The Study on Scheduling Algorithms in Packet Switched Networks

C. L. Lee, E. H. Hwang, and P. C. Wang

Abstract

One of most important issues in providing performance guaranteed service is the design of the packet scheduling algorithm at each switching node. In packet-switched networks, packets from different connections interact with each other at each switching node. Without proper control, the interaction of each packet will affect the performance of each connection. The packet scheduling algorithms control the order in which packets are serviced and determine the relation of the interaction at each connection. In this paper, we reviewed a lot of scheduling algorithms and traffic models.

We will study the property of these algorithms and compare these packet scheduling algorithms focusing on end-to-end delay bound, fairness, and implementation time complexity.

Key Words : Quality of Service Guarantees, Packet Scheduling Algorithm, Traffic Model

C. L. Lee、P. C. Wang : Instructor of the Information Management Department
E. H. Hwang : Assistant Professor of the Information Management Department

1 Introduction

A network with QOS service requires a resource reservation scheme to allocate network resources for individual connections. Two mechanisms affect the design of a resource reservation scheme are connection admission control mechanisms that limits the number of allowable connections and the traffic policing mechanisms that specify the traffic of individual connections [1,2]. With the proper design of these mechanisms the network will admits a large number of connections, leading to a high network utilization.

Future high-speed networks must support a wide variety of traffic classes with different QOS requirement. Depending on the type of QOS guarantee being provided, the service model can be classified into three classes: deterministic guarantee, statistical guarantee, and best effort services [3-6]. A deterministic guarantee is a QOS guarantee for a connection that holds for every packet transmitted over the network during the connection duration. Statistical guarantees, in contrast to deterministic guarantees, are not required to hold for every packet transmitted during the connection duration. For example, a typical statistical guarantee might require 95% of a connection's traffic to meet an end-to-end delay requirement of 100 ms. Since our research is limited to the deterministic guarantees, we will review these factors at section 2.

The Internet provides a best-effort service to all of its applications. That is, the Internet makes its best effort to transmit each of the sender's packets to the receiver as quickly as possible. One of most important issues in providing performance guaranteed service is the design of the packet scheduling algorithm at each switching node. In packet-switched networks, packets from different connections interact with each other at each switching node. Without proper control, the interaction of each packet will affect the performance of each connection. The packet scheduling algorithms control the order in which packets are serviced and determine the relation of the interaction at each connection. In this paper, we reviewed a lot of scheduling algorithms and traffic models. We will study the property of these algorithms and models and compare these packet scheduling algorithms focusing on end-to-end delay bound, fairness, and implementation time complexity.

The remainder of this paper is structured as follows. In section 2 we present the traffic model, the packet scheduling algorithm, and the methods for the computation of the end-to-end delay bound. In section 3 we compared and analyzed 5 rate-based packet scheduling algorithms focusing on end-to-end delay bound, fairness, and implementation time complexity. Finally, we will conclude our discussion in section 4.

2 Background Reviews

2.1 Traffic Characterization

A traffic characterization appropriate for use in a deterministic delay guaranteed service must satisfy four requirements [7]. First, the characterization must provide a worst case description of the source traffic that determine an upper bound on a source's packet arrival. Second, the characterization must conform to a parameterized traffic model such that a source can specify its traffic characterization to the network with parameters. Third, it must conform to traffic policing mechanisms, which can be implemented such that the network can enforce a source's traffic characterization. Finally, it must be sophisticated enough to describe the traffic accurately such that the connection admission mechanisms can estimate the resources requirements by the connection.

Since a deterministic service provides worst case guarantees, a traffic characterization must specify the worst case traffic of a connection [1,2]. Let A

denote the actual traffic on a connection, where $[\tau, \tau + t]$ denotes the traffic arrivals at time interval $[\tau, \tau + t]$. Then a worst case characterization of the traffic A is given by a traffic constraint function A^*, which provides an upper bound on A. That is, a function A^* provides a bound for A if for all times $\tau \geq 0$ and $t \geq 0$ the following holds:

$$A[\tau, \tau + t] \leq A^*(t)$$

Since a traffic constraint function A^* bounds the maximum traffic over any time interval of length t, the connection admission control can be made independent of the starting time of a connection. Practical traffic characterizations are obtained from a parameterized traffic model that express the maximum traffic admitted by some traffic policing mechanism.

Even the use of the traffic models is essential to practical traffic characteristics; many traffic models are not enough to characterize some traffic sources. For example, video traffic compressed with the MPEG compression algorithm has complex and irregular timely correlations those are difficult to characterize accurately with a function of few parameters. In general, a model with more parameters can achieve a more accurate or tight traffic constraint function. However, the additional parameterization causes an increase in the complexity of policing the traffic model. Therefore, the selection of an appropriate traffic model for a deterministic service must find a compromise between the high complexities preferred by the connection admission control mechanisms and the simplicity required for the implementation of traffic policing mechanisms [8,9].

In this section, we review 5 traffic models that have been considered for use in deterministic delay guaranteed service. We formulate the traffic constraint function A^* for each traffic model and discuss traffic policing mechanisms.

2.1.1 Peak-rate Model

The peak-rate model is the simplest and most widely used traffic model. For this model, two parameters are used to describe traffic for a connection: the minimum inter-arrival time $X_{\min}$ and the maximum transmission time $s^{\max}$ of any packet. The maximum traffic for a connection that conforms to peak-rate model is given by the following traffic constraint function:

$$A^{*}(t)=\left(\left\lfloor \frac{t}{X_{\min}} \right\rfloor+1\right)s^{\max}, \forall t \geq 0$$

2.1.2 (r,T) model

A rate parameter r and a framing interval T are used to describe the traffic for the (r,T)-model [10]. Time is partitioned into frames of length T, and the maximum traffic for a connection during any time frame is limited to rT bits. Hence, the (r,T) model enforces an average rate r and allows for some bursts.

