Proceedings of International Conference
on Networking and Multimedia

A Rate Based Fair Queueing Scheme with Minimum Bandwidth Guarantee for ATM Network

Yen-Jen Chen and Suh-Yin Lee
Department of Computer Science and Information Engineering,
National Chiao Tung University, Hsin-Chu, Taiwan, R.O.C.
ctchen@csie.nctu.edu.tw

Abstract

The rate-based congestion control scheme was adopted for the available bit rate (ABR) service in ATM forum. When using the ABR service to support applications, there are two important issues on the packet scheduling. One is the fairness on the bandwidth allocation; another is the guarantee of minimum bandwidth. We propose a scheduling scheme, called minimum-bit-rate fair queueing (MBRFQ), which inherits the fairness nature of the self-clocked fair queueing scheme (SCFQ) and guarantees minimum bandwidth. The fairness of the scheme similar to SCFQ can be proved via mathematical analysis. On the other hand, the nature of the minimum bandwidth guarantee can be verified by simulation results. Our scheme not only possesses the capability of the two issues required by ABR services, but also guarantees the bounded delay and delay jitter on the constant bit rate (CBR) services. This makes the scheme more applicable for broadband services.

1. Introduction

Since asynchronous transfer mode (ATM) became the solution [3] for B-ISDN, the ATM Forum has defined several service categories [1], shown as below, to support user applications. The constant bit rate (CBR) service is used to support circuit switching connections. The variable bit rate (VBR) is used in the real-time packet switching. The unspecified bit rate (UBR) supports applications with no guarantee; it only provides minimal service capability for applications. With dynamic monitoring of the status of network system and adjusting the amount of input traffics, the available bit rate (ABR) service can support a number of applications, especially those with vague requirements for bandwidth, delay and loss. Therefore, the ABR service becomes the focus of ATM forum, since the most of applications can not specify their requirements definitely.

For the ABR service, ATM forum adopts the rate-based congestion control scheme [1] as the standard of its flow control mechanism. There are two important issues proposed in the rate-based congestion control scheme. One issue is the fairness of bandwidth allocation between connections; the other is the guarantee of minimum bit rate (MBR) of a connection. The rate fairness issue plays an essential role in ABR service to prevent the bad-behavior users from occupying the bandwidth belonging to the well-behavior users. The MBR issue extends the scope of the ABR service in the following ways [1]:
- To support the replacement of the resources supporting fixed-bandwidth with ABR connections.
- To allow the prioritization of ABR traffic.
- To support quasi-real-time applications.
- To inter-operate with other emerging protocols, e.g., RSVP [6].

Although the MBR issue is argued with its necessity, we think it is important and can give a good solution to a number of real-time applications while the VBR service has not been done well.

The concept of rate fairness is first applied to the *fluid-flow traffic model* by Demers, *et al.* [17]. This is an idealized model in which multiple connections can receive service in parallel, i.e. simultaneously. Demers, *et al.* defined the rate fairness and design a flow-fluid fair queueing (FFQ) scheme to implement it. However, in a realistic system, connections are served concurrently, since the server serves packets, one at a time. This service model is called *packet traffic model.* The model is merely a special case of fluid-flow traffic model if the same input packet traffic in the two models is scheduled with the same departure sequence. The rate fairness queueing scheme implemented in packet traffic model by Parekh and Gallager [18, 19] is called packet fair queueing (PFQ). The drawback of the PFQ is the complex computation in packet scheduling. Therefore, Golestani [2] proposed a self-clocked fair queueing (SCFQ) scheme to solve the problem.

Based on the SCFQ, we design a new packet scheduling scheme called minimum-bit-rate fair queueing (MBRFQ), which guarantees both the rate fairness and the MBR issue. The MBRFQ proportionally allocates bandwidth to each connection whose packets are accumulated in queue, and if congestion occurs, it would change the proportion of the bandwidth allocation to guard the minimum bit rate of each connection.

We describe the FFQ, PFQ, and SCFQ schemes in section 2, and propose the MBRFQ scheme in section 3. The analysis of the rate fairness of MBRFQ is shown in section 4. In section 5, the simulation results explain that the scheme can guarantee minimum bit rate when congestion occurs. Finally, conclusions are given in section 6.

