香港浸會大學
HONG KONG BAPTIST UNIVERSITY

**FACULTY OF SCIENCE**
**DEPARTMENT OF COMPUTING STUDIES**
**TECHNICAL REPORT (JNG05-97)** **May, 1997**

**Title:**

# Integrated Delay Analysis of Regulated ATM Switch for Real-Time Applications[α]

**Authors:**

Joseph Kee-Yin, NG (Dept. of Computing Studies, HKBU)
Shibin, SONG (Dept. of Computing Studies, HKBU)
Wei, ZHAO (Dept. of Computer Science, CityU)

**ABSTRACT:**

In this report, we present an efficient and effective method to derive the worst case delay in an ATM switch. In an ATM switch, admitting a hard real-time connection requires the delays of cells belonging to the connection meeting their deadline without violating the guarantees already provided to connections that are currently active. Effective traffic and service description plays a major role in delay analysis. Previous studies have shown that the real-time connection traffic and the available service can both be described by piece-wise linear functions in terms of time. These functions provide adequate information about the worst case traffic behavior of connections as well as the minimum service available for the transmission of cells in an ATM switch. By innovatively utilizing the inverse of the arrival and service functions, we obtain an efficient and effective method to compute the worst case delay of a connection to an ATM switch. We analyze and compare the performance of an ATM switch with priority driven and FIFO scheduling policies under different utilization. We also compare the performance using our proposed "integrated" method with the traditional "independent" method. From simulation experiments, we found out that our method always obtain a higher admission probability and a better estimation of cell delay within an ATM switch.

**Keywords:**

Real-Time Networks, ATM switch, Worst case delay, Delay analysis. Hard real-time connection

---

[α] This is a joint-research project between Hong Kong Baptist University and the City University of Hong Kong. This work is supported by the Faculty Research Grant of Hong Kong Baptist University (FRG/96-97/II-52)

# TABLE OF CONTENTS

# Glossary

| Term | Definition | Location |
|---|---|---|
| $d^{independent}$ | The worst case delay of a connection by independent method. | 5.1 |
| $d^{integrated}$ | The worst case delay of a connection by integrated method. | 5.1 |
| $e$ | The relative error, $e = (d^{independent} - d^{integrated}) / d^{integrated}$. | 5.1 |
| f(c) | The inverse function of F(I). | 3 |
| $F(I)$ | The maximum arrival function of all the connections. | 4.2 |
| $F_i(I)$ | The maximum arrival function of connection $M_i$. | 2.1 |
| $F_i(t,I)$ | The number of cells arrived from connection $M_i$ in [t,t+I). | 2.1 |
| G(I) | $G(I) = \rho_i(I - F(I_i)) + S_i(F(I_i))$. | 4.2 |
| $I_i$ | $I_i = \beta_i / (1 - \rho_i)$. | 2.1 |
| L(I) | The maximum number of cells that may departure from a leaky bucket during time interval of length I. | 2.3 |
| $\{M_1,\ldots,M_{n-1}\}$ | The set of n-1 hard real-time connections that already admitted by the network. | 2.1 |
| s(c) | The inverse function of S(I). | 3 |
| $S_i(I)$ | The minimum available service function for connection $M_i$. | 3 |
| $S_i(t,I)$ | The number of $M_i$ cells that can be served by a server during time interval [t,t+I). | 3 |
| $S_i^{H.P.}(I)$ | The maximum service time spent for the connections which priority is higher than the priority of connection $M_i$ in any time interval of length I. | Appendix A |
| $S_i^{L.B.}(I)$ | The minimum available service function for connection $M_i$ at the leaky bucket regulator . | Appendix A |
| $S_i^{P.D.}(I)$ | The minimum available service function for connection $M_i$ at the priority driven scheduler. | Appendix A |
| S(X) | A piece-wise linear function. | 3 |

# 1. Introduction

Advances in high-speed computing and networking technologies have opened up opportunity for new applications such as tele-medicine, video conferencing, video-on-demand, control systems in manufacturing, etc. Many of these systems are classified as distributed real-time systems. Tasks in these systems could be executed at different locations in different hosts. They communicate among each other by exchanging messages. These systems are also considered to be hard real-time because these applications have some stringent timing requirements. Hence, networks used to support such system must ensure a timely delivery of messages.

ATM networks are getting increasingly popular in supporting these hard real-time systems/applications. ATM is a connection-oriented packet-switched technology. Before two hosts communicate, a connection must be established between them. A new connection requires a certain quality of service (QOS) and is admitted by the network only if the network can guarantee the requested quality of service without violating the guarantees made to the connections that are currently active. In order to guarantee that the message deadlines are met for each particular connection, we must derive the worst case delay of cells belonging to the connection in an ATM switch. In this report, we present an efficient and effective method to derive such worst case cell delay.

As a high performance switching and transmission facility, an ATM switch is complicated. A typical approach of delay analysis is to decompose a switch into several components. Each component is then analyzed *independently*. The worst case delay of a cell is then the sum of the worst case delays of components it traverses. The advantage of this independent method is its simplicity. However, it is usually pessimistic because if a cell suffers the worst case delay at one component, the cell may not suffer the worst case delay at the next component.

In this report, we design and analyze an integrated method that will analyze all the involved components in an *integrated* manner and then derive the total delay directly. This method can produce a tighter delay bound than the independent method, hence improving the system efficiency. However, an integrated delay analysis may be too complicated to use in on-line connection admission because of its complexity. To address this problem, we utilize effective traffic and service description functions. Previous studies have shown that the real-time connection traffic and the available service can both be described by piece-wise linear functions in terms of time. These functions provide adequate information about the worst case traffic behavior of connections as well as the minimum service available for the transmission of cells in an ATM switch. We derive the worst case delay based on the inverse functions of the traffic arrival and available service functions. As we will demonstrate, this dramatically reduces the computation complexity while still producing tight delay bounds.

Our work compliments the recent progress in real-time communications [HAN95a,HAN95b,LEE96, HAN97,SAH94,MUK93]. In particular, much of the previous work in ATM has been concentrating on obtaining the delay bounds and connection admission criteria for individual scheduling polices [CHA94,CRU91,CRU95, KNI95,RAH95,RAH96,WRE96,]. Ferrari and Verma [FER90] and Zheng and Shin [ZHE96] studied the use of Earliest Deadline First scheduling in wide area networks. Zhang and Ferrari [ZHA94] discuss how local deterministic delay bounds can be guaranteed over an ATM link for bursty traffic, even when the sum of peak rates of all the connections exceeds one. Deterministic delay bounds in networks have also been studied by Yates, Kurose and Towsley in [YAT93] and by Cruz in [CRU91,CRU95].

