi_i} = \frac{NS_j(t_1,t_2)}{\phi_j}$ . This means:

$\left| \frac{NS_i(t_1,t_2)}{\phi_i} - \frac{NS_j(t_1,t_2)}{\phi_j} \right| = 0$ . So, $\alpha = 0$. □

The following theorem gives the fairness guarantee under BBWFQ.

**Theorem 5** *For the BBWFQ server and both schemes: preemptive and non-preemptive, the following fairness guarantee holds,*

$$\alpha \le \frac{1}{\tau_0} * \frac{1}{\min_{i,j \in N}\{\max(r_i, r_j)\}},$$

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

**Proof:** By the definition of the fairness index, for arbitrary two 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 $[t_1, t_2]$.
(iv) Both session i, j receive service in interleaved order in interval $[t_1, t_2]$.

In case 1,

$$\left|\frac{NS_i(t_1,t_2)}{r_i(t_2-t_1)} - \frac{NS_j(t_1,t_2)}{r_j(t_2-t_1)}\right| = \left|\frac{NS_i(t_1,t_2)}{r_i(t_2-t_1)}\right| = \frac{1}{\tau_0 r_i}.$$

In case 2,

$$\left|\frac{NS_i(t_1,t_2)}{r_i(t_2-t_1)} - \frac{NS_j(t_1,t_2)}{r_j(t_2-t_1)}\right| = \left|\frac{NS_j(t_1,t_2)}{r_j(t_2-t_1)}\right| = \frac{1}{\tau_0 r_j}.$$

In case 3, $\left|\frac{NS_i(t_1,t_2)}{r_i(t_1-t_2)} - \frac{NS_j(t_1,t_2)}{r_j(t_2-t_1)}\right| = 0$ .
In case 4,

$$\left|\frac{NS_i(t_1,t_2)}{r_i(t_2-t_1)} - \frac{NS_j(t_1,t_2)}{r_j(t_2-t_1)}\right| \le \frac{1}{\tau_0} * \min(\frac{1}{r_i}, \frac{1}{r_j}) ,$$

because, for k=i or j,

$$\left|\frac{NS_k(t_1,t_2)}{r_k(t_2-t_1)}\right| \le \frac{1}{r_k} * \frac{1}{\tau_0} .$$

Therefore, we proved our claim. □
An analogous but less general result has been presented for packet switching in [8].

## 3 Conclusion

The scheduling discipline plays a critical role for QoS. We demonstrated that BBWFQ is superior to the cell PGPS. It can offer a better fairness and lower delay for delay sensitive applications. Since ATM switch uses a small fixed-size cell as a switching unit, it expedites the cell switching process and lowers the switching delay and it is also easier for the communication synchronization. Then, we have to use a more complex queuing discipline than the FIFO and so the processing overhead is no longer negligible. Our algorithms use the burst unit to reduce the processing time for each session, thus the overall performance is better than other scheduling algorithms.

**Acknowledgement:** Some comments of anonymous referees, which helped improve the presentation of some parts of the paper are gratefully acknowledged. This work was supported in part by NSF Grant ASC-9634775.

## References

[1] R. L. Cruz, ”A calculus for network delay, Part I: network elements in isolation and Part II: network analysis”, *IEEE Transactions on Information Theory, vol.37 No.1 Jan. 1991, pp. 114-142.*

[2] S.J. Golestani, ”Congestion-free transmission of real-time traffic in packet networks”, *proc. IEEE Infocom 90, June 1990.*

[3] P. Goyal, H. M. Vin, Haichen Cheng, ”Start-time Fair Queueing: A scheduling algorithm for integrated services packet switching networks”, *SIGCOMM’96 8/96.*

[4] S. Lam and G. Xie, ”Burst scheduling networks”, *Technical Report TR-95-28, University of Texas at Austin, July 1995.*

[5] 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, June 1993, pp. 344-357.*

[6] H. Zhang et al. ”$WF^2Q$: Worst-case fair weighed fair queuing”, *In Proc. IEEE Infocom’96, March 1996.*

[7] L. Zhang, ”VirtualClock: A new traffic control algorithm for packet-switched networks”, *ACM Transactions on Computer Systems, Vol.9, No.2 May 1991, pp.101-124.*