2. Background

The following discussion is focused on the busy period of the queueing system of a link. We first introduce some notations. The transmission rate of the link is *C*. Let the set of the connections *k* set up on this link be denoted by *K*. We say that connection *k* is backlogged at *t* if there are at least one packet of *k* whose service has not finished at *t*. Let $A(t)$ and $B(t)$ denote the set of the connections not backlogged and the set of connections backlogged at *t*, respectively. Also define by $\mathcal{A}(t_1,t_2)$ and $\mathcal{B}(t_1,t_2)$, the set of connections continuously non-backlogged and the set of connections continuously backlogged from t_1 to t_2, respectively. $\mathcal{W}_k(t_1,t_2)$ is defined as the data amount of connection *k* having been transmitted during (t_1,t_2). Let r_k and σ_k denote the requested rate and minimum rate of connection *k*, respectively. The normalized service received by connection *k* during (t_1,t_2) is denoted by $w_k(t_1,t_2)$ and defined as follows:

$$w_k(t_1,t_2) \equiv \mathcal{W}_k(t_1,t_2)/r_k\,. \tag{1}$$

We define the fluid-flow fair queueing (FFQ) as the normalized services offered to two different backlogged connections in time interval (t_1,t_2) are equivalent, i.e.

$$w_i(t_1,t_2) = w_j(t_1,t_2)\,. \tag{2}$$

Since the packet traffic model is the special case of the fluid-flow traffic model in packet departure scheduling, to implement the packet fair queueing (PFQ) scheme is only to use the packet's departure sequence of FFQ to schedule the PFQ. However, it is difficult to find the actual departure time of a packet when it is arriving in a FFQ system. Fortunately, with the *virtual time* concept from Zhang [21], there is a mapping from actual time to virtual time in the FFQ system, and the mapping function is mono-increasing and easily found. Since the objective of packet scheduling is to find the departure order of incoming packets, the mono-increasing nature of the mapping function makes the virtual time correctly and efficiently replace the actual time on the role of scheduling.

The mapping between virtual time $v(t)$ and actual time *t* in the FFQ system is shown as below.

$$v(0) \equiv 0\,; \tag{3}$$

$$\frac{d\,v(t)}{d\,t} = \frac{C}{\sum_{i \in B(t)} r_i}\,. \tag{4}$$

Eq. (4) depicts that the differential of virtual time to actual time is equal to the ratio of link's transmission rate and the summation of the specified rates of backlogged connections at *t*. The derivation of Eq. (4) is shown in details in the document [4].

We describe how to find a packet's virtual departure time in the FFQ system. Let P_k^i be the *i*th packet of connection *k*, and d_k^i and a_k^i be the actual departure and arrival time of P_k^i, respectively. L_k^i is the length of P_k^i. Time stamp F_k^i denotes the virtual departure time of P_k^i, which is written as

$$F_k^i \equiv v(d_k^i)\text{, where } k \in K,\ i = 0, 1, 2, \dots \tag{5}$$

We can evaluate F_k^i by the following expressions:

$$F_k^0 \equiv 0\,; \tag{6}$$

$$F_k^i = \max\{F_k^{i-1}, v(a_k^i)\} + \frac{L_k^i}{r_k}, \tag{7}$$

where $k \in K$, and $i = 1, 2, \dots$

Eq. (7) means the key of finding the virtual departure time is to find the virtual arrival time, $v(a_k^i)$.

PFQ discipline is implemented by scheduling the departure order of incoming packets by the time stamps calculated in Eq. (6) and (7). However, to compute the value of $v(a_k^i)$ by Eq. (4), we must keep track of the backlog set $B_k(t)$. If the set changes frequently in short time interval, the computation of $v(a_k^i)$ would become the bottleneck of performance. To solve the problem, the SCFQ provides a good solution. Instead of referring to the virtual time of the FFQ, the SCFQ uses its own virtual time to find the time stamps of packets for scheduling. This makes the computation very simple. For distinguishing it from FFQ, we use different denotation in the following explanation.

