Traffic Optimization at UNI of ATM Networks

Yuan Miao[1], Chee Kheong Siew[1], Zhonghua Yang[2]

[1]*Information Communication Institute of Singapore, School of Electrical and Electronic Engineering*
Nanyang Technological University (S2-B3)
Nanyang Ave. Singapore 639798
Email: eymiao@ntu.edu.sg

[2]*School of Computing and Information Technology*
Griffith University, Nathan
Qld 411 Australia
Email: z.yang@cit.gu.edu.au

Abstract: In this paper, we propose a new concept, relative burst index, to describe the dynamic burst status of ATM networks. The burst index takes into account both the number of delayed cells and the delay they have experienced. Therefore it describes the burst characteristics of traffic more precisely than existing notions do. These notions represent either only cell backlogs or only delays. An optimal algorithm: Dynamic Multiplexing Algorithm (DMA) is presented to minimize the relative burst index and smooth out the traffic at the User Network Interface (UNI). The maximum relative burst index and maximum delay of DMA are derived. The performance improvement is analyzed. The comparative study with the well-known Generalized Process Sharing (GPS) method is also illustrated by simulations.

Key Words: ATM, Traffic, Multiplexing, UNI, Optimization

1. Introduction

The rapid development of the data networks has surprised many observers. Compared with the bandwidth of only a few kilo-bps or less several years ago, the networks with mega-bps bandwidth are now a commonplace. However, this development does not match with the ever-increasing requirement of bandwidth. People are no longer satisfied with simply exchanging email over networks. Within a short period of time, the diversity of Internet applications appeared such as reading news, browsing various resources via WWW, chatting over Internet (Internet Phone), listening to online music service, holding multimedia conferences and so on. All these impose greater demands on the network bandwidth and quality of service (QoS).

Traffic flow in networks has to be controlled. Otherwise, network users can pour traffic into networks at will. This uncontrolled traffic flow tends to result in a significant delay and dramatic degradation of QoS of the networks. To ensure the availability of bandwidth over the Internet, one can lease a line and have all the bandwidth. But this is expensive and defeats the purpose of the Internet and often beyond the reach for most network users. Another approach is to apply for a connection with a predefined QoS (Quality of Service) in an ATM Network. With the ATM traffic control, ATM networks can provide end to end virtual paths or channels with guaranteed QoS. Though, if no optimization is made, this approach leads to a lot of bandwidth wastage and therefore a high cost.

The bandwidth wastage over ATM networks is mainly due to the uncertainty and burstiness of traffic. Usually, the traffic flow over networks can be described by parameters such as SCR (sustainable cell rate), PCR (peak cell rate), MBS (maximum burst size) and MCD (maximum cell delay). Among these parameters, MCD describes the time sensitivity of a connection; PCR/SCR and MBS describe the traffic burstiness. For the diverse applications, the PCR/SCR may range from 1 to 200. To see how the burstiness leads the bandwidth wastage, let us consider a simple example. Suppose PCR/SCR=20, SCR=5 cells/ms (2Mbps). This application requires the average bandwidth of only 2 Mbps. However, the bandwidth allocation needs to be 40 Mbps to satisfy the PCR. This obviously results in a bandwidth waste.

For better bandwidth utilization, various types of traffic shaping, scheduling approaches have been developed [1-9]. These traffic control strategies are usually integrated into the network devices at User Network Interface (UNI). UNI of ATM networks is where all types of traffic flow into the high speed network trunks (Figure 1). These devices thus become traffic managing devices.

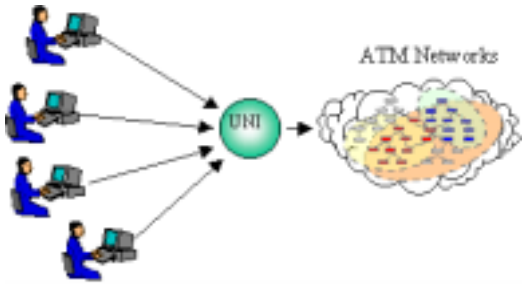


Figure 1. The User Network Interface

Depending on whether a traffic policer or enforcer is used, we can roughly divide the traffic-managing devices into two categories: multiplexers and shapers.

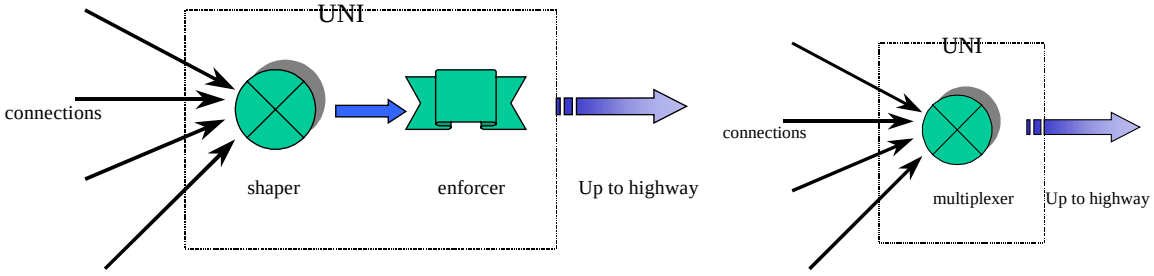


Figure 2. Multiplexer and Shaper

While a multiplexer trys to send out the incoming traffic as soon as possible, a traffic shaper smoothes out the input traffic so that they conform to their traffic parameters. They will then not be tagged or dropped by the enforcer. This paper focuses on traffic optimization of multiplexers.

This paper will propose a concept: burst index, which can describe the dynamics of bursty traffic. Based on the index, we propose a dynamic multiplexing algorithm: DMA. The

maximum delay and burst indexes are theoretically derived. Simulations show that DMA improves Generalized Process Sharing (GPS [3]) in term of less maximum delay and more fairness among cells. The paper is organized as follows: Section 2 briefly introduces several typical multiplexing algorithms and their problems. Section 3 formally defines our new concept, burst index, and proposes a dynamic multiplexing algorithm (DMA) . Section 4 analyzes the DMA algorithm and derives the maximum delays. Section 5 discusses some limitations of DMA and outlines its extension. In Section 6, the simulation results as compared with GPS methods are presented. Finally, Section 7 concludes the paper.

2. Existent Multiplexing Algorithms and Problems

In this section, we give a brief review of several typical existing algorithms and discuss their problems that motivate our proposal.

One of the simplest multiplexer traffic control strategies is simply sending out traffic according to their arrival time. If cell $c(i,k_i)$ arrives earlier than $c(j,k_j)$, it will be sent out earlier. Here $c(i,k_i)$ is the k_i-*th* cell of connection i, $c(j,k_j)$ is the k_j-*th* cell of connection j. If some cells arrive at the same time, the order of their leaving time is random. Obviously, this mechanism posts a lot of problems in terms of QoS. Miao [9] showed that when the network is busy, the connections with high bandwidth requirement suffer high delays. One of early significant improvement is the Weighted Round Robin (WRR) algorithm [1,2]. WRR assigns a weight w_i for every connection (or session) i, $i=1,\ldots, n$; n is the number of connections. Connection i is served at rate of $\frac{w_i}{\sum_{j=1}^{n} w_j} r$. Thus, service is allocated to connections more fairly and the total delay is shared by all connections fairly on average. One key problem of the WRR is that if the incoming cell of a session just missed the current round, it has to wait for a long time to get the service. This can cause quite significant delay when the number of sessions is large.

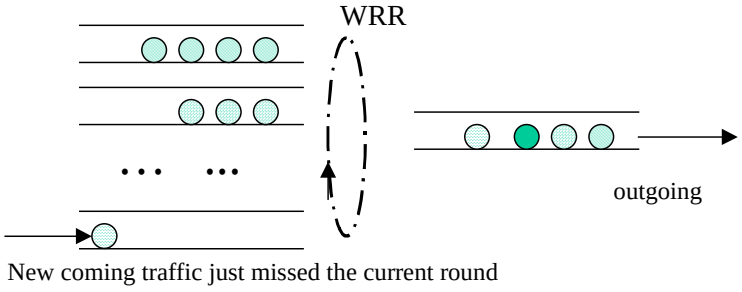


Figure 3 Illustration of Weighted Round Robin multiplexing algorithm

WRR can be improved by introducing a scanner. It scans all the connections continuously. Theoretically, it scans connection i w_i times every $\sum_{j=1}^{n} w_j$ round. If the outgoing link rate is r, then a busy connection i is served at rate of $\frac{w_i}{\sum_{j=1}^{n} w_j} r$ on average. Idle connections are bypassed. This is essentially equivalent to the GPS method.