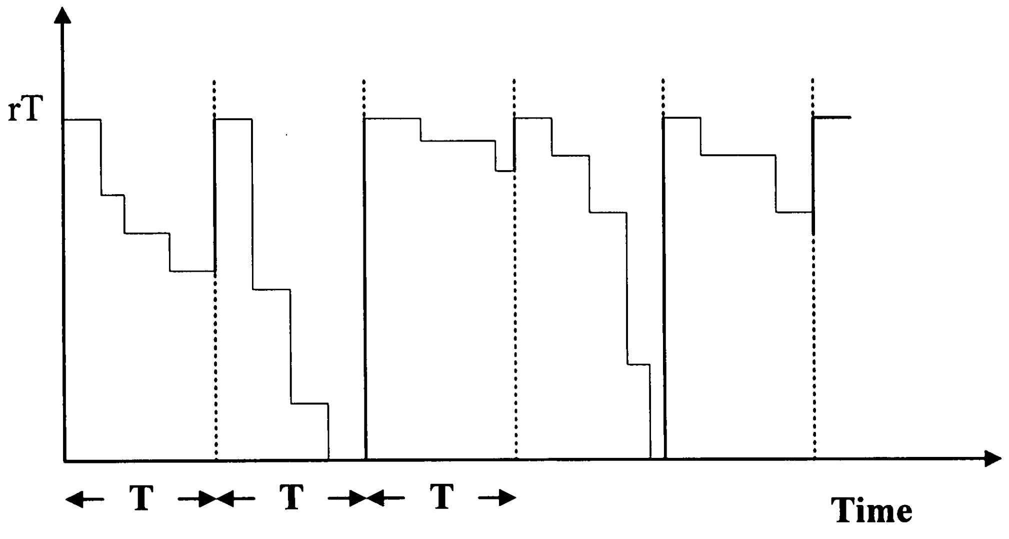

Figure 2.1 Illustration of the jumping window

Traffic conforming to the (r,T) model can be policed with the jumping window policing mechanism [11]. At the beginning of each frame, a credit counter is set to rT. When each bit enters the network, this counter is decremented by 1. Packets can only enter the network if sufficient credits are available, that is, the counter is always non-negative. Every T time units, the credit variable is reset to rT. If the arrival of a packet will result in a negative counter value, it will not be allowed into the network and will be discarded or buffered until next frame. This mechanism is illustrated in Fig. 2.1, where we plot the value of the counter over the period of 4 frame times. In this figure, the credit variable is reduced whenever a packet arrives to the scheduler. The credit variable is not depleted during the first interval, but it is shown to fully drain in next frame interval. The traffic constraint function for the (r,T)-model is given by

$$A^{*}(t)=\left(\left\lfloor \frac{t}{T}\right\rfloor+1\right)rT,\forall t\geq 0$$

In Fig. 2.2, we illustrated the traffic constraint function.

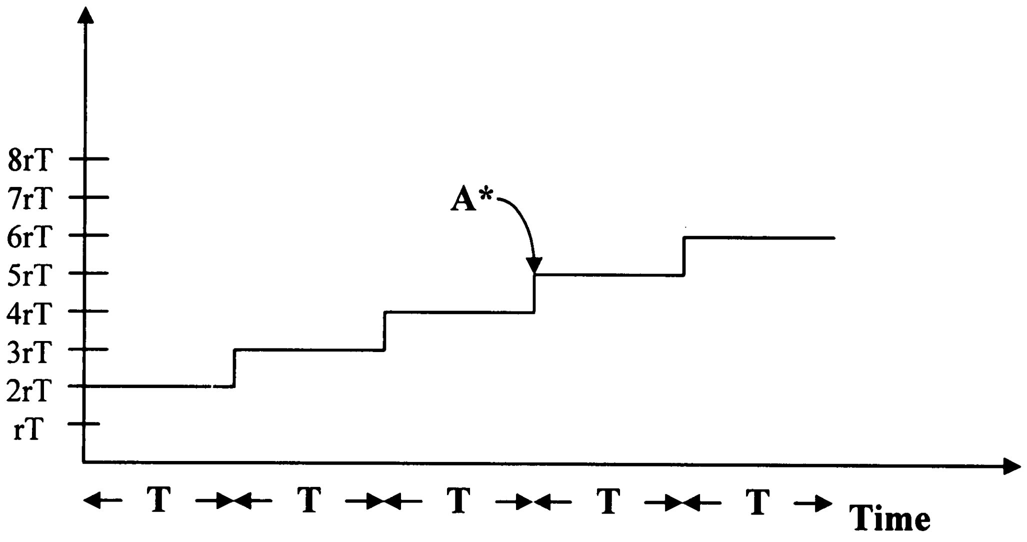

Figure 2.2 Traffic constraint function for the (r,T) model

2.1.3 (σ, ρ) Model

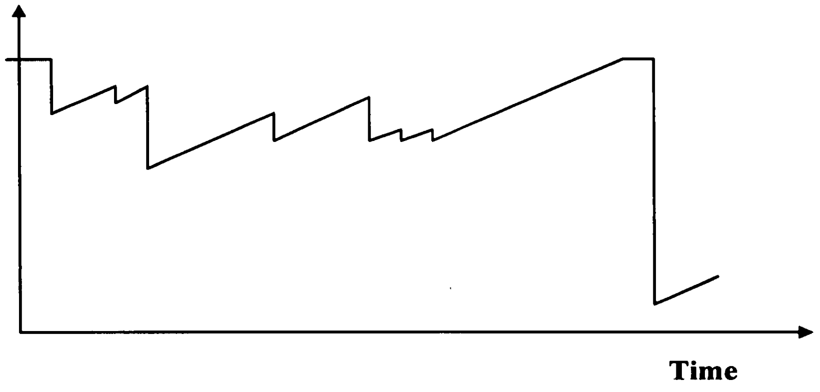

Figure 2.3 Illustration of the leaky bucket for the (σ, ρ) model

A burst parameter σ and an average rate parameter ρ are used to describe the traffic of the (σ, ρ) model [2]. The traffic from a connection over any interval of length t is limited to $\sigma + \rho t$. The traffic conform to the (σ, ρ) model can be policed by a well-known leaky bucket policing mechanism [12,13]. A credit counter is initialized to σ, and traffic may only enter the network if the credit counter is nonzero. The credit counter is decremented for each bit that enters the network, and the credit counter is incremented continuously at rate ρ when the value of credit counter is less than σ. The credit counter of a leaky bucket is illustrated in Fig. 2.3. In this figure, the credit counter is reduced by the packet transmission time of new packet arrivals and that is always increased at rate ρ when its value is less than σ. This model enforces a rate ρ while allowing some burst up to σ. Detailed discussion about leaky bucket can be found in ref. [12].

The traffic constraint function for (σ, ρ) model is denoted by L^* and obtained as follows:

$$L^*(t) = \sigma + \rho t, \forall t \geq 0$$

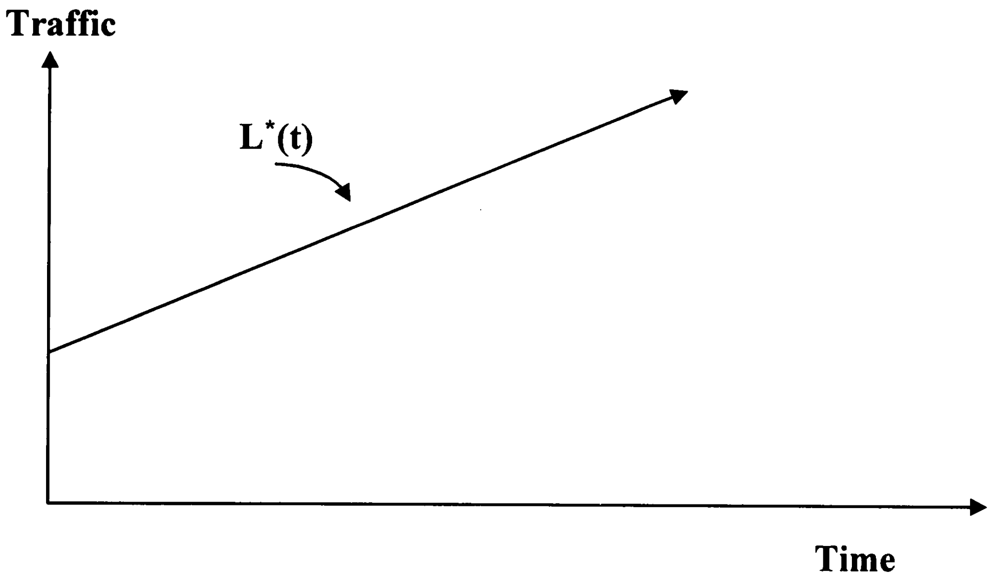

Figure 2.4 Traffic constraint function for the (σ,ρ) model

We illustrate $L^*(t)$ in Fig. 2.4. The (σ,ρ) model is more flexible than the (r,T) model. In (σ,ρ) model, the burrstones parameter σ is independent of the average rate ρ. In (r,T) model, the maximum burst is proportional to the rate and is given by $2rT$.