The SCFQ scheme is shown as below. Let each arriving packet P_k^i be marked with a time stamp $\hat{F}_k^i$.

$$\hat{F}_k^i \equiv 0 \text{, if } i = 0; \tag{8}$$

$$\hat{F}_k^i \equiv \max\{\hat{F}_k^{i-1}, \hat{v}(a_k^i)\} + L_k^i / r_k \text{ , if } i \neq 0, \tag{9}$$

where $\hat{v}(t)$ denotes the virtual time at the actual time t in the SCFQ. Let $\hat{s}_h^j$ and $\hat{d}_h^j$ be the respective actual time when P_h^j starts and finishes service; that is, they are the actual sending and departure times. $\hat{v}(t)$ is defined as

$$\hat{v}(t) \equiv \hat{F}_h^j \text{, if } \hat{s}_h^j < t \leq \hat{d}_h^j. \tag{10}$$

The fairness in the SCFQ is merely approximate because the virtual arrival time $\hat{v}(a_k^i)$ in here is different from $v(a_k^i)$ in FFQ. However, Golestani [2] proves that the approximate fairness is under a near optimal bound. We show the bound in the following. Let $L_k^{\max}$ be the length of the longest packet of connection k. For any connections i and j, $i, j \in \hat{\mathcal{B}}(t_1, t_2)$, according to the concept from Eq. (2), the error of fairness between connection i and j is defined as $|\hat{\omega}_i(t_1,t_2) - \hat{\omega}_j(t_1,t_2)|$ and its bound is as follows:

$$|\hat{\omega}_i(t_1,t_2) - \hat{\omega}_j(t_1,t_2)| \leq \frac{L_i^{\max}}{r_i} + \frac{L_j^{\max}}{r_j}; \tag{11}$$

where the meanings of $\hat{\mathcal{B}}(t_1,t_2)$ and $\hat{\omega}_k(t_1,t_2)$ are the same as those of $\mathcal{B}(t_1,t_2)$ and $\omega_k(t_1,t_2)$, but only in different systems.

3. Proposed Scheme and Architecture

Our proposed scheme, MBRFQ, implements the minimum-bit-rate and bandwidth fairness issues in the packet traffic model. It schedules the packets in different way depending on which state, either normal or abnormal, the system stays. Initially, the system is in normal state, and, for a connection k, we use its requested rate r_k as its share of bandwidth. When the combination of backlogged connections in the system makes the original bandwidth allocation not satisfy some connection's minimum bit rate, we must replace r_i with σ_i as the new share base of bandwidth for each i. Then, the system is in abnormal state until a new combination of backlogged connections can make system return to normal state. In the following context, we give the abnormal state a more representative name 'MBR state'.

For keeping the fairness of bandwidth allocation, regardless of the states in which the system stays, we can not assign the departure time stamp to a packet when it arrives, since the state might not be the same as the packet is served. We utilize a per-connection queue structure to assign the departure time stamp to the packet when it becomes the front of the queue of the connection it belongs to. Select the packet with the minimum time stamp among the front of all queues, and then serve it.

In the following, Theorem 1 depicts how to know the time when a state changes. The formal definition of the MBRFQ is described in Definition 1 and its operation is shown in Algorithm MBRFQ_OP.

Theorem 1. During a busy period, the MBRFQ system will enter the MBR state if

$$C / \sum_{i \in \tilde{B}(t)} r_i < \max_{j \in \tilde{B}(t)} \{\sigma_j / r_j\} \tag{12}$$

The proof is depicted in the document [4].

Definition 1. When a packet P_k^i arrives the MBRFQ system, it is tagged with a time stamp $\tilde{F}_k^i$ as its departure priority. The smaller the $\tilde{F}_k^i$, the earlier the departure. The time stamp is defined as follows:

$$\tilde{F}_k^i \equiv 0 \text{, if } i = 0; \tag{13}$$

$$\tilde{F}_k^i \equiv \max\{\tilde{F}_k^{i-1}, \tilde{v}(a_k^i)\} + \frac{L_k^i}{R_k} \text{, if } i \neq 0, \tag{14}$$

where $\begin{cases} R_k = r_k \text{ , in normal state} \\ R_k = \sigma_k \text{ , in MBR state} \end{cases}$

and $\tilde{v}(t)$ denote the virtual time in the MBRFQ system at actual time t. Let $\tilde{s}_h^j$ and $\tilde{d}_h^j$ be the respective time instants when P_h^j starts and finishes service in the MBRFQ system; that is, they are the actual sending time and departure time. Then, $\tilde{v}(t)$ is defined as below:

$$\tilde{v}(t) \equiv \tilde{F}_h^j \text{, where } \tilde{s}_h^j < t \leq \tilde{d}_h^j. \tag{15}$$

According to Definition 1, the system architecture is designed as Fig. 1. Each connection established in the system has a dedicated queue where only the control informations of the packets are stored, and the packet bodies are stored in the shared buffer. In Fig. 1, assume there are total n backlogged connections at time t, denoted by 1, 2, 3, ..., n, whose packets are scheduled and transmitted by the server. The server has two parts: scheduler and transmitter. The transmitter transmits a packet bit by bit. The scheduler has two tasks. One is the adding of a time stamp into the candidate packet, which is in the front of each connection queue; the other is the selection of a candidate packet into the transmitter.