Analyzing delay in an integrated manner has been addressed by [CRU95,GEO96]. While Cruz provided a general framework for integrated analysis, Georgiadis, Guerin, Peris, and Sivarajan concentrated on a network consisting of rate-controlled services. Our work differs from the previous work in that 1) we study ATM switches with an output port scheduler plus a leaky bucket regulator, where the scheduler can have either FIFO or priority driven scheduling policy. Most existing ATM switches are of this kind of settings; 2) by using the inverse functions of arrival and service functions, we obtain a simpler method to compute of the worst case delay of a connection in an ATM switch. At the same time, we formally state the requirement with which the analysis based on traffic and service functions is valid. By clearly understanding the domain of this technique, one will be able to apply it to many other real-time scheduling problems; 3) we thoroughly analyze and compare the performance of our integrated method and the traditional independent method. As we will show, our proposed integrated method always obtain a higher admission probability and a better estimation of cell delay within an ATM switch.

The remainder of this report is organized as follows. In Section 2, we give a brief review for the hard real-time connections and present our ATM switch model. In Section 3 we formally define the requirement for a server and consider the worst case delay computation. Section 4 discusses about the integrated method in analyzing the worst case delay. We consider both the regulated priority driven server as well as the regulated FIFO server. Section 5 carries out the performance comparison between the two methods and Section 6 concludes the report with a summary and future work.

## 2. Connection and Network Models

### 2.1 Connections

ATM is a connection-oriented packet-switched technology. Before two hosts begin to communicate, a connection has to be set up between them. The term *connection* is then often to denote the stream of messages sent from a source host to a destination host. In ATM networks, messages from individual connections are segmented into fixed size packets called *cells*. Let $\tau$ be the transmission time of an ATM cell. In this report, we assume that time is normalized in terms of $\tau$ as one time unit. This simplifies our analysis without any loss of generality.

In this report, we deal with *hard real-time* connections. A hard real-time connection has a stringent deadline constraint on the delay of its cells. The problem of admitting a hard real-time connection is as follows: Consider that a network has already admitted a set of n-1 connections $\{M_1, M_2, \ldots, M_{n-1}\}$ and their deadline constraints are all met. Let the request for a new hard real-time connection $M_n$ arrive. Now the network must determine if $M_n$ can meet its deadline constraint without violating the guarantees already provided to connections $M_1, M_2, \ldots, M_{n-1}$.

For connection $M_i$, its deadline is $D_i$. Furthermore, we assume that in any time interval of length I, the maximum number of the cells from connection $M_i$ that can arrive at the entry of the network is upper-bounded by the following constraint function:

$$F_i(I) = \min(I, \beta_i + \rho_i I) = \begin{cases} I & 0 \le I \le I_i, \\ \beta_i + \rho_i I & I > I_i. \end{cases} \tag{2.1}$$

where

$$I_i = \frac{\beta_i}{1 - \rho_i}. \tag{2.2}$$

This kind of traffic constraint function has been used widely for network analysis [GEO96, CRU91]. Let $F_i(t, I)$ be the number of cells arrived from connection during time interval of [t, t+I). Then, for $t \geq 0$

$$F_i(t, I) \leq F_i(I). \tag{2.3}$$

Given this relation, $F_i(I)$ is called *the maximum arrival function* for connection $M_i$.

## 2.2 ATM Switch

As cells belonging to different connections traverse the network, they may share network resources such as communication links and ATM switches. It is the task of ATM switches to multiplex cells onto shared links. Figure 1 shows the general architecture of an ATM switch. Switch architecture directly impacts the network performance [NEW92,ROO94]. In this report, we consider an ATM switch to be an $n \times n$ output-buffered switch with an internal switching speed n times faster than that of the input and output links. This architecture has a better throughput-delay performance than various other architectures (e.g., input buffered or input smoothing switches) [HLU88] and is widely used in commercially available ATM LANs.

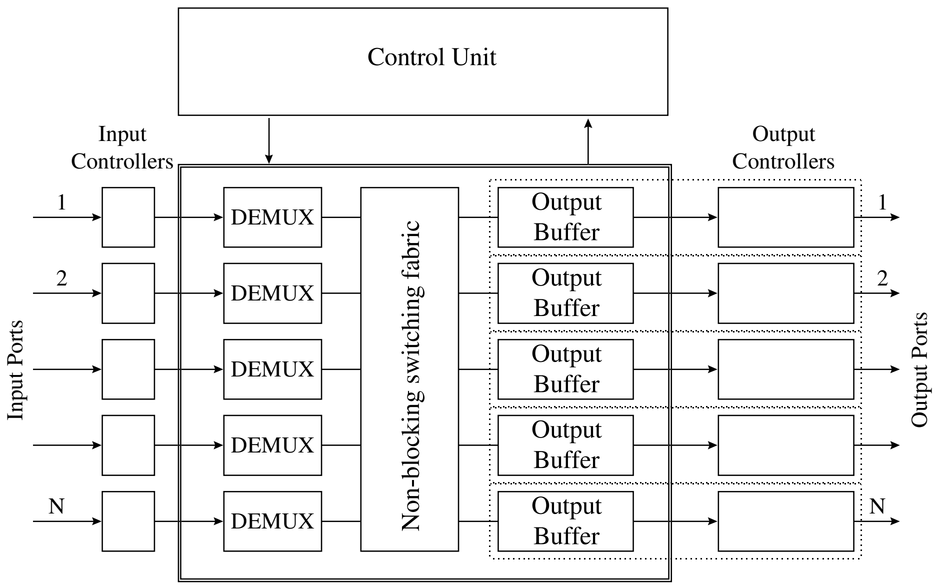



Figure 1. A General ATM Switch Architecture.

An ATM switch of this type consists of input port controllers, switching fabric, and output port controllers. Note that delays suffered by a cell at an input port controller and the switch fabric are *constant*. That is, these delays are the same for all the cells. However, the delay at the output port controller is different. Cells from different connections (if arriving from different input ports) are multiplexed there. The delay of a cell depends on the queue length at the output port and the scheduling mechanism utilized.

To simplify the delay analysis, we can subtract the appropriate constant delays encountered by a connection from its deadline. Consequently, we eliminate all the constant delays from consideration and focus only on the output port controllers. For the rest of this report, we assume that all the deadlines of connections have been modified under this assumption.

### 2.3 Output Port Controller

The scheduling policy at an output port controller of an ATM switch determines the order of the cells (from different connections) being transmitted. Typical scheduling policies utilized in the ATM switches are either FIFO or priority driven. For example, the Lattiscell switch by Bay Networks uses the FIFO policy while the ForeRunner switch from the Fore Co. adopts a priority driven scheduling algorithm.

Output port controllers can also be classified as *regulated* or *non-regulated*. For a regulated output port controller, cells from a particular connection will be regulated before being transmitted over the link. Regulating traffic limits the burstiness of the traffic, increases the stability of the network, and simplifies the delay analysis [CRU91,LIZ97]. Therefore, we focus ourselves on the regulated output port controllers.

A *leaky bucket* is a typical mechanism used for traffic regulation [CRU91]. It consists of a token bucket and a cell buffer. The cells from a connection associated with the leaky bucket are queued at the cell buffer. A waiting cell from the cell buffer is transmitted if at least one token is available in the token bucket and a token is consumed whenever a cell is transmitted. Tokens are stored at the token buffer with a maximum size of $\beta$ and tokens of a leaky bucket are generated automatically at a rate of $\rho$. In this report, we assume that the output port controllers are equipped with leaky bucket regulators.