When there are traffic backlog at connection i, GPS provides service rate at $r_i(t)$, $r_i(t) = r \frac{\phi_i}{\sum_{j \in B(t)} \phi_j}$. Here r is the outgoing rate of the server; ϕ_i is an integer related to the bandwidth share of connection i ; $B(t)$ is the set containing the numbers of the connections which have traffic backlogged at time t. If there is no traffic at connection i, $r_i(t)=0$. GPS can guarantee a busy connection i with outgoing rate $r_i(t)$ of at least $r \frac{\phi_i}{\sum_{j=1}^{n} \phi_j}$. This rate is not affected by the traffic of other connections.

To see how GPS manages the traffic, suppose there are 3 connections and $\phi_1 = 2$, $\phi_2 = 2$, $\phi_3 = 4$ respectively; $r=8$ Mbps . If at time t_0, connection 3 is idle and connection 1 and 2 are busy, the bandwidth allocations of the 3 connections are:

$$r_1(t_0) = r \frac{\phi_1}{\sum_{j \in B(t_0)} \phi_j} = \frac{2}{2+2} r = 4 \text{ Mbps},$$

$$r_2(t_0) = r \frac{\phi_2}{\sum_{j \in B(t_0)} \phi_j} = \frac{2}{2+2} r = 4 \text{ Mbps},$$

$$r_3(t_0) = 0 .$$

Here $B(t_0) = \{1,2\}$. While if at time t_0, all the three connections are busy, $B(t_0) = \{1,2,3\}$. The bandwidth allocation is:

$$r_1(t_0) = r \frac{\phi_1}{\sum_{j \in B(t_0)} \phi_j} = \frac{2}{2+2+4} r = 2 \text{ Mbps},$$

$$r_2(t_0) = r \frac{\phi_2}{\sum_{j \in B(t_0)} \phi_j} = \frac{2}{2+2+4} r = 2 \text{ Mbps},$$

$$r_3(t_0)) = r \frac{\phi_3}{\sum_{j \in B(t_0)} \phi_j} = \frac{4}{2+2+4} r = 4 \text{ Mbps}, .$$

To explain the minimum bandwidth guarantee, let us take connection 3 as an example.

Connection 3 can get at least 4 Mbps if it has backlog. If the backlog is 50 cells, 50cells and 100 cells for connection 1, 2, and 3 respectively, connection 3 can have 4 Mbps bandwidth. While if the backlogged traffics are 500 cells, 1000 cells and 100 cells for connection 1, 2 and 3 respectively, connection 3 still has 4 Mbps. By GPS, the proportion among busy connections is fixed. That is, if connections *i* and *j* are busy, $\frac{r_i}{r_j} = \frac{\varphi_i}{\varphi_j}$.

GPS is to some degree regarded as an ideal multiplexing approach for next-generation ATM switches [4]. Many researchers have devoted much time on the practical issues of GPS. The resulting approximate algorithms are more practical with a little performance degradation. However, we found that the performance can still be improved, especially for bursty traffic [6]. Debanjan Saha [6] pointed out that most of the existent multiplexing algorithms are based on fixed rate service. GPS is somehow a dynamic algorithm. The dynamics come from its work conserving properties. As the above example shows, when connection 3 is idle, the service-rate allocations for connection 1 and 2 are increased to 4 Mbps each. However, the service rate proportion among busy connections is still fixed. This is not affected by the dynamics of bursty traffic. Thus GPS basically belongs to the category of static bandwidth allocation algorithms. When a burst occurs, different connections are affected differently, treating them equal obviously is not an optimal choice.

Kyeong Soo Kim, et. al. [7,8] provided traffic management approaches based on buffer occupancy. The buffers are divided into several levels. The occupancy reaching different level means that different burst level occurs and thus different bandwidth is allocated. These approaches are really dynamic. Though, buffer occupancy level can only give very brief description of the burst status. Additionally, it does not take the delay into account. Suppose two connections both have backlog traffic occupying half of the buffer. The average delays of the backlog traffic from two connections could be 3 time units and 30 time units respectively. Obviously, the server should not treat the two connections equally.

In this paper, we will use burst index as the feedback to optimize the allocation of the processor share dynamically. This can reach the optimal traffic control under a delay objective function. The mechanism is also able to guarantee a minimum bandwidth for each connection, which is the main advantage in GPS.

3. Dynamic Multiplexing Algorithm for Optimal Service Allocation

In this section we introduce a new concept, called burst index, to describe the burst of traffics. Based on the index, a dynamic multiplexing algorithm (DMA) is designed to optimize the traffic. We show that DMA provides better performance for the bursty traffic.

3.1 Burst Index and Objective Function for Traffic Optimization

Delay is one of the most important parameters to network applications. Service providers and network end users negotiate maximum delay and average delay as QoS requirements. Cell lost rate sometime is also related to delays because of dropping mechanism in some network devices. However, in many cases, maximum delay and average delay are not sufficient to represent the dynamic nature of network performance. Suppose connection A is served where one of 100 cells is delayed 600 ms and negligible delay for the rest 99 cells. For connection B, every cell is delayed 200 to 300 ms. Although the maximum delay of connection A is much larger than that of connection B, we usually consider connection A as being better serviced than connection B. The average delay measure also suffers obvious deficiency. Average delay of whole period of a connection is a very rough description. To give more precise description of the service performance, we need to take into account both the number of delayed cells and how long they have been delayed. This notion is captured by a new concept, called the ***burst index*** (**BI**) of connection i, which is formally defined as follows.

Definition 1. *T*he burst index, denoted by $\bar{x}_i(t)$, of connection i, is

$$\bar{x}_i(t) = \sum_{C_i(t) \in S_i(t)} D[C_i(t)], \quad i=1,2,\ldots,n, \qquad (1)$$

where $C_i(t)$ is a delayed cell of connection i, $S_i(t)$ is the set containing all the delayed cells of connection i at time t, $D[C_i(t)]$ is the delay of cell $C_i(t)$, n is the number of connections.

Here we count the delay in time units: $D[C_i(t)]$ is a positive integer. In this paper, a new coming cell is regarded as being delayed 1 time unit even if it will be sent out within that unit time. This is illustrated by figure 4.

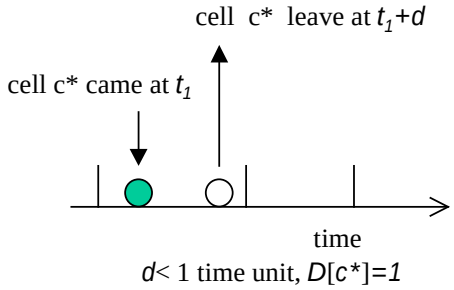


Figure 4 Any buffered cell is regarded as being delayed for 1 time unit.

The burst index dynamically represents both the amount of delayed cells and how long they are delayed. That is, if m_i cells of connection i are delayed d_i, they contribute to $\bar{x}_i(t)$ as $m_i \cdot d_i$: the product of the backlog cell number and their delay. To illustrate this, suppose we have two connections, both with a same 1 Mbps average bandwidth requirement; the outgoing link rate is 2 Mbps, as shown in figure 5.

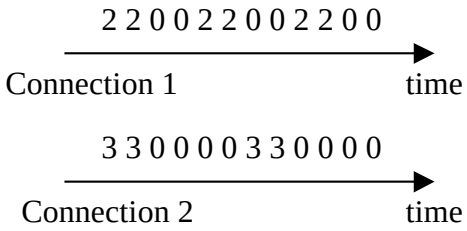


Figure 5 Different connections have different bursitness

In figure 5, the number represents the traffic which come at the corresponding second. For example, 2 2 0 (→) means in the successive 3 seconds, there are 2 M, 2 M, 0 M bits traffic sequence. According to GPS, $\phi_1 = \phi_2$. It can be checked that when the burst comes, connection 2 has more delay than connection 1. This can be illustrated by the imbalance of the burst indexes of the two connections.