2.1.4 $\left(\vec{\sigma},\vec{\rho}\right)$ Model

A generalization of the (σ,ρ) model is the $\left(\vec{\sigma},\vec{\rho}\right)$ traffic model that corresponds to a traffic policing mechanism where multiple leaky buckets are connected in series. For a connection that conforms to the $\left(\vec{\sigma},\vec{\rho}\right)$ model with a set of n pairs $\{(\sigma_i,\rho_i)\}_{1\le i\le n}$, the amount of traffic admitted to the network is limited by each of the (σ_i,ρ_i) pairs [2,14]. The traffic constraint function, denoted by L_n^*, is a function consisting of n piecewise linear segments and is given by

$$L_n^*(t) = \min_{1\le i\le n}\{\sigma_i + \rho_i t\}, \forall t \ge 0$$

We illustrate the traffic constraint function L_n^* for three (σ,ρ) pairs in Fig. 2.5.

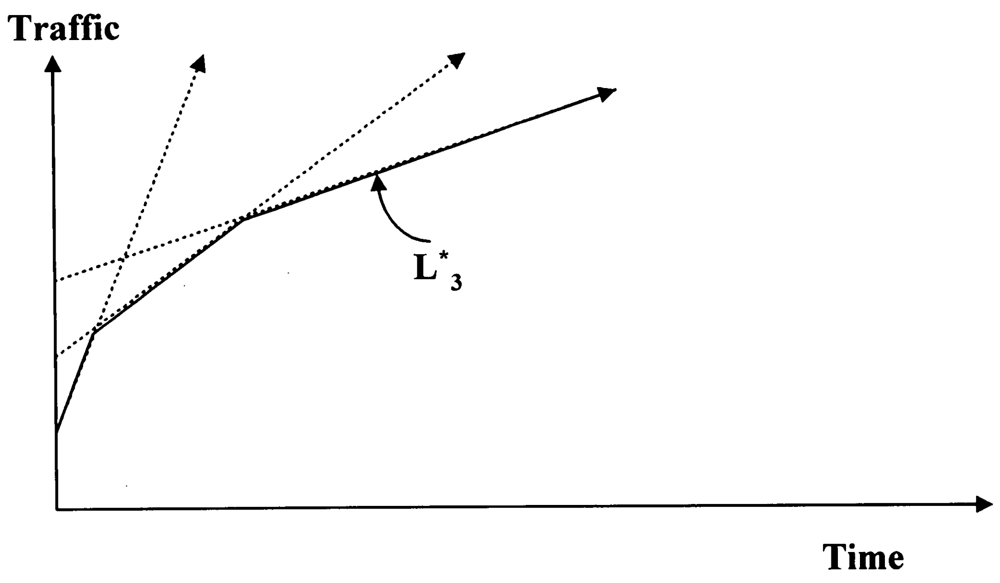

Figure 2.5 Traffic constraint function for the $\left(\vec{\sigma},\vec{\rho}\right)$ model

2.1.5 $\left(X_{\min}, X_{ave}, I, s^{\max}\right)$ model

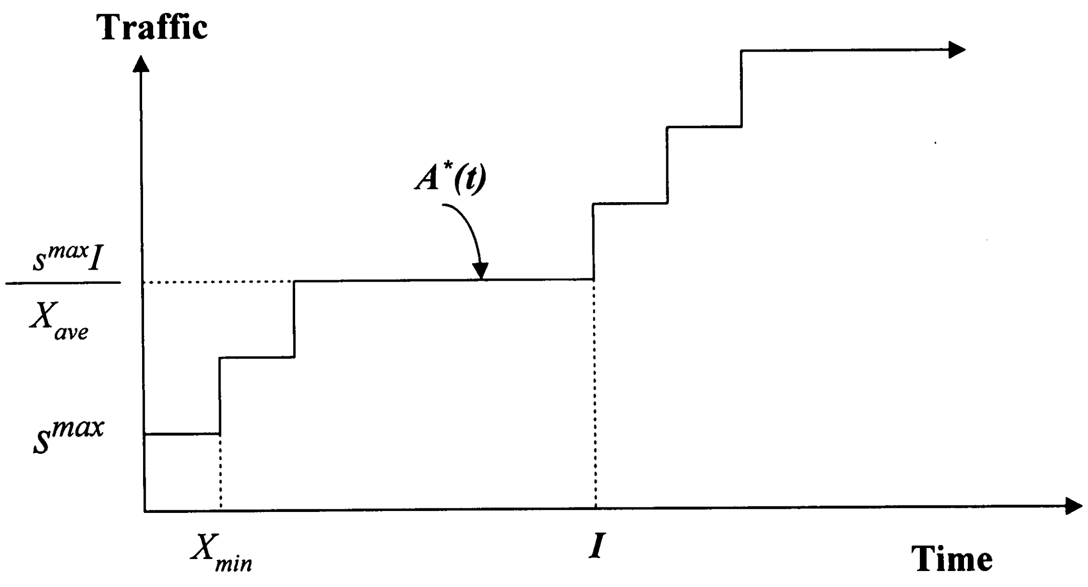

Figure2.6 Traffic constraint function for the $\left(X_{\min}, X_{ave}, I, s^{\max}\right)$ model

In the $(X_{\min}, X_{ave}, I, s^{\max})$ model, $X_{\min}$ is the minimum packet inter-arrival time, X_{ave} is the maximum average packet inter-arrival time over any time interval of length I, and $s^{\max}$ is the maximum packet transmission time [3,5]. This traffic model limits the peak rate of a connection and ensures that the traffic admitted during any interval of length I is at most $\frac{Is^{\max}}{X_{ave}}$. The traffic constraint function is illustrated in Fig. 2.6 and given as follows:

$$A^{*}(t) = \left\lfloor \frac{t}{I} \right\rfloor \frac{Is^{\max}}{X_{ave}} + \min\left\{ \left\lceil \left(\frac{t}{I} - \left\lfloor \frac{t}{I} \right\rfloor \right) \frac{I}{X_{\min}} \right\rceil, \frac{I}{X_{ave}} \right\} s^{\max}, \forall t \geq 0$$

The traffic models described above have all been considered for use in deterministic delay guaranteed service. However, the choice of a particular traffic model must be consider the following two requirements:

- The model should accurately describe the traffic pattern of the connection.
- The policing mechanism should have simple implementation.

The evaluation of the various traffic models with respect to these two requirements can be found in ref. [11,14,15].