Let us consider how the two parameters, $\beta$ and $\rho$, of a leaky bucket should be assigned. It is obvious that the maximum number of cells that may departure from a leaky bucket during any time interval of length I is upper-bounded by

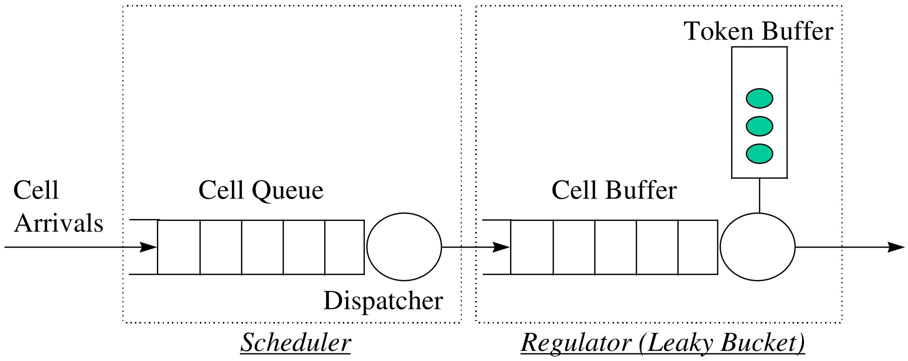
$$L(I) = \min(I, \beta + \rho I). \tag{2.4}$$

Since the main purpose of the leaky bucket is to eliminate the burstiness of the traffic introduced by multiplexing, $L(I)$ should be the same as $F_i(I) = \min(I, \beta_i + \rho_i I)$, if the leaky bucket serves connection $M_i$. This implies the following:

$$\beta = \beta_i \quad \text{and} \quad \rho = \rho_i. \tag{2.5}$$

Thus, we assume that $\beta$ and $\rho$ are assigned in accordance with (2.5) during the connection establishment time.

Another advantage of using regulated output controllers is that the departure traffic of a connection from a switch (consequently, the arrival traffic to the next switch, if any) is fixed and known when the regulator's parameters are determined. Hence, each switch can be analyzed independently. This simplifies the analysis of end-to-end delay of a connection. As a result, we only need to concentrate on analyzing a single switch. Within a switch, after eliminating the constant delay components we can view cells from a connection are served by two components at an output port controller: a scheduler and a regulator, as shown in Figure 2. We are thus interested in analyzing the worst case delay of a connection at an output port controller with such architecture.




The output port controller from the view point of the cells from a connection

Note that the scheduler is shared by all the connections that go through the output port controller while each connection has its own regulator.

Figure 2. Output Port Controller.

There are two kinds of methods to analyze the delay at an output port controller. An *independent method* will analyze the scheduler and the regulator separately and then compute the total delay at the output port as the sum of the delays at the scheduler and the regulator. Techniques for analyzing FIFO and priority driven schedulers and leaky bucket regulators have been developed [CRU91,CRU94,LIE96,PAR93]. Hence, the independent method is ready to use. However, the independent method assumes that a cell that suffers the worst case delay at the scheduler also suffers the worst case delay at the regulator. As a result, possible correlation between the scheduler and the regulator is ignored and the worst case delay may be over-estimated.

An *integrated method*, on the other hand, will analyze both components in an integrated manner and derive the total delay directly. We are going to develop the integrated methods. As we will show, our methods can take into account the relationship between the scheduler and the regulator and hence can accurately derive the worst case delay suffered by the cell of connection at an output port controller.

## 3. A General Framework for Analyzing the Worst Case Delays

Before we proceed to present our integrated delay analysis method, we need to introduce a framework that is useful for the derivation of the worst case delays. The framework is general in the sense that it not only helps the delay analysis we concern in this report but is also applicable to a large class of real-time scheduling problems.

A *server* here means an entity that controls the transmission of cells. That is, the FIFO scheduler is a server, so are the priority driven scheduler and the leaky bucket regulator. Furthermore, the output port controller itself is a (compound) server that consists of a scheduler and a regulator.

**Definition 3.1.** A server is *sequential* if for every connection it serves, cells from the connection are served in the order of their arrivals.

Note that the sequentiality is a weaker condition than requiring a server to be FIFO. An FIFO server serves cells from *all* the connections in the order of their arrivals while a sequential server only need to serve cells from *each* connection in the order of their arrivals. That is, if two cells come from the *same* connection, a sequential server *has* to serve them in the order of their arrivals. However, if these two cells come from *different* connections, a sequential server *may not* observe their arrival order while an FIFO server has to! Clearly, a non-sequential server is more difficult to implement and does not result in a better delay bound than the sequential one. All the ATM switches that are currently available provide sequential services. Hence, we assume that all the servers we concern in this report are sequential ones.

A *busy interval* of a server is a time interval during which the queue of the server is not empty. Obviously, we are only interested in busy intervals. An instant time $t$ is the *starting point* of a busy interval if at $t^-$ the queue of the server is empty and at $t$ it is not. For the sake of simplicity, any time interval mentioned in the rest of this report has its left boundary being a starting point of a busy interval.

Cells of a connection are delayed at a server because the server does not serve the connection in the full rate. For the FIFO scheduler and the priority driven scheduler, this may happen due to the fact that the server is shared with other connections. For the leaky bucket regulator, this is because of the non work-conserving nature of the regulator. For the sake of delay analysis, we need to quantify the service available to a connection from a server.

**Definition 3.2.** $S_i(t, I)$ is the available service function for connection $M_i$ during time interval between t and t + I. That is, $S_i(t, I)$ specifies the number of $M_i$ cells that can be served (i.e., transmitted) by a server during the time interval between t and t + I.

**Definition 3.3.** $S_i(I)$ is the minimum available service function for connection $M_i$. That is,

$$S_i(I) = \min_t [\, S_i(t, I) \,] \tag{3.1}$$

Note that $S_i(I)$ specifies the minimum service available for $M_i$. This does not necessarily mean that $M_i$ can always consume the available service because the exact service to $M_i$ also depends on the cells arrived from $M_i$. Nevertheless, in general, $S_i(I)$ depends on the scheduling policy of the server and the arrival functions of connections which share this server. We illustrate this by the following example.