As both of the two connections are busy, every connection has 1 Mbps service rate. For instance, when traffic 2 2 0 0 (→) come from connection 1 in sequence, the average cell rate is 1 Mbps. Then after 4 seconds, all cells are sent out. The outgoing cells, backlogs and burst index are shown in the following table.

Time	t+1	t+2	t+3	t+4
New comings	2	2	0	0
Sent outs	1	1	1	1
Backlogs	1	2	1	0
Burst index	2×(1)=2 Two new coming cells. According to the definition of burst index, 2×(1)=2	2×(1)+1×(2)=4 Two new coming cells: 2×(1); One cell came in at previous second, the delay is 2: 1×(2)	2×(2)=4 Two cell came in at previous second, the delay is 2: 2×(2)	1×(3)=3 One cell came in at t+2, the delay is 3: 1×(3)

Similarly, burst index of traffic 3 3 0 0 0 0 (→) in connection 2 is shown in the following table.

Time	t+1	t+2	t+3	t+4	t+5	t+6
New comings	3	3	0	0	0	0
Sent outs	1	1	1	1	1	1
Backlogs	2	4	3	2	1	0
Burst index	3×(1) =3	3×(1)+2×(2) =7	3×(2)+1×(3)=9	3×(3) =9	2×(4) =8	1×(5) =5

Obviously, connection 2 suffers more burst than connection 1 does. This is shown in Figure 6.

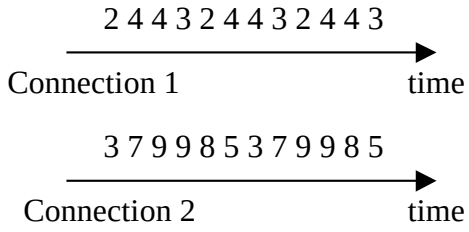


Figure 6 The burst index of the two connections in Figure 5

Connections either with high backlog or with high delay results in high burst index. They both require more bandwidth allocation. Burst index is a synthetical index of backlog and delay. It describes the traffic burstiness more precisely. This is its main advantage over the existing indexes. Objective functions and corresponding optimal algorithms based on burst index can thus have better performance.

To allocate bandwidth according to burst index, we need to compare the burst index among all connections. However, different connections usually have different bandwidth requirement, i.e. different SCR. To make them comparable, we define the ***relative burst index*** (**RBI**) as follows:

Definition 2. The relative burst index of connection i, $x_i(t)$, is

$$x_i(t) = \lambda_i \sum_{C_i(t) \in S_i(t)} D[C_i(t)],\ \lambda_i = \frac{\max_{1 \le j \le n} SCR_j}{SCR_i}, \tag{2}$$

where SCR_i is the SCR of connection i.

Definition 3. The burst index and the relative burst index of the whole system are $\bar{X}(t) = \sum_{i=1}^{n} \bar{x}_i(t)$ and $X(t) = \sum_{i=1}^{n} x_i(t)$ respectively.

With relative burst index defined, the burstiness of different connections can be compared. We define the optimal objective function as

$$\text{Min} \ \underset{1\le i\le n}{Max}\, x_i(t) \ . \tag{3}$$

Equation (3) means that the optimal solution minimizes the maximum burst index of the n connections. I.e., if $\{x_1^*(t), \ldots, x_n^*(t)\}$ is the optimal solution, $x_{i^*}(t) = \underset{1\le i\le n}{Max}\, x_i^*(t)$, for any solution $\{x_1(t), \ldots, x_n(t)\}$, $x_{j^*}(t) = \underset{1\le j\le n}{Max}\, x_j(t)$, it holds that $x_{i^*}(t) \le x_{j^*}(t)$.

Intuitively, a connection should have less service at light burst than at heavy burst. On other words, light burst connections give away bandwidth to connections with heavy burst. As connections get burst randomly (rather than in turn, it is illustrated in Figure 7), the dynamic bandwidth allocation adjustment benefits all connections. The multiplexing algorithm as described below is based on the objective function (3). It tries to balance the burst situations between different connections to reduce the maximum burst index. Therefore it can achieve the optimal over-all performance as well as improves the performance of individual connections.

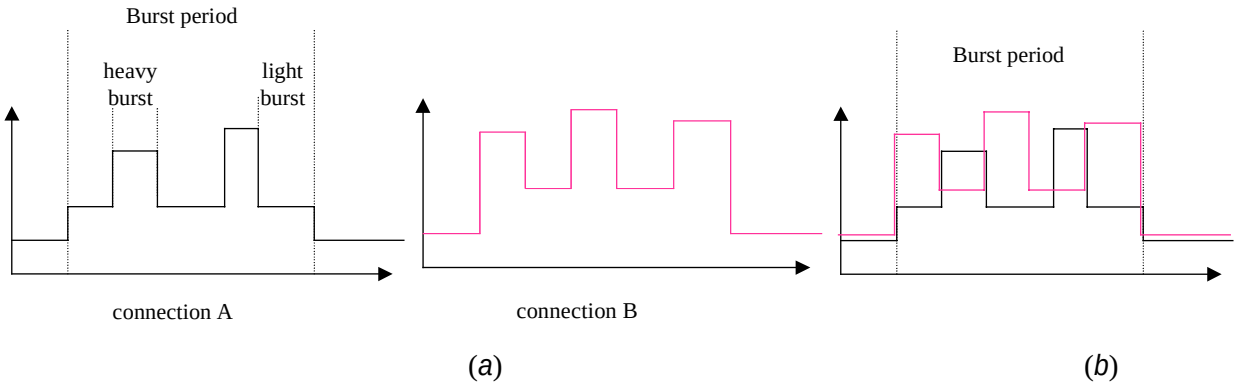


Figure 7 Different connections have burst randomly, i.e. at different time.

In this case, optimal algorithms corresponding to the objective function based on relative burst index can have better performance than GPS, which allocate bandwidth on fixed proportion for connections having bursty inputs.

3.2 Dynamic Multiplexing Algorithm

The Dynamic Multiplexing Algorithm (DMA) is a work-conserving algorithm. That is, the server is kept busy when any of the connection has backlog. Suppose x_i is the relative burst index of connection i. DMA optimizes the objective function of (3) by sending out the backlog in the following way. Suppose $L(i)$, $1 \leqslant i \leqslant n$, $i=1, \cdots, n$ are integers such that

$$x_{L(1)} > x_{L(2)} > \cdots > x_{L(n)} .$$

DMA decides traffic outgoing rate according to $x_{L(i)}$ $i=1, \cdots, n$. Suppose r is the outgoing rate of

the multiplexer, r_i is the allocated rate for connection i. The outgoing device sends traffic out at rate r_i for connection i, while at the same time, DMA decides the r_i , $i=1, \cdots, n$. Denote δ as the time unit for delay computation, i.e. all the cell delays are integer numbers of δ. δ can be set according to the computation capacity of the multiplexer. The minimum value of δ is one slot. For example, for 155 Mbps outgoing rate, $\delta \geqslant 1\ slot = 2.7355\ \mu s$. Assume that the algorithm starts at time t_1.