3 Packet Scheduling Algorithms

Packets from different connections multiplexed on a single output link of a switching node are stored in a transmission queue, and the packet scheduler at the switching node determines the transmission order of these packets. The set of rules a packet scheduler uses for ordering queued packets is called the packet scheduling algorithm. The packet scheduling algorithm at the output link of a switching node manages three independent resources: bandwidth (which packets are transmitted), promptness (when are those packets transmitted) and buffer space (which packets are discarded)[16]. The allocation of these three resources will affect the three performance parameters: throughput, delay, and packet loss ratio.

A packet scheduling algorithm can be classified as work-conserving and non-work-conserving [17,18]. With a work-conserving packet scheduling algorithm, a switching node is never idle when there is a packet to transmit. With a non-work-conserving packet scheduling algorithm, the switching node may be idle even when there are packet waiting to be transmitted. In this section, we will describe five work-conserving packet scheduling algorithms: Virtual Clock (VC)[19], Weighted Fair Queueing (WFQ)[20-22], Self-Clocked Fair Queueing (SCFQ)[23], Start-time Fair Queueing (SFQ)[24], and Minimum Starting-tag Fair Queueing (MSFQ).

3.1 Virtual Clock

The VC packet scheduling algorithm aims to emulate the Time Division Multiplexing (TDM) system. On packet arrival, the VC server will assign each packet a virtual clock value and service the packet in their increasing order of virtual clock values. If connection f is reserved a rate r_f at server i, then the virtual clock value for the $k-th$ packet of connection f, p_f^k, denoted by $VC^i(p_f^k)$ is computed as follows [45][57]:

$$VC^i(p_f^0) = 0$$

$$VC^i(p_f^k) = \max\{A^i(p_f^k), VC^i(p_f^{k-1})\} + \frac{l_f^k}{r_f} \quad \text{......................................} \quad (3.1)$$

Where $A^i(p_f^k)$ be the arrival time of packet p_f^k at server i and l_f^k is the length of packet p_f^k. The delay guarantee of VC was presented in [25,26] under the condition that, $\sum_{n \in B^i(t)} r_n \le C^i$, where $B^i(t)$ represents the set of backlogged connections at time t, C^i is the capacity at server i, and n is the index of connection.

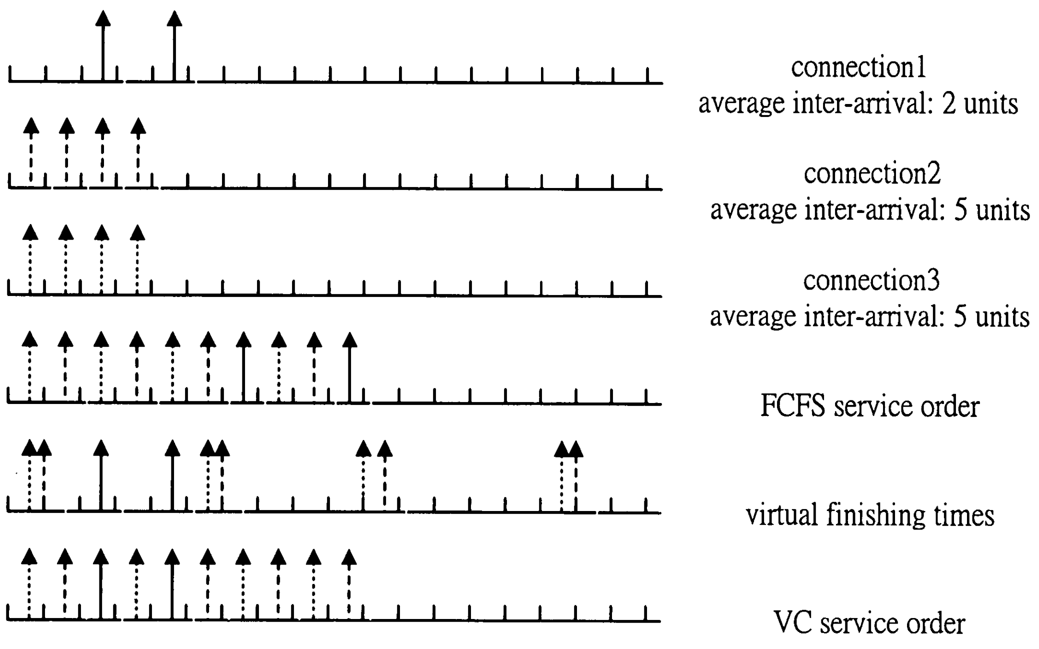
connection1
average inter-arrival: 2 units
connection2
average inter-arrival: 5 units
connection3
average inter-arrival: 5 units
FCFS service order
virtual finishing times
VC service order

Figure 3.1 Illustrations of the FCFS and VC

In Fig. 3.1, we use an example (given by ref. [24]) to illustrate how VC works. In this example, there are three connections that share the same output link. Each connection specifies its traffic characteristics and reserves enough resources. Connection 1 has an average packet inter-arrival time of 2 time units; connection 2 and connection 3 have an average inter-arrival time of 5 time units. We assume packets from all connections have the same packet length, and the transmission time of one packet takes one time unit. Hence, connection 2 and connection 3 reserve 20% of the link bandwidth and connection 1 reserves 50% of the link bandwidth. The arrival traffic patterns for these three connections are shown in the first three lines of the figure. In this figure, connection 2 and connection 3 transmit packets at higher rates than reserved, and connection 1 transmit packet according to its specified traffic pattern. The fourth line shows the service order under the First Come First Serve (FCFS) packet scheduling algorithm. In this case, even connection 1 reserves more resources; the misbehavior of connection 2 and 3 will affect its performance.

The VC packet scheduling algorithm assigns each packet a virtual finishing time based on the arrival pattern and the reservation of the connection to which the packet belongs. The fifth line shows the virtual finishing times. The service order of the packet under the VC packet scheduling algorithm is shown in the sixth line. Even connection 2 and 3 are transmitting packets at higher rates, the VC packet scheduling algorithm ensures that each well-behaving connection (for connection 1) get good performance. The VC packet scheduling algorithm has a drawback that a connection may be penalized for utilizing spare bandwidth that it received when other connections were idle [21,24]. The next packet scheduling algorithm will address this problem.

3.2 Weighted Fair Queueing

The WFQ packet scheduling algorithm (known as Packet-by-packet Generalized Processor Sharing) is an approach to approximate the Generalized Processor Sharing (GPS) policy [20-22]. With GPS, there is a separate FIFO queue for each connection sharing the same output link. A GPS server is characterized by N positive real number, $\phi_1, \phi_2, ..., \phi_n$, each corresponding to one queue. At time t, the service share for a non-empty queue f is $\frac{\phi_f}{\sum_{j \in B^i(t)} \phi_j}$, where $B^i(t)$ is the set of backlogged connections at time t. Thus, the bandwidth allocates to all connections are proportional to their service shares. However, GPS is impractical to implement since it does not transmit packets as entities but rather requires bit-by-bit multiplexing. To address this problem, WFQ have been considered.