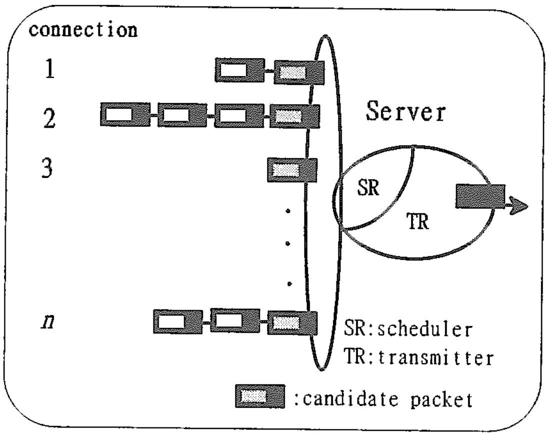

Fig. 1. MBRFQ system architecture

The operation of the MBRFQ system is described in the following algorithm.

```
Algorithm MBRFQ_OP ();
{
  define {
    event1 ≡ a packet $P_k^i$ arrives;
    event2 ≡ a packet $P_h^j$ departs;
  }
  label waiting, s1, s2, s3, p1, p2, p3;
  variable {
    $\tilde{B}$ : set of connections;
      /* currently backlogged connection set */
    $\tilde{v}$ : real number;
      /* current virtual time */
    event : {event1, event2};
  }
  $\tilde{B} = \varnothing$;
  $\tilde{v} = 0$;
waiting: wait (event);
  if (event == event1) {/* $P_k^i$ arrives */
    if ($k \notin \tilde{B}$) {
      $\tilde{B} = \tilde{B} \cup \{k\}$;
s1:   if ($C / \sum_{x \in \tilde{B}} r_x < \max_{y \in \tilde{B}}\{\sigma_y / r_y\}$)
        $R_i = \sigma_i, \forall i \in \tilde{B}$;
      else
        $R_i = r_i, \forall i \in \tilde{B}$;
      $P_k^i$ becomes a candidate packet;
p1:   $\tilde{F}_k^i = \tilde{v} + L_k^i / R_k$;
      if ($\tilde{v}$ == 0)
      /* the arrival of $P_k^i$ starts the busy period */
        move $P_k^i$ into transmitter;
    }
    else
      if (the queue of $i$ is empty) {
        $P_k^i$ becomes a candidate packet;
p2:     $\tilde{F}_k^i = \tilde{v} + L_k^i / R_k$;
      }
  }
  if (event == event2) {/* $P_h^j$ departs */
    if ($h$ is not backlogged) {
      $\tilde{B} = \tilde{B} - \{h\}$;
      if ($\tilde{B} = \varnothing$) {
      /* the departure of $P_h^j$ ends the busy period */
        $\tilde{v} = 0$;
        goto waiting;
      }
s2:   if ($C / \sum_{x \in \tilde{B}} r_x < \max_{y \in \tilde{B}}\{\sigma_y / r_y\}$)
        $R_i = \sigma_i, \forall i \in \tilde{B}$;
      else
        $R_i = r_i, \forall i \in \tilde{B}$;
      $P_k^i$ becomes a candidate packet;
    }
s3: Find $\tilde{F}_z^m$, where $\tilde{F}_z^m$ = min {the time stamps of all candidate packets};
    move $P_z^m$ into transmitter;
    $\tilde{v} = \tilde{F}_z^m$;
    if ( $P_z^{m+1}$ is in the queue of $z$) {
      $P_z^{m+1}$ becomes a candidate packet;
p3:   $\tilde{F}_z^{m+1} = \tilde{v} + L_z^{m+1} / R_z$;
    }
  }
  goto waiting;
}
```

In the algorithm above, the three statements labeled by s1, s2, and s3 exhaust most of the computation time, since the time complexity of the computation max{...} (min{...}) may be O(N), where N is the maximum number of the connections allowed to establish in the server. We use a heap [7] structure for the operation of max{...} (min{...}), and this decreases the complexity to O(log N). Since a heap tree can keep the maximum (minimum) value at the top, we can find the maximum (minimum) in O(1) and insert or delete a value in O(log N). Another advantage is that heaps are easily implemented by hardware, because its data structure can be a linear array.