Dynamic Multiplexing Algorithm (DMA)

$T = t_1 + \delta$
Calculate $x_i(t)$, $i=1, \cdots, n$
Assign $L(i)$, $i=1, \cdots, n$
For $i=1$ to n
  If no new traffic comes and $t<T$
    While $x_{L(i)}(t) > x_{L(i+1)}(t)$

$$\begin{cases} r_{L(j)} = \dfrac{SCR_i}{\sum_{j=1}^{i} SCR_j} r \dfrac{1}{i} & 1 \le j \le i \\ r_{L(j)} = 0 & i < j \le n \end{cases} \qquad (*)$$

    end while
  else i.e. there is new traffic come or $t \geq T$
    recalculate $L(i)$
    $i=0$
    $T=t+\delta$
  Endif
Next i

Explanation:

1. When $i=1$, all the bandwidth is assigned to connection $L(1)$ ($r_{L(1)}=r$) until $x_{L(1)} = x_{L(2)}$, or a new cell comes, or t=T. In case that a new cell comes or t=T, $L(i)$ $1 \le i \le n$ should be recalculated as $x_i(i=1, \cdots, n)$ are changed. In case that $x_{L(1)} = x_{L(2)}$, the bandwidth is reallocated and connection $L(1)$ and $L(2)$ share the multiplexer: $r_{L(1)} = \dfrac{SCR_1}{SCR_1 + SCR_2} r$, $r_{L(2)} = \dfrac{SCR_2}{SCR_1 + SCR_2} r$ and so on.
2. The computational complexity of DMA is very small. If δ=1 slot, at most one cell can be sent out. Suppose the cell belongs to connection i, then

$$x_i(t+\delta) = x_i(t) + \lambda_i [N[S_i(t)] - 1 - D_i(t)] , \qquad (4)$$

$$x_j(t+\delta) = x_j(t) + \lambda_j [N[S_j(t)]] \quad j \neq i , \qquad (5)$$

   where $N[S_j(t)]$ $j=1, \cdots, n$ is the number of cells $S_j(t)$ contains, $S_j(t)$ is the same as in definition 1. $D_i(t)$ is the delay of the cell which suffers the longest delay among backlogs of

connection i. As x_j does not change much, recalculating $L(j)$ does not involve much calculation either. **Furthermore**, it can be carried out **in parallel**.

3. There is no worry for some connections, which burst continuously, will block other connections. This is due to the following conditions:

   (1) Connections burst randomly.

   (2) According to the traffic parameters, the maximum time for a connection to have traffic at PCR is $\frac{MBS}{PCR}$. After that, it reduces to at most SCR. The earliest next such burst can only occur after $\frac{MBS}{SCR}-\frac{MBS}{PCR}$. This the shortest idle period between two maximum bursts.

   (3) For any time period, △T, the incoming traffic of a connection is bounded by $\min\left(\left\lceil 1+\Delta T\cdot SCR+(MBS-1)\cdot(1-\frac{SCR}{PCR})\right\rceil, \left\lceil 1+\Delta T\cdot PCR\right\rceil\right)$, where $\lceil\bullet\rceil$ is the integer part of $\bullet$. When △T is large, it converges to △T SCR.

   (4) The burst index of a blocked connection increases rapidly if it has consistent incoming traffic. Suppose a connection with average traffic input is blocked by n time units, the burst index is

   $$(1+2+\cdots+n)\ SCR = 0.5\ n\ (n+1)\ SCR\ .$$

   The order is $O(n^2)$. When traffic is managed according only to backlogs or delays, the order is $O(n)$. Therefore, a badly blocked connection can get service earlier by DMA than other algorithms.

3.3 An Example of DMA

In this section, we use an example to illustrate DMA. Suppose we have 3 connections with the same SCR. This means that burst indexes equal to the relative burst indexes, $\lambda_1=\lambda_2=\lambda_3=1$. The multiplexer is able to send 4 cells in one time unit. At time t_0, suppose the backlogged traffic are: {4(2)} for connection 1, {5(2), 2(3)} for connection 2 and {2(2),1(3)} for connection 3. Here $m(s)$ means m cells that have been backlogged s units. For instance, 2(3) means 2 cells were backlogged 3 units. From t_0 to t_0+6, the incoming traffic is shown in the following table:

	t_0	t_0+1	t_0+2	t_0+3	t_0+4	t_0+5	t_0+6
Connection 1	3	0	0	0	0	2	0
Connection 2	2	0	0	0	0	0	0
Connection 3	6	0	0	0	0	1	0

We use $i[k]$ to denote that k cells are sent out for connection i. t^+ is the time just after t while t^- is the time just before t. Then the DMA procedure is :

At time t_0

1	2	3	4	5	6	A	B
	Backlogged traffic at time t0+	Out going traffic		2[2]	2[1] 3[1]	Backlogged traffic at time (t0+1)-	
Connection 1	3(1), 4(2)	Burst index of connection 1	11	11	11	3(1), 4(2)	Connection 1
Connection 2	2(1), 5(2), 2(3)	Burst index of connection 2	18	12	10	2(1), 4(2)	Connection 2
Connection 3	6(1),2(2),1(3)	Burst index of connection 3	13	13	10	6(1),2(2)	Connection 3

The above table at time t_0 can be read as follows:

- **Column 2** shows the backlogged traffic at time t_0 . We can see that for connection 1, the backlogged traffic are 4(2) and 3(1). 4(2) means 4 cells are backlogged 2 units. This is from the initial assumption. 3(1) indicates the 3 new coming cells. According to the definition of burst index, their delays are 1 unit. Similarly, for connection 2, the backlogged traffic is {2(1), 5(2), 2(3)}; for connection 3, the backlogged traffic is {6(1), 2(2), 1(3)}.
- **Column 4** holds the bust index of the three connections when DMA starts. For connection 1, its $3\times(1)+4\times(2)=11$. Similarly, $x_2= 2\times(1)+5\times(2) +2\times(3)=18$; $x_3= 6\times(1)+2\times(2) +1\times(3)=13$.
- **Column 5** is a middle step of the DMA algorithm. We can see that L(1)=2, L(2)=3, L(3)=1. According to DMA, all the bandwidth should be used to send backlogs of connection L(1)=2 until $x_{L(1)} \le x_{L(2)}$. This happens when 2 cells of connection 2 are sent out. Details are as follows: After one cell is sent out, which should be the one with longest delay, the backlogs is {2(1), 5(2), 1(3)}. The burst index of connection 2 is now $x_2= 2\times(1)+5\times(2) +1\times(3)=15$. The while condition of DMA is not met and another cell of connection 2 is sent out. Then the backlogs is {2(1), 5(2)} and $x_2= 2\times(1)+5\times(2)=12$. The condition of $x_{L(1)} \le x_{L(2)}$ is now met and bandwidth is reallocated. In this round, 2 cells of connection 2 are sent out. Therefore the first row of column 5 is 2[2].
- **Column 6** shows the that connection L(1)=2 and connection L(2)=3 share the bandwidth, until $x_{L(2)} \le x_{L(3)}$. Now both connection 2 and connection 3 send out one cell. Their remaining backlogs are {2(1), 4(2)} and {6(1), 2(2)}. Their burst indexes are all reduced to 10. The first row of column 6 shows two cells are sent out, one for connection 2 and one for connection 3. That is 2[1] and 3[1].

- **Column A** is the final status of current time unit. As 4 cells are sent out, this unit is over. Compare this column with the column 2 of next table at time t_0+1, we can see after one unit, the delay of every backlogged cell has to be increased by 1 and the burst indexes have to be modified.

All the rest tables can be read similarly and we simply put them up.

At time t_0+1

1	2	3	4	5	6	A	B
	Backlogged traffic at time $(t_0+1)^+$	Out going traffic		1[1] 3[1]	1[1] 2[1]	Backlogged traffic at time $(t_0+2)^-$	
Connection 1	3(2), 4(3)	Burst index of connection 1	18	15	12	3(2), 2(3)	Connection 1
Connection 2	2(2), 4(3)	Burst index of connection 2	16	16	13	2(2), 3(3)	Connection 2
Connection 3	6(2),2(3)	Burst index of connection 3	18	15	15	6(2),1(3)	Connection 3

Ideally, the three connections should share the 2 cells service at the gray part, i.e., 1[2/3], 2[2/3], 3[2/3] . As we cannot send 2/3 cell, the service is allocated as 1[1], 2[1].

At time t_0+2

	Backlogged traffic at time $(t_0+2)^+$	Out going traffic		3[1]	2[1] 3[1]	1[1]	Backlogged traffic at time $(t_0+3)^-$	
Connection 1	3(3), 2(4)	Burst index of connection 1	17	17	17	13	3(3), 1(4)	Connection 1
Connection 2	2(3), 3(4)	Burst index of connection 2	18	18	14	14	2(3), 2(4)	Connection 2
Connection 3	6(3),1(4)	Burst index of connection 3	22	18	15	15	5(3)	Connection 3

Similarly, we cannot assign 1/3 cell service to a connection. Thus the gray part is 1[1]. In practice, the multiplexer can assign service to connections randomly in such situation.