Consider a priority driven scheduler that serves 2 connections. Their maximum arrival functions are given in the form of (2.1). Furthermore, assume that the priority of $M_1$ is higher than that of $M_2$. Given that $M_1$ is the connection with the highest priority, for $I \geq 0$,

$$S_1(I) = I. \tag{3.2}$$

That is, I cells can be transmitted for connection $M_1$ during a time interval of length I, if they have arrived. However, the case for $M_2$ is different. For an interval of length I, the total service available to both connections is I cells. Given the fact that the maximum number of cells arrived from $M_1$ is $F_1(I)$, the service available for $M_2$ is I - $F_1(I)$. That is,

(3.3) $$S_2(I) = I - F_1(I).$$

Substituting (2.1) into (3.3), we have

(3.4) $$S_2(I) = \begin{cases} 0 & 0 \le I \le \dfrac{\beta_1}{1-\rho_1}, \\ (1-\rho_1)I - \beta_1 & I > \dfrac{\beta_1}{1-\rho_1}. \end{cases}$$

The physical meaning of (3.4) is clear: consider a time interval [t, t + I). When I is less than $\beta_1/(1 - \rho_1)$, the cells from connection $M_1$ may arrive at rate 1. Hence, all the service is taken by $M_1$. Thus, the service available to $M_2$ is zero. After that, $M_1$ can only consume the service at rate $\rho_1$. Thus, the available service rate for $M_2$ is 1 - $\rho_1$. Figure 3 shows curves for both $F_2(I)$ and $S_2(I)$.

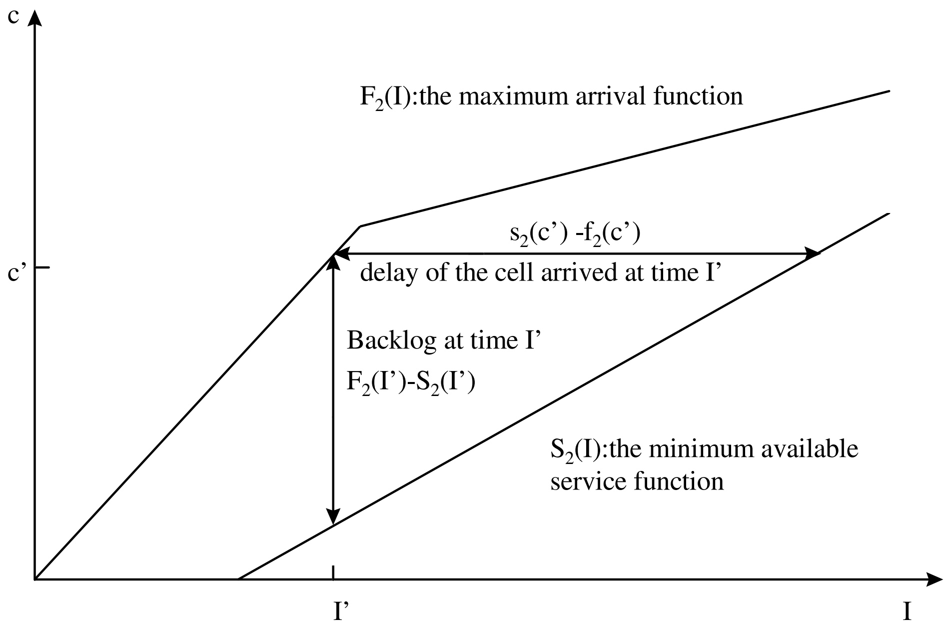



Figure 3. An Example of the Arrival and Service Functions for $M_2$.

**Definition 3.4.** Assume that t and t' are starting points of two busy intervals, respectively, and $F_i(t, I) \geq F_i(t', I)$. A server is *normal* if

(3.5) $$S_i(t, I) \geq S_i(t', I)$$

and

(3.6) $$F_i(t, I) - F_i(t', I) \geq S_i(t, I) - S_i(t', I).$$

(3.5) states that the available service provided by a server to a connection should be non-decreasing as the arrival increases. However, by (3.6) the amount of increased available service (i.e., $S_i(t, I) - S_i(t', I)$) should not be more than the amount of increased arrival (i.e., $F_i(t, I) - F_i(t', I)$). When both conditions are satisfied, we say that the server is normal. It is obvious that all the (basic and compound) servers we consider in this report are normal. However, there are abnormal servers. For example, consider a queue length sensitive server that stops serving a connection when its queue length is between

X and Y. It can be proven by constructing a counter-example that this sever is not normal. We encourage the reader to verify this fact in order to be familiar with this concept.

We are going to use extensively inverse functions to derive delays. Formally, an inverse of a function is defined only if the function itself is one-to-one. We, however, take a generalized definition. Let y = H(x) be a continuous function. h(y), the inverse of H(x) is defined as follows:

$$h(y) = \max(x \mid y = H(x)).$$

With these definitions, for the class of sequential and normal servers we have the following important results.

**Lemma 3.1.** Assume that a sequential and normal server serves a connection. Let S(I) and F(I) be the minimum service function and the maximum arrival function for the connection, respectively. The worst case delay of the cells from the connection is given by

$$d = \max_{c \geq 0} [s(c) - f(c)] \tag{3.7}$$

where s(c) and f(c) are the inverse functions of S(I) and F(I), respectively.

**Proof**. Consider a busy interval starting at time t. Given that the server is sequential and normal, it follows from (3.5) that the worst case delay may occur when the arrival is maximized, i.e., F(t, I) = F(I). Under this situation, the delay is maximized if the service provided is minimal, i.e.,, S(t, I) = S(I). Consider that a cell arrives at t + I. Then, the number of cells arrived including this one is c = F(I). Given that the server is sequential, this cell will be served at t + I' where c = S(I'). The delay of this cell is I' - I = s(c) - f(c). If we maximize it over c, then (3.7) follows. Q.E.D.

Expressions of the worst case delay in terms of the maximum arrival function and the minimum service function have also been derived in [CRU95,KNI95]. Our result differs from theirs in two aspects: we utilize the inverse functions which requires one step of maximization to obtain the delay while the previous work used the original functions and required two steps of maximization. Thus, our method is simpler. Furthermore, we formally stated the requirement of sequentiality and normality. This helps to clearly define the domain where this kind of analysis, the one based on arrival and service functions, is applicable. Hence, we can use this technique to deal with not only switched computer networks but also many other real-time scheduling problems.

For some specific forms of F(I) and S(I), (3.7) can be further simplified. In the rest of this section, we develop a simplified method for delay computation. As we will see, the function S() is often a piece-wise linear and non decreasing function. In particular, let S(X) be segmented at $X_0, X_1, \ldots, X_m$, where $0 = X_0 \leq X_1 \leq \ldots \leq X_m$ and $0 = S(X_0) \leq S(X_1) \leq \ldots \leq S(X_m)$. Then S() can be given in the following form:

$$S(X) = \begin{cases} 0 & X = 0, \\ \mu_k (X - X_k) + S(X_k) & 0 \leq k < m \text{ and } X_k < X \leq X_{k+1}, \\ \mu_m (X - X_m) + S(X_m) & X > X_m. \end{cases} \tag{3.8}$$

where for k = 0, 1, …, m, $\mu_k \geq 0$ is the slope of the linear segment starting at $X_k$.