4. Analysis of Fairness

The degree of fairness in the bandwidth allocation of a queueing system is measured by the normalized service difference of any pair of backlogged connections. Let $\tilde{\mathcal{A}}(t_1,t_2)$ and $\tilde{\mathcal{B}}(t_1,t_2)$ denote the non-backlogged and backlogged connection set in the MBRFQ system, respectively. The following theorem illustrates the MBRFQ is a good scheme in rate fairness, since it has the similar bound to the SCFQ.

Theorem 2. $\forall i,j \in \tilde{\mathcal{B}}(t_1,t_2)$, the difference of the normalized services of any two backlogged connections in a busy period (t_1,t_2) is bounded as follows:

$$|\tilde{w}_i(t_1,t_2)-\tilde{w}_j(t_1,t_2)| \le L_i^{\max}/\sigma_i + L_j^{\max}/\sigma_j \,.$$

The proof of Theorem 2 is shown in [4].

5. Simulation Result on MBR Issue

For verifying that MBRFQ supports the guarantee of minimum bit rate (MBR), we performed simulations and do a lot of tests with some representative input traffics. These are variable-rate, constant-rate, and bursty traffics. For saving space, we only show one test in here. The tests mainly focus on the constant-rate traffics, since this kind of traffics must be served with enough bandwidths at any time. The requested rates of constant-rate connections become the minimum bit rate guaranteed by the MBRFQ. Therefore, if a constant-rate traffic always departs from the system with a acceptable bounded delay, the scheme is successful in the guarantee of the MBR. We also take care of the bursty and variable-rate traffics and have observation and analysis [4] on them. However, due to limited space, we do not show them in here.

The system architecture is like Fig. 1, and we set up five connections in the system with various traffic characteristics. The parameters of the test are shown in Table 1. For an input traffic from connection *k*, its characteristics is described by its peak rate r_k, average rate λ_k, and average bursty length β_k. The system will allocate bandwidth to connection *k* according to either the peak rate r_k or minimum rate σ_k, depending on the state in which the system stays.

In the test, connection 0 is modeled as a variable-rate traffic with a peak rate r_0. Let *L* be the length of the ATM cell, and then the minimum interarrival time of connection 0, denoted by η_0, is equal to L/r_0. We first generate the cell interarrival time according the poisson process whose average arrival rate is Λ_0. If the interarrival time is less than η_0, it will be replaced with η_0. Therefore, the rate of the connection is under the peak rate r_0. The relation among the average rate λ_0, Λ_0, and η_0 is as follows:

$$1/\lambda_0 = 1/r_0 + 1/\Lambda_0 \cdot e^{-\Lambda_0/r_0} \,. \tag{16}$$

This connection provides a variable-rate traffic, shown in Fig. 3, and makes the traffics of other connections oscillate.

Another connection 1, 3, and 4 are modeled as constant rate traffics. They should obtain the fixed bandwidth in any time since their minimum rate is equal to their maximum rate, even if congestion occurs.

Finally, connection 2 is modeled as a bursty traffic. Its traffic source alternates between active and silent periods. We model the traffic by the two-state Modulated Markov Deterministic Process (MMDP [8] [9] [20]). Fig. 2 depicts its state transition diagram. Since the sending rate in the active period is r_2, the size of a cell time slot is L/r_2. Let the random variable *N* and *M* represent the lengths of active and silent period respectively in terms of number of time slots. The probability is denoted by *a* that no cell will arrive in the next time slot if the connection is active now. Also the probability is denoted by *b* that a cell will arrive in the next time slot if the connection is silent now. The probability distributions of *N* and *M* are both geometric, and shown as follows:

$$P_N(n)=(1-a)^n a \,,\ P_M(m)=(1-b)^m b \,,$$

where *m* and *n* are non-negative integer numbers.

Given the peak rate r_2, average rate λ_2, and the average burst length β_2 (i.e. the average active period in two-state MMDP), we have

$$E[N]=\beta_2 ;\ E[M]=\beta_2 \cdot \frac{r_2-\lambda_2}{\lambda_2}$$

$$a=\frac{1}{1+\beta_2} ;\ b=\frac{1}{1+\beta_2 \cdot \frac{r_2-\lambda_2}{\lambda_2}}$$

By r_2, *a*, and *b*, we can generate the traffic of connection 2, which is shown in Fig. 5.