At time t_0+3

	Backlogged traffic at time $(t_0+3)^+$	Out going traffic		3[1]	2[1] 3[1]	1[1]	Backlogged traffic at time $(t_0+4)^-$	
Connection 1	3(4), 1(5)	Burst index of connection 1	17	17	17	12	3(4)	Connection 1
Connection 2	2(4), 2(5)	Burst index of connection 2	18	18	13	13	2(4), 1(5)	Connection 2
Connection 3	5(4)	Burst index of connection 3	20	16	12	12	3(4)	Connection 3

At time t_0+4

	Backlogged traffic at time $(t_0+4)^+$	Out going traffic		2[1]	1[1],2[1] 3[1]		Backlogged traffic at time $(t_0+5)^-$	
Connection 1	3(5)	Burst index of connection 1	15	15	10		2(5)	Connection 1
Connection 2	2(5), 1(6)	Burst index of connection 2	16	10	5		1(5)	Connection 2
Connection 3	3(5)	Burst index of connection 3	15	15	10		2(5)	Connection 3

At time t_0+5

	Backlogged traffic at time $(t_0+5)^+$	Out going traffic		1[1]	1[1] 3[1]	1[1]	Backlogged traffic at time $(t_0+6)^-$	
Connection 1	2(1),2(6)	Burst index of connection 1	14	8	2	1	1(1)	Connection 1
Connection 2	1(6)	Burst index of connection 2	6	6	6	6	1(6)	Connection 2
Connection 3	1(1),2(6)	Burst index of connection 3	13	13	7	7	1(1),1(6)	Connection 3

At time t_0+6

	Backlogged traffic at time $(t_0+6)^+$	Out going traffic		3[1]	2[1] 3[1]	1[1]	Backlogged traffic at time $(t_0+7)^-$	
Connection 1	1(2)	Burst index of connection 1	2	2	2	0	0	Connection 1
Connection 2	1(7)	Burst index of connection 2	7	7	0	0	0	Connection 2
Connection 3	1(2),1(7)	Burst index of connection 3	9	2	0	0	0	Connection 3

3.4 Optimization Analysis of DMA

It is intuitive that DMA can optimize the objective function (3). Suppose there is no new incoming traffic from t_0 to $t_0+\delta_0$, $\delta_0 < \delta$, δ is the time unit. $x_i(t_0)$ and $x_i(t_0+\delta_0)$ are the burst indexes of connection i at time t_0 and $t_0+\delta_0$. Define $F(x_i(t_0), x_i(t_0+\delta_0))$ as the decrease of traffic backlog at connection i, $i=1,\cdots,n$. Then we can prove the optimization of DMA in Theorem 1. For the sake of simplicity, we assume that the burst index equals to the relative burst index. That is to say, the SCRs of all the connections are the same. It can be checked that the proof can be extended to general cases without essential difficulties.

Theorem 1. If there is no new incoming traffic in $[t_0, t_0+\delta_0]$, $\delta_0 < \delta$, δ is the time unit, DMA is the optimal solution for objective function (3).

Proof. Suppose, by contradiction, there is another algorithm that can have smaller maximum

burst index at $t_0+\delta_0$. We denote the algorithm as SA. To denote the burst indexes of DMA and SA separately, we use superscript DMA and SA.

According to DMA, there exist k, $1 \leqslant k \leqslant n$,

$$x_{L(1)}^{DMA}(t_0+\delta_0) = \cdots = x_{L(k)}^{DMA}(t_0+\delta_0) > x_{L(k+1)}^{DMA}(t_0+\delta_0),$$

and

$$x_{L(1)}^{DMA}(t_0+\delta_0) > x_{L(j)}^{DMA}(t_0+\delta_0) \qquad k+1 \le j \le n \ .$$

Note that connection $L(k+1), \cdots, L(n)$ do not get service between t_0 and $t_0+\delta_0$. We have

$$\sum_{i=1}^{k} F(x_{L(i)}(t_0), x_{L(i)}^{DMA}(t_0+\delta_0)) = \delta_0 \cdot r \ .$$

As SA has smaller maximum burst index than that of DMA,

$$x_{L(i)}^{SA}(t_0+\delta_0) < x_{L(i)}^{DMA}(t_0+\delta_0) \ , \qquad 1 \le i \le k \ .$$

Then

$$\sum_{i=1}^{k} F(x_{L(i)}(t_0), x_{L(i)}^{SA}(t_0+\delta_0)) > \sum_{i=1}^{k} F(x_{L(i)}(t_0), x_{L(i)}^{DMA}(t_0+\delta_0)) = \delta_0 \cdot r \ .$$

This means that SA exceeded the outgoing capacity r. It is not practical. The contradiction shows no other algorithms can have smaller maximum burst index. DMA is the optimal algorithm for objective function (3).

□

4 Further Analysis of Algorithm DMA

This section presents further analysis on DMA to derive the maximum relative burst index and maximum delay. Although in some cases it is not easy to be proven [3], the worst situation is usually considered when all the connections start the longest burst together at the same time. We first assume the connections have the same traffic parameters: PCR, SCR, MBS or T_p, T_s, τ_s [1], derive the corresponding maximum relative burst index and delay. Then we expand the result to connections with different traffic parameters.

Theorem 2. Suppose the traffic parameters of n input connections are all PCR, SCR, MBS and the outgoing rate is r. If the n connections start their longest burst at the same time k_1, the maximum delay T_1^* occurs at k_1^*, then $k_1^* = k_1 + \frac{n \cdot MBS}{r}$, $T_1^* = \frac{n \cdot MBS}{r} - \frac{MBS}{PCR}$.

[1] $T_s = \frac{1}{SCR}, T_P = \frac{1}{PCR}, MBS = \left\lfloor 1 + \frac{\tau_s}{T_s - T_P} \right\rfloor$, here $\lfloor \bullet \rfloor$ is the integer part of $\bullet$.

Proof. As the n connections have same traffic parameters, $\lambda_1 = \lambda_2 = \cdots = \lambda_n = 1$[2].

According to the traffic parameters, the input burst lasts until $k_1 + \frac{MBS}{PCR}$. After that, the input rate of every connection gets back to SCR (or less) and the cell backlog start to decrease. They would be cleared before $k_1 + \frac{(n \cdot PCR - r)}{r - n \cdot SCR}\frac{MBS}{PCR}$.

However, the maximum delay of the backlog keeps on increasing after $k_1 + \frac{MBS}{PCR}$. This is because there are still backlogs which came in at rate of PCR. The maximum delay stops increasing when the traffic came at the rate of PCR are all cleared. It needs $\frac{(n \cdot PCR - r)\frac{MBS}{PCR}}{r}$. Thus

$$k_1^* = k_1 + \frac{MBS}{PCR} + \frac{(n \cdot PCR - r)\frac{MBS}{PCR}}{r} = k_1 + \frac{n \cdot MBS}{r}.$$

After k_1^*, all the backlogged cells came at rate of SCR or less. This means that every time unit the multiplexer can clear the backlog that accumulated in no less than one unit time. Therefore, the maximum delay decreases. This means at time k_1^*, the cells came at time $k_1 + \frac{MBS}{PCR}$ have the maximum delays. The delay T_1^* is

$$k_1^* - (k_1 + \frac{MBS}{PCR}) = \frac{(n \cdot PCR - r)\frac{MBS}{PCR}}{r} = \frac{n \cdot MBS}{r} - \frac{MBS}{PCR}.$$

Theorem 3. Suppose the traffic parameters of n input connections are all PCR, SCR, MBS and the outgoing rate is r. If the n connections start the longest burst at the same time k_1, the maximum burst index X^* occurs at k_2^*, then

$$k_2^* = k_1 + \frac{MBS}{PCR} + \Delta_{k_3}, \quad X^* = \frac{\Delta_{k_2} + 2 \cdot \Delta_{k_3} + 1}{2} \cdot \Delta_{k_2} \cdot n \cdot PCR + \frac{\Delta_{k_3} + 1}{2} \cdot \Delta_{k_3} \cdot n \cdot SCR,$$

[2] In this case, burst index equals to relative burst index.

where, $\Delta_{k_3} = \dfrac{\dfrac{MBS}{PCR}\dfrac{(n \cdot PCR - r)^2}{n \cdot PCR} - r}{\dfrac{r(n \cdot PCR - r)}{n \cdot PCR} - (n \cdot SCR - r)}$, $\Delta_{k_2} = \dfrac{\dfrac{MBS}{PCR}(n \cdot PCR - r) - r \cdot \Delta_{k_3}}{n \cdot PCR}$.