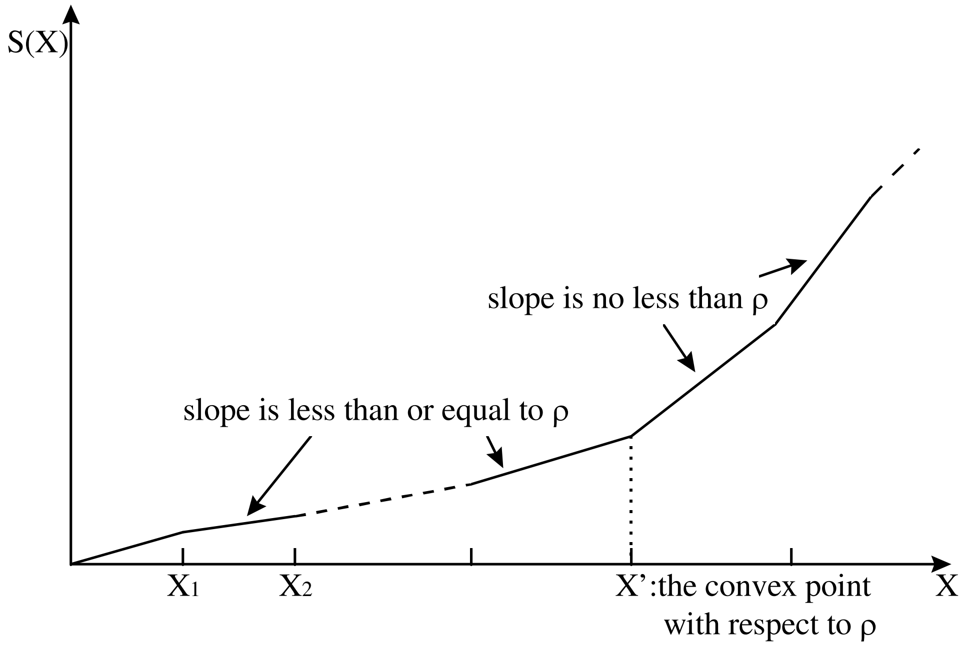



Figure 4. An Example of a Weak Convex Function.

**Definition 3.5.** A piece-wise linear function S(X) is *weakly convex with respect to* $\rho$ if there is X' such that

1) when $0 \leq X \leq X'$, the slope of S(X) is less than or equal to ρ, and

2) when $X > X'$, the slope of S(X) is no less than ρ.

X' is called a *convex point with respect to* $\rho$ --- as shown in Figure 4.

**Lemma 3.2.** Let S(X), the minimum available service function for a connection at a server, be in the form of (3.8). The maximum arrival function of the connection, F(X), is given as follows:

(3.9) $$F(X) = \begin{cases} X & X \leq X^*, \\ \rho X + \beta & t > X^*, \end{cases}$$

where

(3.10) $$X^* = \beta/(1 - \rho).$$

If the server is sequential and normal and S(X) is weakly convex with respect to ρ, then the worst case delay of the cells from the connection is given by

(3.11) $$d = \begin{cases} s(X^*) - X^* & \text{if } X^* > S(X'), \\ \dfrac{\beta}{\rho} + X' - \dfrac{S(X')}{\rho} & \text{otherwise;} \end{cases}$$

where s() is the inverse function of S(), X* is as defined in (3.10), and X' is a convex point with respect to ρ in S().

For the proof of this lemma, see Appendix A. As we will see, this lemma helps in the derivation of the worst case delays of the servers we concern in this report.

## 4. Integrated Delay Analysis

In this section, we analyze the worst case delay for the output port controller which consists of either a priority driven scheduler or a FIFO scheduler to multiplex cells from different connections and a leaky bucket regulator for each connection. We refer to the former compound server as a *regulated priority driven server* and the latter, a *regulated FIFO server*. Without loss of generality, for a regulated priority driven server, we assume that priorities are assigned in the same order as the connection indices. That is, $M_1$ has the highest priority and $M_n$ has the lowest priority.

### 4.1. Regulated Priority Driven Server

In this subsection, we analyze the worst case delay for the regulated output port control that uses a priority driven scheduler. A cell from a connection will first be multiplexed at the priority driven scheduler and then goes through a leaky bucket regulator.

The key step in the integrated analysis is to derive the minimum available service function so that Lemma 3.1 and Lemma 3.2 can be applied. For this purpose, we first establish the following lemma.

**Lemma 4.1** For the regulated priority driven server, the minimum available service function of connection $M_i$ is given by

$$\text{(4.1)} \qquad S_i(I) = \begin{cases} 0 & 0 \le I \le \theta_i, \\ \rho_i(I - \theta_i) & I > \theta_i; \end{cases}$$

where

$$\text{(4.2)} \qquad \theta_i = \frac{\sum_{k=1}^{i-1} \beta_k}{1 - \sum_{k=1}^{i-1} \rho_k}.$$

For a proof of this lemma, see Appendix A. Using both Lemmas 3.2 and 4.1, we can now prove the following theorem.

**Theorem 4.1.** For the regulated priority driven server, the worst case delay for the cells from connection $M_i$ is given by

$$\text{(4.3)} \qquad d_i = \theta_i + \frac{\beta_i}{\rho_i}$$

where $\theta_i$ is given in (4.2).

**Proof**. From (4.1) and Definition 3.5, $S_i(I)$, the minimum available service function for connection $M_i$, is weakly convex with respect to $\rho_i$ with $\theta_i$ being a weakly convex point. The inverse function of $S_i(I)$ is given by

$$\text{(4.4)} \qquad S_i^{-1}(c) = \frac{c}{\rho_i} + \theta_i.$$

By Lemma 3.2 and $I_i > S(\theta_i) = 0$,

$$\text{(4.5)} \qquad d_i = S_i^{-1}(I_i) - I_i = \frac{I_i}{\rho_i} + \theta_i - I_i = \theta_i + \frac{\beta_i}{\rho_i}$$

where by (2.2), $I_i = \frac{\beta_i}{1-\rho_i}$. Q.E.D.

(4.3) indicates that $d_i$ is made by two terms: the first term, $\theta_i = (\sum_{k=1}^{i-1} \beta_k)/(1-\sum_{k=1}^{i-1}\rho_k)$, reflects the impact of the higher priority connections; the second term, $\beta_i/\rho_i$, takes into account of the regulation effect. From the proof of Theorem 4.1, we see the advantage of Lemma 3.2 for the derivation of the worst case delay. Once the inverse function of the minimum available service has been obtained, it is relatively straightforward to obtain the worst case delay for individual connections.

## 4.2. Regulated FIFO Server

In this subsection, we analyze the worst case delay for the regulated output port control that uses a FIFO scheduler. A cell from a connection will first be multiplexed at the FIFO scheduler and then goes through a leaky bucket regulator. We thus refer this compound server as the *regulated FIFO server*. In this subsection, we assume that indices of connections are assigned such that

(4.6) $$I_1 \le I_2 \le \ldots I_{n-1} \le I_n.$$

For the convenience, we define $I_0 = 0$. This assumption, as we will see, helps to simplify the analysis of the regulated FIFO server without loss of generality.