WFQ approximates GPS in the same way that VC approximates TDM. Packets are assigned virtual finishing times corresponding with the time they would complete transmission in a GPS system. In this paper, we assume the Rate Proportional Processor Sharing (RPPS) network [3], that is, $\phi_f = r_f$. Hence, on packet p_f^k arrival, the PGPS server i will assign two tags: starting tag $S^i(p_f^k)$

and finishing tag $F^i(p_f^k)$, computed as follows:

$$F^i(p_f^0)=0$$

$$\frac{dv^i(t)}{dt}=\frac{C^i}{\sum_{j\in B^i(t)} r_j} \quad \cdots\cdots (3.2)$$

$$S^i(p_f^k)=\max\{v^i(A^i(p_f^k)),F^i(p_f^{k-1})\},k\geq 1 \quad \cdots\cdots (3.3)$$

$$F^i(p_f^k)=S^i(p_f^k)+\frac{l_f^k}{r_f},k\geq 1 \quad \cdots\cdots (3.4)$$

The server will service packets in increasing order of their finishing tags. In [21,22], the end-to-end delay guarantee was presented under the RPPS network. In Fig. 3.2, an example (is given by ref. [21]) shows the difference between WFQ and VC. There are two connections; each with a reserved rate of 0.5 packet/sec. Assume all packets are fixed size and channel capacity is 1 packet/sec. At time interval $[0,900)$, packets from connection 1 arrive at a rate of 1 packet/sec and none from connection 2. At time 900, 450 packets of connection 2 arrive at a rate of 1 packet/sec. According to the packet scheduling algorithms, the order of the packet being serviced are described in Figure 1. After time 900, all packets of the connection 2 will be served before any packets of connection 1 in VC server. For WFQ server, packets will be served interleaving. The different behaviors of VC and WFQ are due to that, in VC server, the delay of a packet depends on the arrival history of the connections. When a connection is misbehaved, it will be punished for some time. In WFQ, the virtual time of the packets depend on how many connections are backlogged in the server. The dependencies on how many connections are backlogged in the server that introduces extra complexities for WFQ. Since the system needs to emulate GPS and keep track of the number have backlogged connection at any moment in GPS. To reduce the complexity of computing virtual times, next packet scheduling algorithm, SCFQ, address this problem.

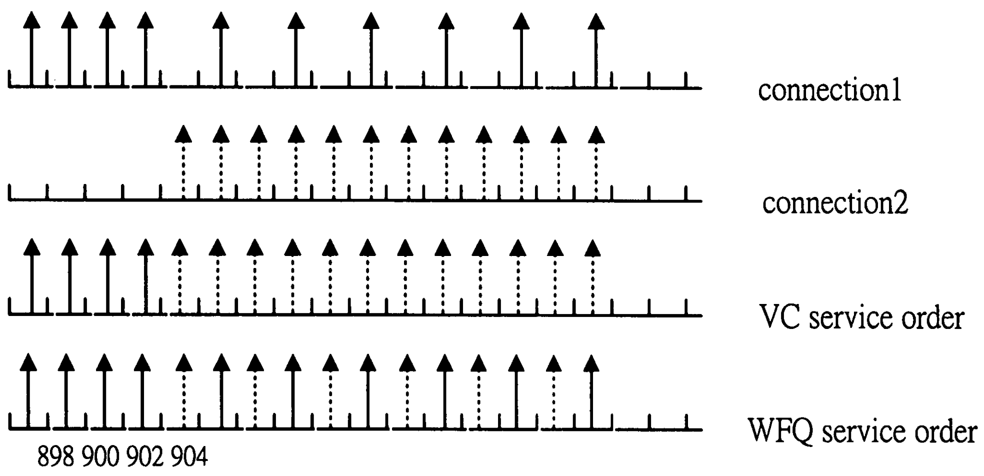

Figure 3.2 Illustrations of the VC and WFQ

3.3 Self-Clocked Fair Queueing

The SCFQ proposed in [23], was designed to simplify the PGPS. The SCFQ packet scheduling algorithm instead computes a sorting criteria that uses using the progress of its own scheduler as a reference rather than that of a simulated GPS scheduler. This algorithm is based on the observation that the system's virtual time at any moment t may be estimated from the virtual service time of the packet currently being serviced. Hence, on packet arrival, the SCFQ server i will assign packet p_f^k with finishing tag $F^i(p_f^k)$, computed as:

$$F^i(p_f^0) = 0$$

$$F^i(p_f^k) = \max\{F^i(p_f^{k-1}), V^i(A^i(p_f^k))\} + \frac{l_f^k}{r_f}, k \ge 1 \cdots\cdots \quad (3.5)$$

The server virtual time, $v^i(t)$, is defined to be equal to the finishing tag of the packet being serviced at time t, and $v^i(t) = t$ when the server i is idle. References [23,27] have shown the end-to-end delay bound of SCFQ under the RPPS network.

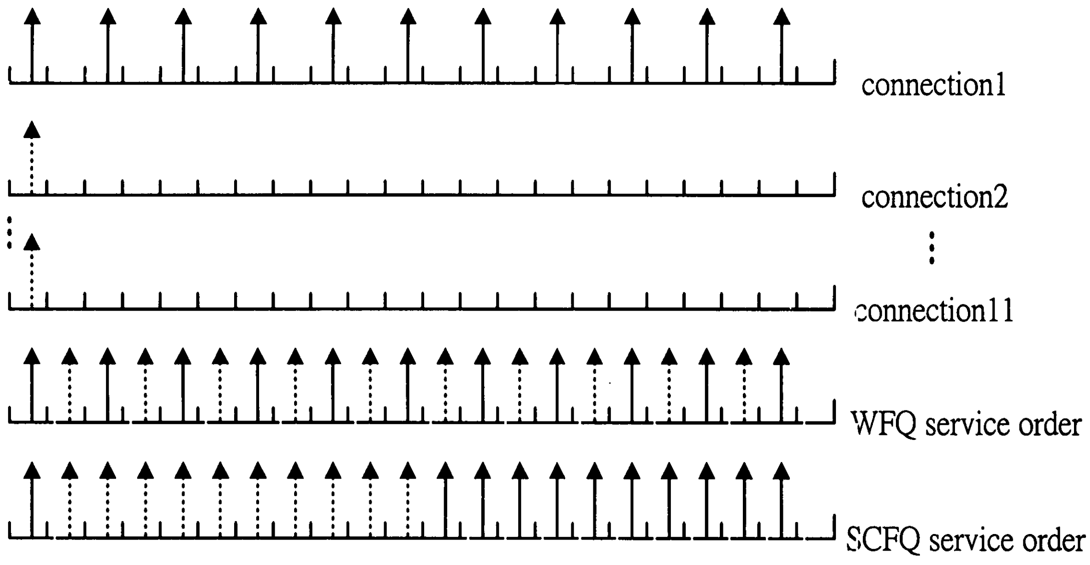

Figure 3.3 Illustrations of the WFQ and SCFQ

While the computation of the virtual time in simpler in SCFQ, the inaccuracy incurred can make SCFQ perform much worse than WFQ. In Fig. 3.3, we use an example (given in ref. [24]) to illustrate this inaccuracy. Assume all packets have the same packet length of 1, the link capacity is 1, the guaranteed service rate for connection 1 be 0.5, and the guaranteed service rate for connection 2-11 be 0.05. Under WFQ, the virtual finishing times will be $2k$ for packets p_1^k, $k=1,\cdots,10$, 20 for packets p_j^1, $j=2,\cdots,11$, and 21 for p_1^{11}, sending packets in order of virtual finishing times, WFQ will produce the service order as shown in the fourth line of Fig. 3.3. When SCFQ is used, at time 0, same as in WFQ, it is p_1^1 that has the smallest virtual finishing time, and it receives service first. At time 1, all packets p_i^1, $i=2,\cdots,11$, have virtual finishing time of $F_i^1=20$. We use the order for the number of connection as the service priority. The first packet from connection 2, p_2^1, is served. Since SCFQ uses the finishing time of the packet in service as the current virtual time, we have $F_2^1=20$. Therefore, when p_1^2 arrives at time 2, the virtual finishing time for this packet is set to be

$F_1^2 = \max\{2,20\} + 2 = 22$. Among all the packets ready to be served, p_1^2 has the largest finishing number. Finally, p_1^2 will not be service until all other ten $p_i^1, i = 2,\cdots,11$, packets finish services.