Proof. Suppose the maximum relative burst index is reached at the time k_2*. k_2* is when the index increase, which is because of the further delay of the backlog, is reduced by the effects of the ending of the burst and the clearing of traffic. As illustrated in Figure 8, suppose at time k_2* , the backlogged traffic is S, the maximum delay is $\Delta_{k_2} + \Delta_{k_3}$.

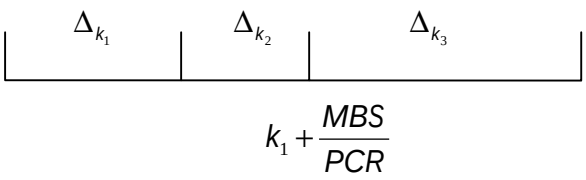


Figure 8 Key points of the variation of the burst index and traffic backlog.

Then, at time k_2*+1, we have[3]

$$(S\text{-}r) - r \cdot (\Delta_{k_2} + \Delta_{k_3}) \le 0 . \quad (6)$$

As maximum delay at k_2* is $\Delta_{k_2} + \Delta_{k_3}$, the traffic backlog at time k_2*, *i.e. S*, equals to the income traffic in [k_1 , k_2*] minus the traffic sent out in the same interval:

$$S = \frac{MBS}{PCR} \cdot n \cdot PCR + n \cdot SCR \cdot \Delta_{k_3} - r \cdot (\frac{MBS}{PCR} + \Delta_{k_3}) \quad (7)$$

.

Traffic backlogged in [k_1 , $k_1 + \Delta_{k_1}$] are cleared by the multiplexer in [k_2*-($\Delta_{k_2} + \Delta_{k_3}$), k_2*]:

$$(\frac{MBS}{PCR} - \Delta_{k_2}) \cdot (n \cdot PCR - r) = r \cdot (\Delta_{k_2} + \Delta_{k_3}) . \quad (8)$$

$$\Delta_{k_2} = \frac{\frac{MBS}{PCR}(n \cdot PCR - r) - r \cdot \Delta_{k_3}}{n \cdot PCR} . \quad (9)$$

Substitute (7) (9) in (6),

$$\frac{MBS}{PCR} \cdot n \cdot PCR + n \cdot SCR \cdot \Delta_{k_3} - r \cdot (\frac{MBS}{PCR} + \Delta_{k_3}) \text{-r} \le (\frac{MBS}{PCR} - \Delta_{k_2}) \cdot (n \cdot PCR - r)$$

$$\le \qquad (\frac{MBS}{PCR} \text{-}$$

[3] We omit the possible 1 slot error in this paper, which will not cause essential difference of the result. This can help to keep the result neat. Accurately, the inequality (6) should be

$$(S\text{-}r)\text{-}\{ r_1 \cdot (\Delta_{k_2} + \Delta_{k_3}) + (r\text{-} r_1) \cdot (\Delta_{k_2} + \Delta_{k_3} \text{-}1)\} \le 0 ,$$

$$\frac{\frac{MBS}{PCR}(n \cdot PCR - r) - r \cdot \Delta_{k_3}}{n \cdot PCR}) \cdot (n \cdot PCR - r) \ , \tag{10}$$

$$\Delta_{k_3} \geq \frac{\frac{MBS}{PCR}\frac{(n \cdot PCR - r)^2}{n \cdot PCR} - r}{\frac{r(n \cdot PCR - r)}{n \cdot PCR} - (n \cdot SCR - r)} \ . \tag{11}$$

Then we have

$$k_2^* = k_1 + \frac{MBS}{PCR} + \Delta_{k_3} \ , \ \Delta_{k_3} = \frac{\frac{MBS}{PCR}\frac{(n \cdot PCR - r)^2}{n \cdot PCR} - r}{\frac{r(n \cdot PCR - r)}{n \cdot PCR} - (n \cdot SCR - r)} . \tag{12}$$

As illustrated in Figure 8, at time k_2^*, the traffic came in during interval $[k_1 + \frac{MBS}{PCR} - \Delta_{k_2}$, $k_1 + \frac{MBS}{PCR} + \Delta_{k_3}]$ are all backlogged. Thus the relative burst index is:

$$X^* = (\Delta_{k_2} + \Delta_{k_3} + \Delta_{k_2} + \Delta_{k_3} - 1 + \cdots + \Delta_{k_3} + 1) \ n \cdot PCR \ + (\Delta_{k_3} + \cdots + 1) \ n \cdot SCR$$

$$= \frac{\Delta_{k_2} + 2 \cdot \Delta_{k_3} + 1}{2} \cdot \Delta_{k_2} \cdot n \cdot PCR + \frac{\Delta_{k_3} + 1}{2} \cdot \Delta_{k_3} \cdot n \cdot SCR \tag{13}$$

In the following analysis, we would like to show some details of the backlogged traffic, which helps the understanding of above results. Let $\triangle = \frac{r}{n \cdot PCR - r}$. Recall that the number of the traffic backlog reaches the maximum at $k_1 + \frac{MBS}{PCR}$.

From k_1 to $k_1 + \triangle$, i.e. $k \in [k_1, k_1 + \triangle]$, the relative burst index of the traffic backlog is

$$x_i(k) = PCR \cdot (k - k_1) - r \cdot (k - k_1) / n \quad i = 1, \cdots, n \ .$$

From $k_1 + \triangle$ to $k_1 + 2\triangle$, i.e. $k \in [k_1 + \triangle, k_1 + 2\triangle]$, the total backlog is

$$n \cdot PCR \cdot (k - k_1) - r \cdot (k - k_1) \ .$$

As

$$n \cdot PCR \cdot (k - k_1) - r \cdot (k - k_1) > n \cdot PCR \cdot \triangle - r \cdot \triangle > r \ ,$$

some of the traffic are delayed by 2 time units. All together, they are

---

where r_1 is delayed $\Delta_{k_2} + \Delta_{k_3}$ and $r - r_1$ is delayed $\Delta_{k_2} + \Delta_{k_3} - 1$.

$$n \cdot PCR \cdot (k- k_1-\triangle) - r \cdot (k- k_1-\triangle) .$$

Thus the burst index of connection i is

$$x_i(k)= PCR \cdot (k- k_1) - r \cdot (k- k_1)/n + PCR \cdot (k- k_1-\triangle) - r \cdot (k- k_1-\triangle)/n \quad i=1,\cdots,n.$$

From $k_1 +\triangle$ to $k_1+2\triangle$ and so on, the process can be worked on similarly.

Suppose the traffic of every connection is between MCR (minimum cell rate) and PCR. As the algorithm tries to eliminate the differences between relative burst indexes, the traffic with the larger relative burst index has higher priority to be sent out. Therefore, the connection with small cell backlog will have larger delay. The worst delay situation occurs when a connection remains at MCR, while $n-1$ other connections start longest burst at the same time.

Theorem 4. Suppose n connections have traffic parameters of PCR, SCR, MBS and r is the outgoing rate. If connection i, $i=1,\cdots,n-1$, burst at time k_1, and the source of connection n transmits at MCR. The maximum delay of connection n is bounded by $\left\lceil \frac{1}{2}\cdot\left(\sqrt{\frac{8x^*+r_n^*}{r_n^*}}-1\right)\right\rceil+1$, where $\lceil\bullet\rceil$ is the integer part of $\bullet$, x^* equals to the X^* in (13), $r_n^*=min\ \{MCR,\ \frac{r}{n-2}-\frac{n-1}{n-2}(PCR-MCR)\}$.

Proof. As the overall input traffic are smaller than all the n connections start to have burst at k_1 , the maximum relative burst index x^* can not exceed the maximum burst in Theorem 3, i.e.

$$x^* < X^* .$$

If $MCR<PCR-\frac{r}{n-1}$, all the traffic of connection n are backlogged;

If $MCR>PCR-\frac{r}{n-1}$, the traffic of connection n can be sent out at the rate

of $\frac{r}{n-2}-\frac{n-1}{n-2}(PCR-MCR)$.