Once again, the key step in the integrated analysis is to derive the minimum available service function, $S_i(I)$, so that Lemma 3.2 can be applied. However, $S_i(I)$ depends on some other functions which we should introduce first.

**Definition 4.1.** F(I) is the maximum arrival function of *all* the connections. That is,

(4.7) $$F(I) = \sum_{k=1}^{n} F_k(I).$$

**Lemma 4.2.** F(I) can be expressed in the piece-wise form as follows:

(4.8) $$F(I) = \begin{cases} 0 & I = 0, \\ \varphi_j \cdot (I - I_j) + F(I_j) & j = 0, 1, ..., n-1, I_j < I \le I_{j+1}, \\ \varphi_n \cdot (I - I_n) + F(I_n) & I > I_n; \end{cases}$$

where $\varphi_0 = n$ and for j = 1, 2, …, n,

(4.9) $$\varphi_j = (n - j) + \sum_{h=1}^{j} \rho_h .$$

**Proof**. We derive F(I) by an induction argument on the linear segments.

For the first segment, $0 < I \le I_1$, because

(4.10) $$F_1(I) = I,\ F_2(I) = I,\ \ldots, \text{and } F_n(I) = I,$$

(4.11) $$F(I) = \sum_{k=1}^{n} F_k(I) = n \times I = \varphi_0(I - 0) + F(0).$$

Assume that (4.8) is right for segment j, that is, $I_{j-1} < I \le I_j$, $F(I) = \varphi_{j-1}(I - I_{j-1}) + F(I_{j-1})$. Now, consider segment j + 1. From (2.1), we have for $I_j < I \le I_{j+1}$

$$F_1(I) = \beta_1 + \rho_1 I = \beta_1 + \rho_1 I_j + \rho_1(I - I_j) = \rho_1(I - I_j) + F_1(I_j);$$

$$F_2(I) = \beta_2 + \rho_2 I = \rho_2(I - I_j) + F_2(I_j);$$

$$\dots$$

$$F_j(I) = \beta_j + \rho_j I = \rho_j(I - I_j) + F_j(I_j);$$

$$F_{j+1}(I) = I = (I - I_j) + I_j = 1 \cdot (I - I_j) + F_{j+1}(I_j);$$

$$\dots$$

$$F_n(I) = I = 1 \cdot (I - I_j) + F_n(I_j).$$

So, by (4.7),
$$F(I) = \sum_{k=1}^{n} F_k(I) = (\rho_1 + \rho_2 + \dots + \rho_j + n - j)(I - I_j) + \sum_{k=1}^{n} F_k(I_j)$$

$$= \varphi_j \cdot (I - I_j) + F(I_j).$$ Q.E.D.

**Definition 4.2.** $S_i^{FIFO}(I)$ is the minimum available service function for connection $M_i$ at the FIFO scheduler.

**Lemma 4.3.** $S_i^{FIFO}(I)$ can be expressed in the piece-wise form as follows:

$$\text{(4.12)}\quad S_i^{FIFO}(I) = \begin{cases} 0 & I = 0, \\ \eta_j \cdot (I - F(I_j)) + S_i^{FIFO}(F(I_j)) & 0 \le j < n \text{ and } F(I_j) < I \le F(I_{j+1}), \\ \eta_n \cdot (I - F(I_n)) + S_i^{FIFO}(F(I_n)) & I > F(I_n); \end{cases}$$

where

$$\text{(4.13)}\quad \eta_j = \begin{cases} \dfrac{1}{\varphi_j} & 0 \le j \le i-1, \\ \dfrac{\rho_i}{\varphi_j} & i \le j \le n; \end{cases}$$

where $\varphi_j$ (j = 0, 1, 2, …, n) is defined in (4.9). The proof of this lemma can be found in Appendix A.

We would like to make some observations from the expression of $S_i^{FIFO}(I)$. First, this function is increasing and segmented at I = 0, $F(I_1)$, …, $F(I_n)$. Thus,

$$\text{(4.14)}\quad 0 = S_i^{FIFO}(F(I_0)) \le S_i^{FIFO}(F(I_1)) \le S_i^{FIFO}(F(I_2)) \le \dots \le S_i^{FIFO}(F(I_n)).$$

Furthermore, because $\varphi_j$ is decreasing as j increases, for j = 0, 1, …, i - 1, $\eta_j = 1/\varphi_j$ is increasing as j increases. Hence, during interval $[0, F(I_i))$, the slope of $S_i^{FIFO}(I)$ is non-decreasing. However,

$$\text{(4.15)}\quad \eta_{i-1} = 1/\varphi_{i-1} > \eta_i = \rho_i/\varphi_i.$$

This implies that at I = F($I_i$) while $S_i^{FIFO}(I)$ itself is still increasing, its slope is decreasing. Starting from I = F($I_i$), we have $\eta_j = \rho_i/\varphi_j$ which increases as j increases. Thus, the slope of the function is non-decreasing in (F($I_i$), $\infty$). Finally, note that $\eta_n = \rho_i/\varphi_n > \rho_i$. That is, the last segment increases at a rate larger than $\rho_i$. Figure 5 shows the curve of a typical $S_i^{FIFO}(I)$ that illustrates these observations.

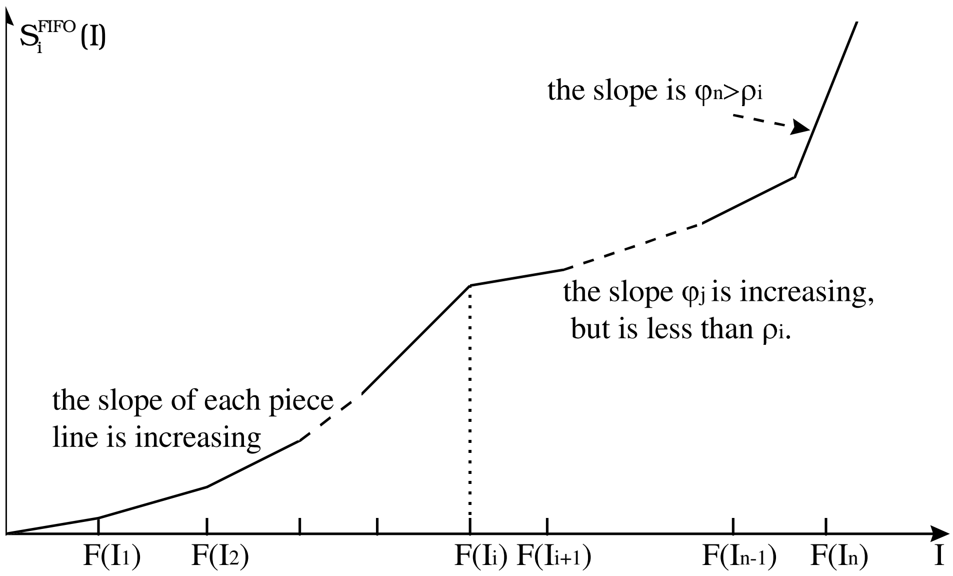



Figure 5. $S_i^{FIFO}(I)$ versus *I*.