3.4 Start-time Fair Queueing

Traditionally, packet scheduling algorithms have been analyzed only for servers whose service rate is constant. However, service rate of flow-controlled, broadcast medium and wireless links may fluctuate over time. Fluctuation in service rate may also occur due to the variability in CPU capacity available for processing packets. In order to accommodate such scenarios, Start-time Fair Queueing (SFQ) packet scheduling algorithms is proposed [28].

In SFQ packet scheduling algorithm, each packet is assigned two tags, a starting tag and a finish tag. Unlike WFQ and SCFQ, packets are scheduled in increasing order of their starting tags at SFC server. Upon packet p_f^k arrival, the SFQ server i will assign two tags: the starting tag $S^i(p_f^k)$ and the finishing tag $F^i(p_f^k)$, computed as follows [28]:

$$F^i(p_f^0) = 0$$

$$S^i(p_f^k) = \max\{v^i(A^i(p_f^k)), F^i(p_f^{k-1})\}, k \geq 1 \quad \cdots\cdots (3.6)$$

$$F^i(p_f^k) = S^i(p_f^k) + \frac{l_f^k}{r_f}, k \geq 1 \quad \cdots\cdots (3.7)$$

The server virtual time, $v^i(t)$, is defined to be equal to the starting tag of the packet being serviced at time t. References [28] have shown the end-to-end delay bound of SFQ for variable rate Fluctuation Constrained and Exponentially Bounded Fluctuation servers.

SFQ is suitable for integrated service networks since

- It achieves low average as well as maximum delay for low-throughput applications.

- It provides fairness, regardless of variation in server capacity.
- It enables hierarchical link sharing.
- It is computationally efficient.

3.5 MSFQ packet scheduling algorithm

In this section, we will describe MSFQ packet scheduling algorithm. We assume that (a) there is a separate queue for each connection sharing the same output link. (b) The queue size of each connection is large enough such that there is no buffer overflow. (We can determine the queue size for each connection based on the delay of each connection.) (c) Each switch will reserve rate r_f for each connection f. (d) For network stability, we let $\sum_{f \in B^i(t)} r_f \le C^i$ at each server i. That is, the sum of the reserved rate of all backlogged connections is less than the server capacity. The MSFQ operates in the following way:

(1) On packet arrival, packet p_f^k is stamped with finishing tag $F^i(p_f^k)$ and starting tag $S^i(p_f^k)$ at server i, computed as:

$$F^i(p_f^0) = 0$$

$$v^i(t) = \min_{j \in B^i(t) \wedge j \neq f} S_j^i(t) \quad \cdots\cdots (3.8)$$

$$S^i(p_f^k) = \max\{v^i(A^i(p_f^k)), F^i(p_f^{k-1})\}, k \ge 1 \quad \cdots\cdots (3.9)$$

$$F^i(p_f^k) = S^i(p_f^k) + \frac{l_f^k}{r_f}, k \ge 1 \quad \cdots\cdots (3.10)$$

(2) Packets are serviced in the increasing order of finishing tag.

MSFQ is almost the same as PGPS, except for the assignment of virtual time. This difference makes MSFQ much easier implement than PGPS. However MSFQ still can provide the same end-to-end delay bound and fairness PGPS does [29].

3.6 Computation of the End-to-end Delay Bound

Two methods can be use to provide end-to-end delay bounds on a per connection basis in a packet switched network environment. One solution is to obtain worst case delay bounds at each switching node independently and use the sum of the local delay bounds at each switching node as the end-to-end delay bound [3-5, 30]. The other solution can be obtained by taking into account the dependencies in the successive switches that a connection traverses [21-24]. For the first method, traffic needs to be characterized on a per connection basis at each switching node inside the network to derive local delay bound. For the second method, end-to-end delay bound is derived based on the source traffic characterization. In general, the traffic needs to be characterized on a per connection basis at each switching node inside the network.

Even if the traffic of a connection can be characterized at the entrance to the network. Traffic pattern may be distorted inside the network to make the source characterization not applicable at each switching node traversed by the connection. It can be described by the following two reasons: (1) the traffic pattern of a connection can be distorted due to network load fluctuation, (2) the distortion will make the traffic bustier and cause instantaneously higher data transmission rate. This distortion can be accumulated, and downstream switching nodes will face burstier traffic than upstream switching nodes. Therefore, there are three solutions to address the problems of traffic distortion [24].

The first is to control the traffic distortion within the network [30]. For the controlling traffic distortions within the network, some packets need to be queued when a switching node has the extra capacity. This implies non-work-conserving packet scheduling algorithms. With a non-work-conserving packet scheduling algorithm, the switching node may be idle even when there are some packets waiting to be transmitted. For the two reasons that the non-work-conserving packet scheduling algorithm were seldom studied in the past. First, most of previous performance analyses emphasis the average delay of all packets and average throughput of the switching node. With a non-work-conserving packet scheduling algorithm, a packet may be queued in the switching node even when the switching node is idle. This will increase the average delay of packets and decrease the average throughput of the switching node. Secondly, in guaranteed QOS service, the more important performance index is the end-to-end delay bound rather than the average delay. Delay needs to be bounded in an end-to-end situation rather than just in a single node. All of the non-work-conserving packet scheduling algorithms can be expressed as a general class of packet scheduling algorithms called rate-controlled service discipline (RCSP) [30]. RCSP consists of two components: a rate-controller and a scheduler. The rate controller that consists of a number of regulators shapes the incoming traffic. The scheduler multiplexes eligible packets coming from different regulators. By having different combinations of regulators and schedulers, a general class of packet scheduling algorithms can be obtained. Furthermore, by appropriately setting parameters for regulators and local delay bounds at schedulers, RCSP can provide end-to-end delay bounds [31,32].

The second is to account for traffic distortions during scheduling [5]. Instead of scheduling packets according to their actual arrival times, the switching node assigns each packet an *expected* arrival time based on its traffic characterization and arrival history, and schedules packets based on their expected arrival times. PGPS, SCFQ, WFQ, MSFQ, and VC use this approach. The packet scheduling algorithm and connection admission control policy ensure that the packet is guaranteed to leave before the deadline, or at most some times units after the expected arrival time of the packet.

The third is to characterize the traffic inside the network [22]. This approach employs a two steps technique. In the first step, analysis of a single node technique is developed to characterize the output traffic of a switching node given the characterizations of all its input traffic. In the second step, starting from the characterizations of all the traffic of the source connection, an iterative process push the traffic characterizations from the links at the edge of the network to those inside the network. However, this method has several limitations: (1) characterizing the traffic inside the network is difficult and may not always be possible [22]. (2) It only applies to networks with constant delay links [4]. (3) The characterization represents a burstier traffic inside the network than that at the entrance, and is independent of the traffic model of the source connection [2].