Let $r_n^*=min\{MCR,\ \frac{r}{n-2}-\frac{n-1}{n-2}(PCR-MCR)\}$. Suppose the traffic of connection n have to be delayed k^* to reach x^*, then

$$(1+2+\cdots+k^*)\, r_n^* \geqslant x^* ,$$

$$k^* \leqslant \left\lceil \frac{1}{2}\cdot\left(\sqrt{\frac{8x^*+r_n^*}{r_n^*}}-1\right)\right\rceil+1 \leqslant \left\lceil \frac{1}{2}\cdot\left(\sqrt{\frac{8X^*+r_n^*}{r_n^*}}-1\right)\right\rceil+1 .$$

I.e. the maximum delay of connection n is also bounded by k^*. $[\bullet]$ is the integer part of $\bullet$.

□

As we have noted that Theorem 4 is the worst situation for traffic of a connection being blocked, k^* is the maximum delay DMA can have.

The results of Theorem 2, 3 and 4 are obtained under the assumption that the traffic parameters of n connections are the same. When the traffic parameters of connections are different, the result and the proof are similar, though more complicated in presentation. We give the result in the following theorem and omit the proof.

Denote the traffic parameters of the connection i as PCR_i, SCR_i, MCR_i, MBS_i, $i=1,\cdots,n$. Suppose the traffic come as discrete event based on the multiplexer time unit, i.e., if the algorithm starts at time k_1, the traffic can only come in and be sent out at k_1+1, k_1+2, $\cdots$. We omit all the 1 time unit error that might occur. Let

$$T(i)=\frac{MBS_i}{PCR_i},\quad B_i(k)=min\{k_1+T(i),k\},$$

$$N_i(k)=min\left\{\left\lceil 1+(k-k_1)\cdot SCR_i+(MBS_i-1)(1-\frac{SCR_i}{PCR_i})\right\rceil,\ [1+(k-k_1)PCR_i]\right\},$$

$[\bullet]$ is the integer part of $\bullet$,

$$N(k)=\sum_{i=1}^{n}N_i(k),$$

$$f_i(k)=PCR_i\,(k-k_1+k-k_1-1+\cdots+k-B_i)+SCR_i\,(k-B_i-1+k-B_i-2+\cdots+1),$$

$$R(k)=(k-k_1+k-k_1-1+\cdots+1)\,r,$$

$$F(k)=\sum_{i=1}^{n}f_i(k)-R(k),$$

$$T=min\{t\mid t>k_1,\ N(T-k_1)=r(T-k_1)\}.$$

Suppose $F^*=F(k^*)=\underset{k>k_1}{Max}F(k)$. Note that $k^*<T$. The relative burst index reaches the maximum at k^*. The multiplexer can clear the backlog of the current burst before T.

Theorem 5. Suppose n connection start to burst at the same time k_1. The maximum relative

burst index under the DMA algorithm is F*. The burst will be cleared out before T. The maximum delay of connection i is $\left\lceil \frac{1}{2} \cdot \left(\sqrt{\frac{8F^* \cdot SCR_i + MCR_i \cdot \sum_{i=1}^{n} SCR_i}{MCR_i \cdot \sum_{i=1}^{n} SCR_i}} - 1 \right) \right\rceil + 1$, $\lceil \bullet \rceil$ is the integer part of $\bullet$.

To help the understanding of the result of Theorem 5, we can roughly explain in the way of the balance of relative burst index. $f_i(k)$ is how much connection i can contribute to the increase of the burst index from k_1 to k. $R(k)$ is how much the multiplexer can clear the accumulation of the relative burst index. Then, F* is the maximum relative burst index of all connections. As long as DMA can be carried out fast enough, we can assume that all the connections tend to have same relative burst index. Suppose:

$$\frac{x_i}{SCR_i} = \lambda \ .$$

As

$$\sum_{i=1}^{n} x_i = F^* \ ,$$

$$\lambda \sum_{i=1}^{n} SCR_i = F^* \ .$$

We have

$$\lambda = \frac{F^*}{\sum_{i=1}^{n} SCR_i} \ .$$

Then

$$x_i = \frac{SCR_i}{\sum_{j=1}^{n} SCR_j} F^* \qquad i = 1, \cdots, n \ .$$

The maximum delay of connection i is :

$$\left\lceil \frac{1}{2} \cdot \left(\sqrt{\frac{8F^* \cdot SCR_i + MCR_i \cdot \sum_{i=1}^{n} SCR_i}{MCR_i \cdot \sum_{i=1}^{n} SCR_i}} - 1 \right) \right\rceil + 1 \ .$$

Here $\lceil \bullet \rceil$ is the integer part of $\bullet$. Obviously, the maximum delay occurs when a connection reaches the maximum relative burst index with the minimum cell rate.

5. Further Extension and Practical Issues of DMA

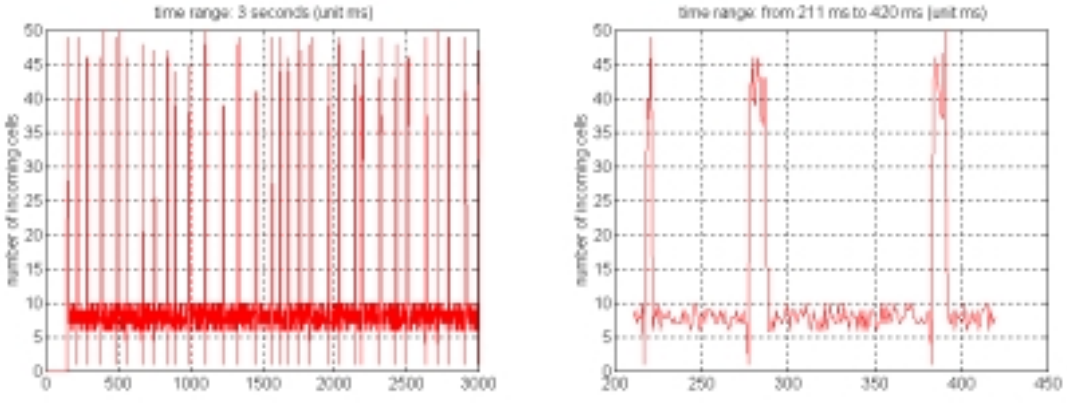
One of the basic advantages of GPS is that the minimum service rate of a connection $\left(i.e. \ r \frac{\phi_i}{\sum_{j=1}^{n} \phi_j} \right)$ is not affected by the burstiness of other connections. This is based on the

feature that GPS regards burst equally. For example, if connection 1 and connection 2 are both bursting, the service rate ratio of them is fixed, i.e.: $\frac{r_1}{r_2} = \frac{\phi_1}{\phi_2}$. As we have illustrated in Figure 7, the bursts of different connections occur randomly. DMA balances these bursts and thus improves the overall performance as well as the individual connection performance. However, DMA cannot guarantee the minimum service rate. A connection may not have service for a while if it has very little backlog (small relative burst index). Though, this traffic will only be backlogged for very short time before its burst index becomes big. This situation is discussed in Section 3.2. Nevertheless, if the property of minimum service rate guarantee is really valuable for the multiplexer, such a mechanism can be added into DMA easily. We can require that every connection can have the minimum service rate of $\frac{MCR_i}{\sum_{j=1}^{n} MCR_j}$ if it has traffic backlog. The remaining service rate of the multiplexer is still allocated according to the burst index. This does not introduce much difficulty to the above algorithm and we omit it here.

6 Simulations

This section presents computer simulations of the DMA multiplexing algorithm. To illustrate the performance improvement, our results are compared with those of GPS. GPS has been somehow regarded as an ideal algorithm [4,5] to be compared with. Lots of works have been done to search for faster algorithm without much performance degradation as compared to those of GPS. Here, we show that the performance of GPS can be improved.

In our simulations, the outgoing rate *r=50 cells/ms , i.e. 20 Mbps* approximately. The multiplexer has four VBR connections. Their traffic parameters are PCR_i*=50 cells/ms* (*20Mbps*), SCR_i*=11 cells/ms* (*4.66Mbps*), MCR_i*=4 cells/ms* (*1.7Mbps*) , $i=1,\cdots,4$. The longest burst at the rate of PCR is *12 ms.* As PCR_i/MCR_i*>10,* these connections have characteristics of being heavily bursty. Figure 9 shows 3 seconds of the VBR stream that is generated randomly.