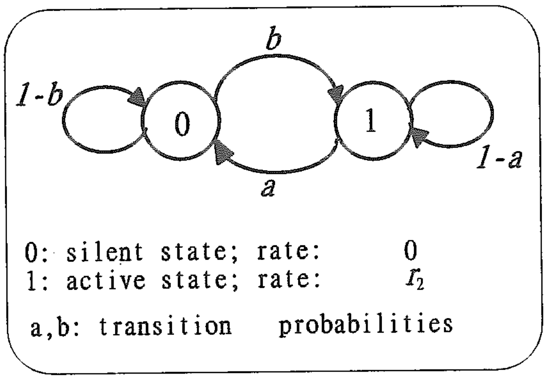

Fig. 2. two-state MMDP

With the test, the total peak rate of established connections is 191 Mbps, much larger than the server capacity 150 Mbps; that is, the server will be saturated often. However, the total average rate is 140.91 Mbps, which means queues will not grow infinitely. Assume, during time interval (t_1,t_2), connection 0, 1, and 2 are backlogged, i.e. $\tilde{\mathcal{B}}(t_1,t_2)=\{0,1,2\}$ and connection 3 and 4 are non-backlogged, i.e. $\tilde{\mathcal{A}}(t_1,t_2)=\{3,4\}$. For (t_1,t_2), if scheduling state is not changed to MBR state, connection 0, 1, and 2 would get the bandwidths of 75 Mbps, 25 Mbps, and 50 Mbps, respectively. Obviously, connection 1 lost the MBR guarantee. The similar situation may happen on other connections in other time. In the following, we analyze the simulation result of the test and explain the MBRFQ system can support MBR issue.

Consider the connection 1, 3 and 4 which are all constant-rate. Ideally, the cell interdeparture time of a constant-rate connection k should keep constant and be L/σ_k (which implies the departure traffic is "isochronous" [10]). However, this is impossible due to the traffic interferences from connection 0 and 2. The packet scheduling scheme is difficult to generate an isochronous traffic, unless adopts the mechanism of time framing [11] [12] with fixed time slot allocation in every frame. From our simulation, we find that the queueing delay of connection k is always under L/σ_k. Fig. 4 depicts the packet queueing delay of connection 1, and the other results can be seen in [4]. With the nature of bounded queueing delay, it is easy to form an isochronous traffic by adding a buffering technique to filter the delay jitters. This gives us a sufficient evidence to say the MBR issue is supported in the MBRFQ scheme.

6. Conclusions

Adding the minimum-bit-rate (MBR) issue into fair queueing schemes is important and not easy. With the issue, network providers can support real-time services and fairly guarantee their users with basic quality. However, doing this carelessly might harm the original fairness or not respond to the immediate congestion which can destroy the MBR guarantee. In the SCFQ scheme, whenever a packet arrives, the scheduler gives it a time stamp and then inserts it to the correct location in the scheduling queue. We call this way as pre-scheduling. Implementing the MBR issue by this mechanism will lose the MBR guarantee because the packet may be scheduled in normal state and sent out in MBR state. Instead, we schedule a packet when it becomes the candidate of entering server, and the mechanism is named critical-scheduling. This almost makes the scheduling and sending of a packet in the same system state. Even if they are in different state, the error is merely on the candidate packet.

Our proposed scheme, MBRFQ, adopts the critical-scheduling mechanism such that the system sensitively allocate proper bandwidth to a connection k according to either the peak rate r_k or the minimum rate σ_k. Therefore, our scheme can efficiently access the available bandwidth in the normal state and guarantee the minimum bandwidth whenever congestion occurs. Also the scheme can fairly allocate bandwidth to each connection. Such guarantee and support give many real-time applications a good solution.

References

[1] Flavio Bonomi and Kerry W. Fendick, "The Rate-Based Flow Control Framework for the Available Bit Rate ATM Service," IEEE Network Mag., pp. 25-39, Mar./Apr. 1995.

[2] S. Jamaloddin Golestani, "A Self-Clocked Fair Queueing Scheme for Broadband Applications," Proc. IEEE INFOCOM, 1994, pp. 636-646.

[3] Prycker, Martin de, "Asynchronous Transfer Mode : Solution for Broadband ISDN," 2nd edition, Ellis Horwood, 1993.

[4] Yen-Jen Chen and Suh-Yin Lee, "A Rate Based Fair Queueing Scheme with Minimum Bandwidth Guarantee for High Speed Network," Technique Report of Information Lab., Institute of C.S.I.E., National Chiao Tung University, Taiwan, R.O.C.