Chapter 4 Comparison of scheduling algorithms

In this section we compared 5 rate-based packet scheduling algorithms focusing on end-to-end delay bound, fairness, and implementation time complexity.

4.1 Comparison of end-to-end delay bound

We compare the end-to-end delay bound provided by these rate-based scheduling algorithms under the condition that all of the source traffic conforms to a leaky bucket with parameter (σ_f, r_f).

For PGPS, VC, and MSFQ, the end-to-end delay bound of connection f is given by

$$d_f^k \le \frac{\sigma_f + (M-1)l_f^{\max}}{r_f} + \sum_{i=1}^{i=M} \frac{l^{\max}}{C^i} \quad \cdots\cdots (4.1)$$

For SCFQ, the end-to-end delay bound of connection f is given by

$$d_f^k \le \frac{\sigma_f + (M-1)l_f^{\max}}{r_f} + \sum_{i=1}^{i=M} \sum_{m \in B^i(t) \wedge m \neq f} \frac{l_m^{\max}}{C^i} \quad \cdots\cdots (4.2)$$

For SFQ, the end-to-end delay bound of connection f is given by

$$d_f^k \le \frac{\sigma_f - l_f^k}{r_f} + \sum_{i=1}^{i=M} \sum_{m \in B^i(t)} \frac{l_m^{\max}}{C^i} \quad \cdots\cdots (4.3)$$

In table 1, we use an example to explain the effects of different delay bounds (The end-to-end delay bounds for MSFQ, VC, and PGPS are the same, we use MSFQ to denote these three scheduling algorithms.). We assume an ATM network environment, of channel capacity C = 150 Mb/sec, packet length l = 424 bits, reserved rate r_f 1.5 MB or 30 MB, the number of backlogged connections V = 37 or 360, the number of hops that connection *f* travel, K = 10, to compare the results of different conditions. From row 1, when the application is low throughput and the number of connections is small, SFQ is the best, which has been proved in [28]. When the number of connections increases to 360, the delay bound for MSFQ is still not changed and the delay bound for SCFQ and SFQ is larger. For high throughput application, the advantage of MSFQ becomes clear. The reason for this phenomenon is obtained from equations (4.1)-(4.3). The delay bound of SFQ and SCFQ is related to the number of backlogged connections at server *i*. As the number of backlogged connections increases, the end-to-end delay bound increases. Besides, this factor will make admission control more difficult in high-speed networks. When the connection established, the admission control procedure must know the number of backlogged connections, however it will change in short time, resulting an incorrect delay bound. If we let V be the number of maximum allowed connections, it will underutilize the network bandwidth. Because the same end-to-end delay bound, you must reserve a more large service rate. Therefore, we conclude that SFQ can be applied for low throughput application, but for the general environment, the MSFQ is a more suitable candidate.

	MSFQ	SCFQ	SFQ
$\frac{r_f = 1.5}{V = 37}$	$\frac{\sigma_f}{1.5} + 2572$	$\frac{\sigma_f}{1.5} + 3562$	$\frac{\sigma_f}{1.5} + 763$
$\frac{r_f = 1.5}{V = 360}$	$\frac{\sigma_f}{1.5} + 2572$	$\frac{\sigma_f}{1.5} + 12720$	$\frac{\sigma_f}{1.5} + 9893$
$\frac{r_f = 30}{V = 37}$	$\frac{\sigma_f}{30} + 156$	$\frac{\sigma_f}{30} + 1145$	$\frac{\sigma_f}{30} + 1032$
$\frac{r_f = 30}{V = 360}$	$\frac{\sigma_f}{30} + 156$	$\frac{\sigma_f}{30} + 10303$	$\frac{\sigma_f}{30} + 10162$

Table 1 The Comparison of delay bound under different conditions

4.2 Comparison of fairness

We calculate $\left|\frac{W_f(t_1,t_2)}{r_f} - \frac{W_m(t_1,t_2)}{r_m}\right|$ through any interval $[t_1,t_2]$ in which both connections f and m are backlogged for rate-based packet scheduling algorithms.

For PGPS, fairness is given by

$$\max\{U_m + \frac{l_m^{\max}}{r_m}, U_f + \frac{l_f^{\max}}{r_f}, \frac{l^{\max}}{r_m} + \frac{l_m^{\max}}{r_m}, \frac{l^{\max}}{r_m} + \frac{l_f^{\max}}{r_f}\}$$

$$U_i = \min\{(V-1)\frac{l^{\max}}{C}, \max_{i\in B(t)}(\frac{l_i}{r_i})\}$$

Where V is the number of backlogged connections.

For MSFQ, fairness is given by

$$\max\{\frac{l_f^{\max}}{r_f}, \frac{l_m^{\max}}{r_m}, \max_{i\in B(t)}(\frac{l_i}{r_i}) + \frac{l_f^{\max}}{r_f}, \max_{i\in B(t)}(\frac{l_i}{r_i}) + \frac{l_m^{\max}}{r_m}, \frac{l_f^{\max}}{r_f} + \frac{l_m^{\max}}{r_m}\}$$

For SCFQ and SFQ fairness is given by

$$\frac{l_f^{\max}}{r_f} + \frac{l_m^{\max}}{r_m}$$

For VC, fairness is given by

$$\infty$$

4.3 Comparison of implementation time complexity

In Chapter3, we find that the distinction among various rate-based scheduling algorithms is the assignment of virtual time. Obviously, this will influence the time complexity for implementation. From the equation (3.1)-(3.10), we can recognize clearly for the computation of finishing tag, *VC = PGPS = SCFQ = SFQ = MSFQ = O(logN)* and for the computation of virtual time, *VC = SCFQ = SFQ = O(1) < MSFQ = O(logN) < PGPS = O(N)*. Hence, consider the implementation time complexity, *VC = SCFQ = SFQ < MSFQ < PGPS*.

The selection of rate-based scheduling algorithms is a tradeoff depended on end- to-end delay bound, fairness, implementation time complexity, and application requirement. In reference [28], they proposed an example to show there is no optimal rate-based scheduling algorithm. For real-time communication in high-speed networks, MSFQ seems to be a better choice.

Chapter 5 Conclusions

Quality of service guarantees of future network application in packet switched networks imposes stringent performance requirement on the network. This paper has examined aspects of providing deterministic QOS guarantees to application in packet switched network. We review the following results.

- We reviewed the traffic models for guaranteed service.
- We reviewed the traffic scheduling algorithms for guaranteed service.
- We compared five packet scheduling algorithms including VC, WFQ,

SCFQ, SFQ and MSFQ focusing on end-to-end delay bound, fairness, and implementation time complexity. The results of comparison make a design tradeoff depending on end-to-end delay bound, fairness, implementation time complexity, and application requirement. For real-time communication in packet switched networks, MSFQ seems to be a better choice.