(*a*) *3 seconds traffic* (*b*) *traffic from 0.2s to 0.45s*

Figure 9 CeII stream of a VBR connection with parameters PCR=50 ceIIs/ms SCR=11 ceIIs/ms

In our simulations shown below, we adopt the revised DMA that guarantees the minimum cell rate of MCR_i. Table 6.1 shows the maximum delay of DMA and GPS algorithm respectively. We can see that DMA has smaller maximum delays. This means DMA has improved the overall performance.

Connections	1	2	3	4
DMA	17	17	14	17
GPS	21	18	19	27

Table 6.1 maximum delay of each connection

Based on a permitable maximum delay for each connections as 15, table 6.2 shows the number of cells to be dropped. It shows that the drops of GPS is much more than that of DMA.

Connections	1	2	3	4
DMA	47	49	0	22
GPS	230	109	168	275

Table 6.2 Dropped ceIIs of each connection, maximum delay tolerance: 15

Take the maximum delay of GPS, that is 27, as 100%, delays larger than 15 are in the scope of 45% from the maximum delay. We call this scope 45% maximum delay scope. It is illustrated in the following figure.

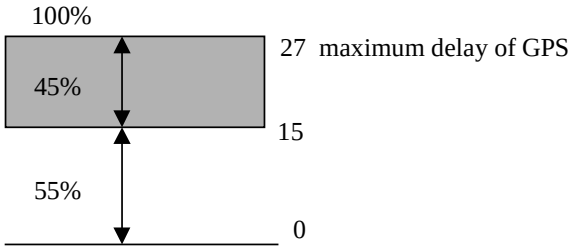
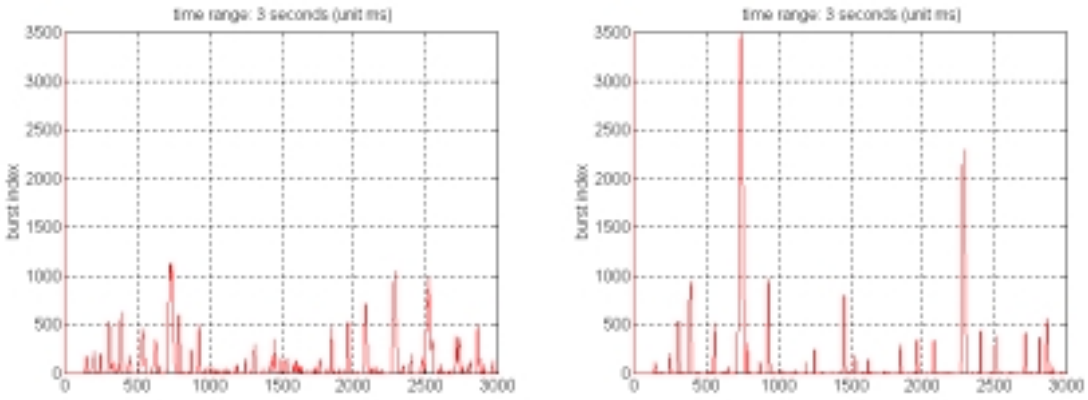


Figure 10. 45% maximum delay scope

From table 6.2 we can work out that the four connections under GPS algorithm have $69.7\times10^{-2}\%$, $33.03\times10^{-2}\%$, $3350.91\times10^{-2}\%$ and $83.3\times10^{-2}\%$ cells fell in the 45% maximum delay scope. While the same four connections under DMA algorithm have only $14.24\times10^{-2}\%$, $14.85\times10^{-2}\%$,0, $6.67\times10^{-2}\%$ cells respectively fell in the 45% maximum delay scope of GPS. The performance improvement is obvious. DMA is more capable to smooth out the bursty traffic and balance the delays among cells.

The advantages of DMA can also be seen via burst indexes. From table 6.1 and 6.2 we observe

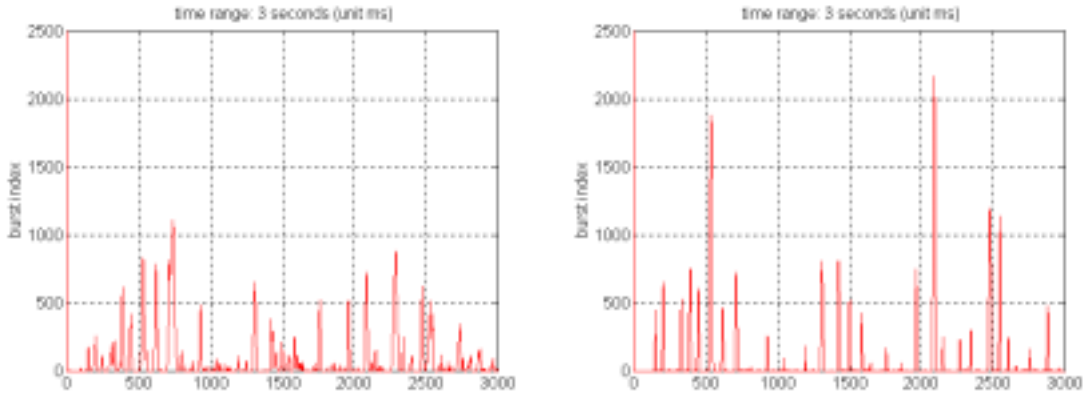
that connection 4 has the largest performance improvement. Figure 11 shows record of the burst index of the connection 4 under GPS and DMA respectively. DMA reduced the relative burst index or the burstiness.



(a) (b)

Figure 11 Burst index of connection 4 under multiplexing algorithm of DMA (a) *and GPS*(*b*).

GPS and DMA have the closest performance in connection 2. Figure 12 (a) and (b) shows the improvement of DMA is still significant.



(a) (b)

Figure12 Burst index of connection 2 under multiplexing algorithm of DMA (a) *and GPS*(*b*).

7. Conclusion

In this paper, we introduce a new concept, burst index, to capture the dynamic nature of bursty traffic. Based on burst index, a dynamic multiplexing algorithm (DMA) is proposed. It balances the burstiness among connections dynamically. Relative burst index takes into account both the number backlog cells and their delays. As the relative burst index describes the burstiness of each connection more precisely, DMA is able to allocate the service dynamically and optimize the objective function.

In our analysis, we show the advantage of DMA over GPS, which is widely accepted as an ideal multiplexing algorithm. The maximum delay of DMA is also derived. Computer simulation shows that DMA improves the performance in term of reducing the maximum delay, the cell loss rate and the burstiness. Further work of extending the results to network links is now being carried out.

References

1. Raha, A., Malcolm N., Zhao, W., Hard real-time communications with weighted round robin service in ATM local area networks, *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems*, pp 96-103, 1995.
2. Saha, D. M., Sarit. T., Satish K., Carry-over round robin: a simple cell scheduling mechanism for ATM networks, *Proceedings of IEEE INFOCOM*, Vol. 2, pp 630-637, 1996.
3. Parekh, Abhay K. Gallager, Robert G., 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, pp 344-357, 1993.
4. Chiussi, F M., Francini, A., Kneuer, J. G., Implementing fair queuing in ATM switches - part 2: the logarithmic calendar queue, *IEEE Global Telecommunications Conference,* Vol. 1, pp 519-525,1997.
5. Chiussi, F M., Francini, *A.,* Implementing fair queuing in ATM switches - part 1: a practical methodology for the analysis of delay bounds, *IEEE Global Telecommunications Conference,* Vol. 1, pp 509-518,1997.
6. Saha, D., Mukherjee, S., Tripathi, S. K., Multirate scheduling for guaranteed and predictive services in ATM networks, *Proceedings of Real-Time Systems Symposium*, pp 155-164. 1996.
7. Kyeong S. Kim., Byeong G. L.，Three level traffic shaper and its application to source clock frequency recovery for VBR video services in ATM networks, *IEEE/ACM Transactions on Networking,* Vol 34, pp 450-458，1995.
8. Cao Z., et.al，Performance analysis of an N-level shaper for VBR video traffic in ATM networks, *Communication Technology Proceedings,* vol 1, pp 271-275，1996.
9. Yuan Miao, Chee Kheong Siew, *Dynamic bandwidth allocation for VBR multiplexing*, Proceeding of The Sixth International Conference on Distributed Multimedia Systems, 1999.