[5] Hui Zhang and Domenico Ferrari, "Rate-Controlled Static-Priority Queueing," IEEE INFOCOM, 1993, pp. 227-236

[6] L. Zhang *et al.*, "RSVP: A New Resource Reservation Protocol," IEEE Network, vol. 7, no. 5, pp. 8-18, Sept.,1993.

[7] Udi Manber, "INTRODUCTION TO ALGORITHM, A Creative Approach," Addison-Wesley, pp.68-70, 1989.

[8] Victor S. Frost and Benjamin Melamed, "Traffic Modeling for Telecommunications Networks," IEEE Communications Mag., pp. 70-81, Mar. 1994.

[9] Tsern-Huei Lee, Kuen-Chu Lai, and Shii-Tyng Duann, "Real-Time Admission Control for ATM Networks with Heterogeneous Bursty Traffic," International Communications Conference, 1994, pp. 80-85.

[10] Heinrich J. Stuttgen, "Network Evolution and Multimedia Communication," IEEE MultiMedia, pp.42-59, Fall 1995.

[11] S. Jamaloddin Golestani, "Congestion-Free Communication in High-Speed Packet Networks," IEEE Transactions on Communications, vol. 39, no. 12, pp.1802-1812, Dec. 1991.

[12] C. R. Kalmanek, H. Kanakia and S. Keshav, "Rate Controlled Servers for Very High-Speed Networks," IEEE Global Telecommunications Conference, 1990, pp. 0012-0020.

[13] Christopher Lefelhocz *et al.*, "Congestion Control for Best-Effort Service: Why We Need a New Paradigm," IEEE Network Mag., pp. 10-19, Jan./Feb. 1996.

[14] H. J. Chao and I. H. Pekcan, "Queue Management with Multiple Delay and Loss Priorities for ATM Switches," IEEE INFOCOM, 1994, pp.1184-1189.

[15] D.-S. Lee, B. Sengupta, "Queueing Analysis of a Threshold Based Priority Scheme for ATM Networks," IEEE Journal on Selected Areas in Communications, vol. 1, no. 6, pp.709-717, Dec. 1993.

[16] Cui-Qing Yang and Alapati V. S. Reddy, "A Taxonomy for Congestion Control Algorithms in Packet Switching Networks," IEEE Network Mag., pp. 34-45, July/August 1995.

[17] A. Demers, S. Keshav, and S. Shenkar, "Analysis and simulation of a fair queueing algorithm," Proc. SIGCOMM, Sep. 1989, pp.1-12.

[18] A. K. Parekh and R. G. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks--- The Single Node Case," Proc. IEEE INFOCOM, 1992, pp. 915-924.

[19] A. K. Parekh and R. G. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks--- The Multiple Node Case," Proc. IEEE INFOCOM, 1993, pp. 521-530.

[20] Chih-Hen Lin, Chung-Ju Chang, and Dah-Sheng Guan, "A Power-Spectrum Based Call Admission Control for ATM Networks," Proc. Workshop on Communication Networks, 1995, pp. 227-243.

[21] L. Zhang, "Virtual Clock: A New Traffic Control Algorithm for Packet Switching," ACM Transactions on Computer Systems, 9(2): pp. 101-124, May 1991.

C: server capacity, $C = 150$ Mbps

connection k	input traffic characteristics	peak rate r_k (Mbps)	average rate λ_k (Mbps)	minimum rate σ_k (Mbps)	average burst β_k (cell)
0	poisson process(1) under a max. rate	90	59.91	50	(2)
1	constant bit rate	30	30	30	1
2	2-state MMPP	60	40	20	29
3	constant bit rate	10	10	10	1
4	constant bit rate	1	1	1	1

Table 1. traffic parameter specification

(1) The average rate, Λ_0, of the poisson process is 75 Mbps

(2) The field is ignored for poisson process, since it is useless on describing the process.