We briefly outline directions for future research. Except for the rate-based packet scheduling algorithms, there is another class called delay-based packet scheduling algorithms. Delay guarantees for rate-based packet scheduling algorithms are only available for a restricted set of traffic models. However, the delay guarantees for delay-based packet scheduling are obtainable for all traffic models [20]. Generalizing the rate-based packet scheduling algorithm such that rate-based packet scheduling algorithms can provide delay guarantees under any traffic model is an interesting problem.

References

[1]. E. W. Knightly, D. E. Wrege, J Liebeherr, and H. Zhang, Fundamental limits and tradeoffs of providing deterministic guarantees to VBR video traffic, Proc. of ACM SIGMETRICS, pp. 98-107, May, 1995.

[2]. R. L. Cruz, A Calculus for network delay, part I: network elements in isolation, IEEE Trans. on Information Theory, Vol. 36, No. 1, pp. 114-131, Jan., 1991.

[3]. D. Ferrari, Client requirement for real-time communication services, IEEE Comm. Mag., 28(11), pp.65-72, Nov., 1990.

[4]. D. Ferrari, Real time communication in an inter-network, Journal of High Speed Networks, Vol. 1, No. 1, pp. 79-103, 1992.

[5]. D. Ferrari, and D. Verma, A scheme for real time channel establishment in wild area networks, IEEE Journal on Selected Area in Comm., Vol. 8, No. 3, pp.368-379, Apr., 1990.

[6]. D. Ferrari, A. Banerjea, and H. Zhang, Network support for multimedia: a

discussion of the Tenet approach, Computer Network and ISDN Systems, Vol.26, No. 10, pp. 1167-1180, Jul., 1994.

[7]. E. W. Knightly, Traffic models and admission control for integrated services networks, PhD thesis, University of California at Berkeley, May, 1996.

[8]. M. Krunz and H. Hughes, A traffic model for MPEG-Coded VBR streams, Proc. Of ACM SIGMETRICS, pp. 47-55, May, 1995.

[9]. D. Le Gall, MPEG: a video compression standard for multimedia application, Communication of ACM, Vol. 34, No. 4, pp. 46-58, Apr., 1991.

[10]. S. J. Golestani, A framing strategy for congestion management, IEEE Journal on Selected Area in Comm., Vol. 9, No. 7, pp. 1064-1077, Sep., 1991.

[11]. E. P. Rathgeb, Policing of realistic VBR video traffic in an ATM network, International Journal of Digital and Analog Comm. System, Vol. 6, pp. 213-226, 1993.

[12]. ATM Forum, ATM forum traffic management specification version 4.0, contribution 95-0013R11, Mar., 1996.

[13]. J. Turner, " New directions in communications (or which way to the information age?) ", IEEE Comm. Mag., Vol. 24, pp.8-15, Oct., 1986.

[14]. E. Knightly, D. Wrege, J. Liebeherr and H. Zhang, " Fundamental limits and tradeoffs for providing deterministic guarantees to VBR video traffic ", IEEE/ACM Trans. on Networking, Vol. 4, No.3, pp.352-362, June, 1996.

[15]. R. Reibman and A. W. Berger, Traffic descriptors for VBR video teleconferencing over ATM networks, IEEE/ACM Trans. on Networking, Vol. 3, No. 3, pp. 329-339, Jun., 1995.

[16]. S. Keshav, Congestion control for computer networks, PhD thesis, Univ. of California at Berbekey, 1991.

[17]. L. Kleinrock, Queueing systems, John Wiley and Sons, 1975.

[18]. L. Kleinrock, Queueing systems, Vol. 2: computer applications, Wiley, 1976.

[19]. L. Zhang, " VirtualClock: a new traffic control algorithm for packet switching networks ", ACM Trans. on Comp. Systems, Vol. 9, No.1, pp.3-26, 1990.

[20]. Demers. S. Keshav, and S. Shenker, " Analysis and simulation of fair queueing algorithm ", Journal of Internetworking Research and Experience,

pp.3-26, Oct., 1990.

[21]. A. Parekh and R. G. Gallager, " A generalized processor sharing approach to flow control in integrated services networks: The single-node case ", IEEE/ACM Trans. on Networking, Vol. 2, No. 2, pp.344-357, June, 1993.

[22]. A. Parekh and R. G. Gallager, " A generalized processor sharing approach to flow control in integrated services networks: The multiple-node case, IEEE/ACM Trans. on Networking ", Vol. 2, No. 2, pp.137-150, Apr., 1994.

[23]. S. Golestani, " A self-clocked fair queueing scheme for broadband application ", Proceedings of IEEE INFOCOM'94, pp.636-646, Toronto, ~ACA~Z, Jun., 1994.

[24]. H. Zhang, " Service disciplines for guaranteed performance service in packet-switching networks ", Proceeding of IEEE, 83(10): pp.1374-1399, Oct., 1995.

[25]. G. G. Xie and S. S. Lam, " Delay guarantee of Virtual Clock server ", IEEE/ACM Trans. on Networking, Vol. 3, No. 6, pp.683-689, Dec., 1995.

[26]. N. R. Figueira and J. Pasquale, An upper bound on delay for the VirtualClock service discipline, IEEE/ACM Trans. on Networking, Vol.3, No. 4, Aug., 1995.

[27]. S. Golestani, " Network delay analysis of a class of fair queueing algorithms ", IEEE Journal on Selected Areas in Comm., Vol. 13, pp.1057-1070, Aug. 1995.

[28]. P. Goyal, S. Lam and H. Vin, " Start-time Fair Queueing: a scheduling algorithm for integrated services packet switching networks ", Proceedings of SIGCOMM, 1996.

[29]. Y. P. Chu and E. H. Hwang, A new packet scheduling algorithm: minimum starting-tag fair queuing, IEICE Transactions on Communications, vol. E80-B, no.10 October 1997.

[30]. H. Zhang and D. Ferrari, Rate-controlled service disciplines, Journal of High Speed Networks, 3(4), PP.389-412, 1994.

[31]. L. Georgiadis, R. Guerin, and V. Peris, and K. N. Sivarajan, "Efficient network QOS provisioning based on per node traffic shaping", IEEE/ACM Trans. on Networking, Vol.4, No.4, pp.482-501, Aug. 1996.

封包交換網路排程演算法之研究

李乾麟　黃一泓　王鵬程

摘　要

網路技術之發展，已進入高速封包交換時代，高速網路需能提供各種不同的應用所需之服務品質，尤其如：視訊、音訊……等多媒體封包資訊需求，更爲殷切。

網路上每一交換節點欲保證其服務的品質，封包排程演算法的設計扮演極重要的角色。在封包交換網路上，每一交換節點的封包是來自於不同的連線，且各連線彼此交互影響，若無適當的管制，將影響每一連線的傳輸效能。

封包排程演算法控制每一封包被服務的順序，且決定每一連線間的相互關係；本研究報告參照多種排程演算法與交通模式(Traffic model)之相關文獻，研究其演算法與交通模式之特質，且對這些封包排程演算法的連線延遲上限(End-to-end delay bound)、服務的公平性(Fairness)及實作的複雜度(Implementation time complexity)提出評估比較，期使學者先進能在這些主題上提出更深入的研究與探討。

關鍵字：服務品質保證、封包排程演算法、交通模式

李乾麟、王鵬程：資訊管理系講師
黃一泓：資訊管理系助理教授