We are now ready to derive $S_i$(I), the minimum available service function for connection $M_i$ in the regulated FIFO server. We will first look into a portion of $S_i$(I). We have the following lemma.

**Lemma 4.4.** For $0 \le I < F(I_i)$, the minimum available service function for connection $M_i$ in the regulated FIFO server is given by

(4.16) $$S_i(I) = \begin{cases} 0 & I = 0, \\ \min(\eta_k, \rho_i)\cdot(I - F(I_k)) + S_i(F(I_k)) & 0 \le k < i \text{ and } F(I_k) < I \le F(I_{k+1}). \end{cases}$$

**Proof.** Our proof will be informal in order to help to understand the physical intuition.

According to Lemma 4.3, for $0 \le I < F(I_i)$, the minimum available service function at the FIFO scheduler is given by

(4.17) $$S_i^{FIFO}(I) = \begin{cases} 0 & I = 0, \\ \eta_j \cdot (I - I_j) + S_i^{FIFO}(F(I_j)) & I_j < I \le I_{j+1},\ j = 0, 1, ..., i-1; \end{cases}$$

where $\eta_j = 1/\varphi_j$ . During interval [0, F($I_i$)), the slope of $S_i^{FIFO}(I)$ is non-decreasing .

Given the operational nature of the leaky bucket regulator, $S_i^{L.B.}(I)$, the minimum available service for connection $M_i$ at its leaky bucket is given by

(4.18) $$S_i^{L.B.}(I) = \rho_i I.$$

Now, we derive $S_i$(I) by an induction argument on the linear segments.

Consider the first segment, that is, $0 \le I \le F(I_1)$. In this segment, the service rate at the FIFO scheduler is $\eta_0$. If $\eta_0 \ge \rho_i$, the leaky bucket is the bottleneck. The service will be constrained at rate $\rho_i$. If $\eta_0 < \rho_i$, the FIFO scheduler is the bottleneck and at the start the cell queue in the leaky bucket is empty. The service will be constrained at rate $\eta_0$. Thus, for $0 \le I < F(I_1)$,

(4.19) $$S_i(I) = \min(\rho_i, \eta_0)I = \min(\rho_i, \eta_0)(I - F(I_0)) + S_i(F(I_0))$$

where $F(I_0) = 0$ and $S_i(F(I_0)) = S_i(0) = 0$. That is, the lemma holds for the first segment.

Assume that the lemma holds for segment j. Now consider segment j + 1, i.e., $F(I_j) < I \le F(I_{j+1})$, $1 < j < i$. If $\eta_j \ge \rho_i$, the leaky bucket is the bottleneck. The service will be constrained at rate $\rho_i$. If $\eta_j < \rho_i$, the FIFO scheduler is the bottleneck from 0 to $F(I_{j+1})$, that is due to the service rate at FIFO scheduler is non-decreasing. So the cell queue in the leaky bucket is empty. The service will be constrained at rate $\eta_j$. Thus, for $F(I_j) < I \le F(I_{j+1})$,

(4.19) $$S_i(I) = \min(\rho_i, \eta_j)(I - F(I_j)) + S_i(F(I_j))$$ Q.E.D.

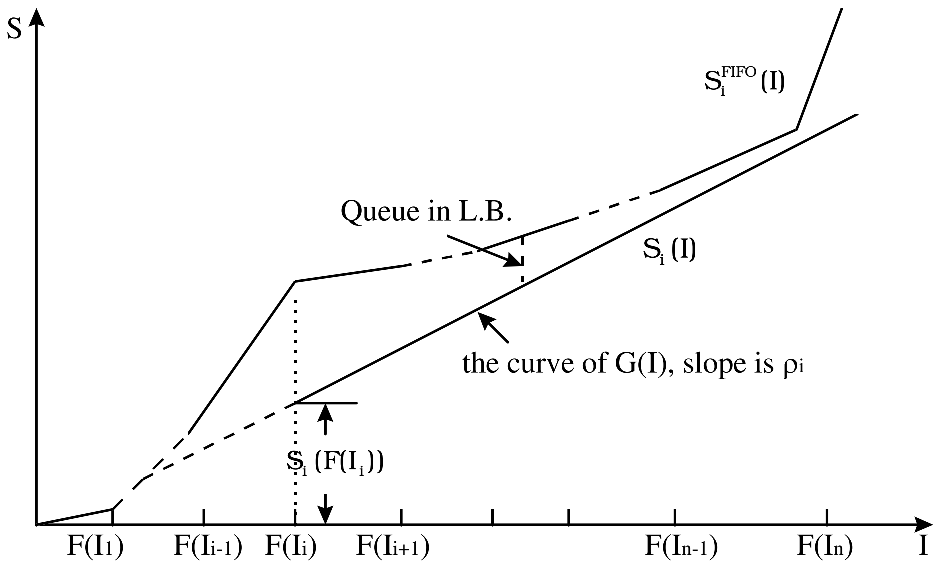



Figure 6.a. Case 1: *G(I)* is always no more than $S_i^{FIFO}(I)$.

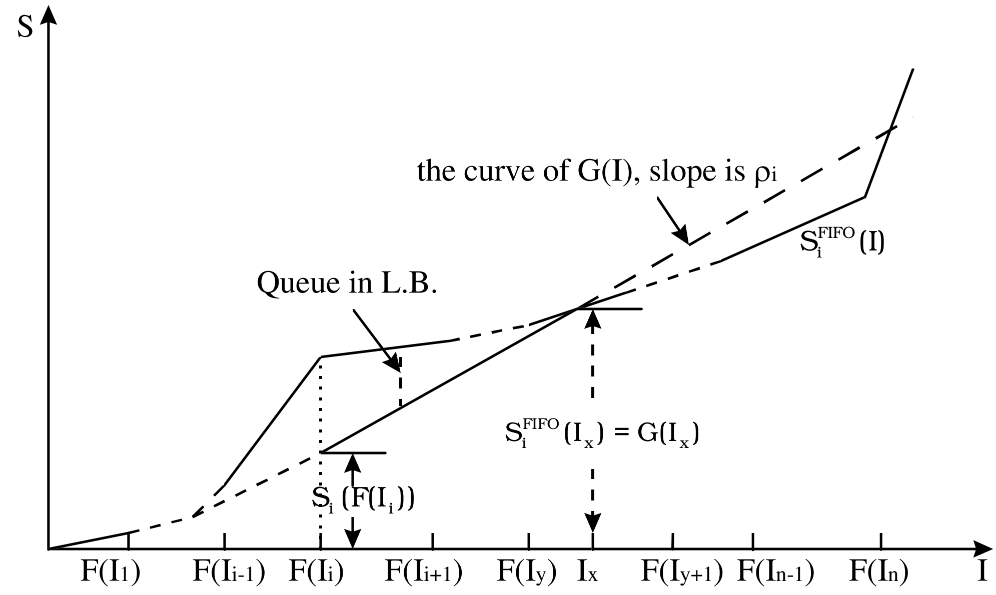



Figure 6.b. Case 2: *G(I)* becomes larger than $S_i^{FIFO}(I)$ at *Ix*.