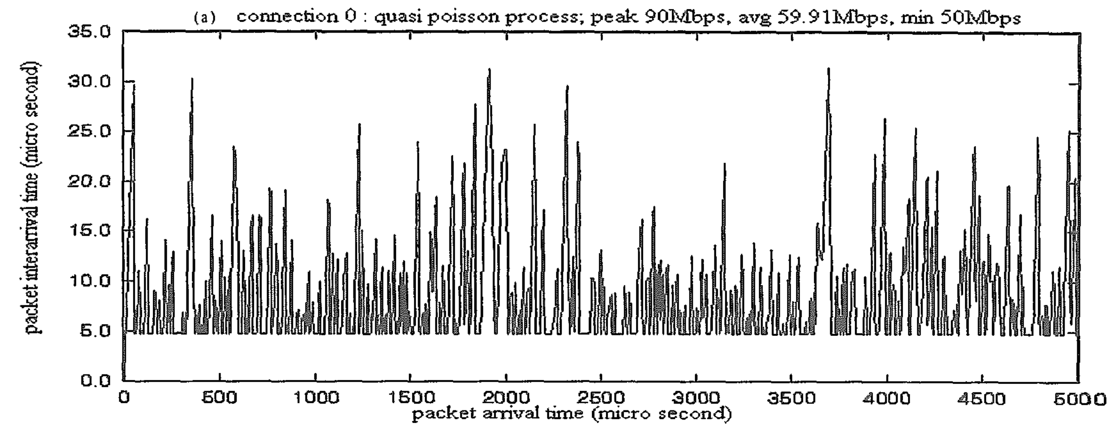

Fig. 3. The input traffic pattern of connection 0

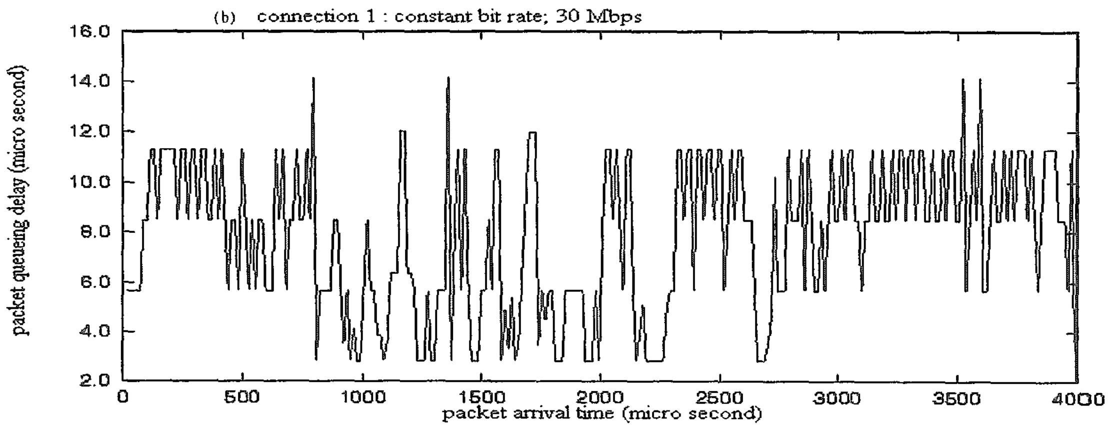

Fig. 4. The packet queueing delay of connection 1

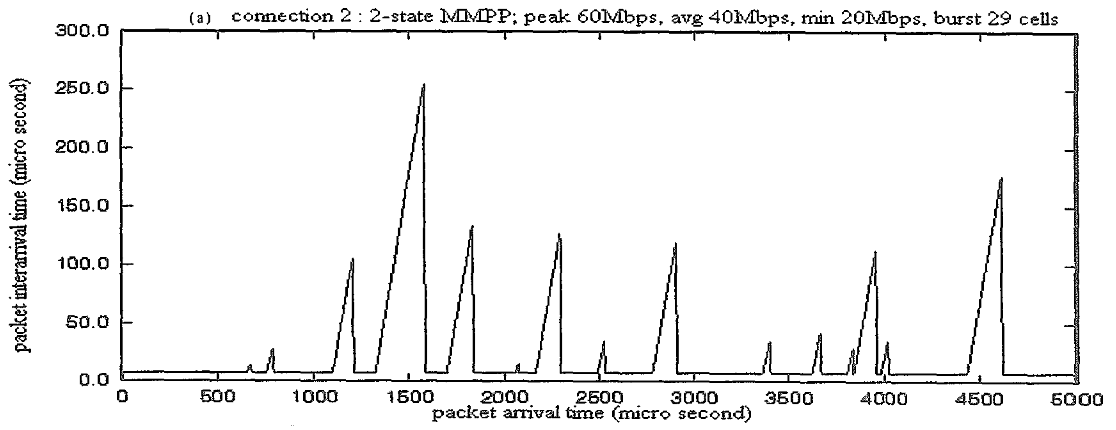

Fig. 5. The input traffic pattern of connection 2