We now construct $S_i(I)$ for $I > F(I_i)$. If the leaky bucket is the bottleneck, then the service provided by the compound server must be constrained by its service rate. If this is the case, potentially the minimum available service function could follow G(I) for $I > F(I_i)$ where

(4.21) $$G(I) = \rho_i(I - F(I_i)) + S_i(F(I_i)).$$

Unfortunately, the case may not be straightforward. We have to consider two cases:

**Case 1.** G(I) is always no more than $S_i^{FIFO}(I)$ for $I \geq F(I_y)$. See Figure 6.a for this situation. This implies that the cell queue in the leaky bucket is not empty. Hence, G(I) is indeed the service function. Thus,

(4.22) $$S_i(I) = \begin{cases} 0 & I = 0, \\ \min(\eta_k, \rho_i) \cdot (I - F(I_k)) + S_i(F(I_k)) & 0 \leq k < i \text{ and } F(I_k) < I \leq F(I_{k+1}), \\ \rho_i \cdot (I - F(I_i)) + S_i(F(I_i)). & I > F(I_i). \end{cases}$$

**Case 2.** Due to the fact that the rate of $S_i^{FIFO}(I)$ is reduced at $I = F(I_i)$, it is possible that G(I) becomes larger than $S_i^{FIFO}(I)$ at some point. Let this point be $I_x$. Because $S_i^{FIFO}(I)$ eventually increases faster than $S_i^{L.B.}(I)$, $S_i^{FIFO}(I)$ will eventually become larger than $S_i^{L.B.}(I)$. Figure 6.b shows this case.

Let us analyze the service rate that can be provided in this case. For $F(I_i) < I \leq I_x$, the cell queue in the leaky bucket is not empty. Hence, the total service rate can be provided must be the same as that of the leaky bucket. Hence, for $F(I_i) < I \leq I_x$,

(4.23) $$S_i(I) = \rho_i \cdot (I - F(I_i)) + S_i(F(I_i)).$$

When I is larger than $I_x$, the service should then limited by either the FIFO scheduler or the leaky bucket, whichever has the smaller rate. Let y be an index such that

(4.24) $$F(I_y) \leq I_x < F(I_{y+1}).$$

Then, $y \ge i$. By (4.13) and (4.9), $\min(\eta_j, \rho_i) = \eta_j$ if $i \le j < n$. That is, for $I_x < I \le F(I_n)$ the leaky bucket can serve a cell as soon as it is released by the FIFO scheduler. Thus, the FIFO server dominates the service rate. Hence, for $I_x < I \le F(I_n)$,

$$\text{(4.25)} \qquad S_i(I) = \begin{cases} \eta_y \cdot (I - I_x) + S_i(I_x) & I_x < I \le F(I_{y+1}), \\ \eta_j \cdot (I - F(I_j)) + S_i(F(I_j)) & y < j \le n \text{ and } F(I_j) < I \le F(I_{j+1}). \end{cases}$$

Finally, when $I > F(I_n)$, because $\eta_n > \rho_i$ the cell queue in the leaky bucket is not empty. Hence, service is constrained at rate $\rho_i$. Thus, for $I > F(I_n)$,

$$\text{(4.26)} \qquad S_i(I) = \rho_i \cdot (I - F(I_n)) + S_i(F(I_n)).$$

Synthesizing (4.23), (4.25), (4.26), we have, for Case 2,

$$\text{(4.27)} \qquad S_i(I) = \begin{cases} 0 & I = 0, \\ \min(\eta_k, \rho_i) \cdot (I - F(I_k)) + S_i(F(I_k)) & 0 \le k < i \text{ and } F(I_k) < I \le F(I_{k+1}), \\ \rho_i \cdot (I - F(I_i)) + S_i(F(I_i)) & F(I_i) < I \le I_x, \\ \eta_y \cdot (I - I_x) + S_i(I_x) & I_x < I \le F(I_{y+1}), \\ \eta_j \cdot (I - F(I_j)) + S_i(F(I_j)) & y < j \le n \text{ and } F(I_j) < I \le F(I_{j+1}), \\ \rho_i \cdot (I - F(I_n)) + S_i(F(I_n)) & I > F(I_n). \end{cases}$$

Using the expressions of $S_i(I)$ in (4.22) and (4.27), we can establish the following theorem.

**Theorem 4.2.** For the regulated priority driven server, the worst case delay for the cells from connection $M_i$ is given by

$$\text{(4.28)} \qquad d_i = \frac{\beta_i}{\rho_i} + F(I_n) - \frac{1}{\rho_i} S_i(F(I_n)).$$

**Proof**. We use Lemma 3.2 to derive (4.28). From (4.22) and (4.27), the minimum available service function $S_i(I)$ is weakly convex with respect to $\rho_i$ with $F(I_n)$ being a convex point.

The maximum arrival function of connection Mi is given in (2.1). From Lemma 3.2, the worst case delay of $M_i$ is

$$\text{(4.29)} \qquad d_i = \begin{cases} s_i(I_i) - I_i & \text{if } I_i > S_i(F(I_i)), \\ \dfrac{\beta_i}{\rho_i} + F(I_n) - \dfrac{S_i(F(I_n))}{\rho_i} & \text{otherwise;} \end{cases}$$

where $s_i()$ is the inverse function of $S_i()$.

From (4.22) and (4.27), we can solve $s_i()$ as follows: for $c > S_i(F(I_n))$,

$$\text{(4.30)} \qquad s_i(c) = \frac{1}{\rho_i}[c - S_i(F(I_n))] + F(I_n).$$

Thus, if $I_i > S_i(F(I_n))$,

$$\text{(4.31)} \qquad s_i(I_i) = \frac{1}{\rho_i}[I_i - S_i(F(I_n))] + F(I_n).$$

Substituting (4.31) into (4.29), we can have (4.28) established. Q.E.D.

While the derivation of $S_i(I)$ seems to be complicated, we observe that the final expression of the worst case delay from the integrated analysis is relatively simple. Note that at run-time, we will only use the delay expression to make the connection admission decisions.

## 5. Performance Comparison

In this section, we compare the worst case delay derived by our integrated method with that by the independent method. Table 1 lists the worst case delays derived by the independent methods [CRU91,RAH95]. For the sake of completeness, we include the details of these derivations in Appendix B.

| | Delay at the Scheduler | Delay at the Regulator | The Total Delay |
|---|---|---|---|
| Regulated Priority Driven Server | $\theta_i + I_i \cdot \frac{\pi_{i-1}}{1-\pi_{i-1}}$ | $\theta_i + I_i \cdot \frac{\pi_{i-1}}{1-\pi_{i-1}} + \frac{\beta_i}{\rho_i}$ | $2\cdot\theta_i + 2\cdot I_i \cdot \frac{\pi_{i-1}}{1-\pi_{i-1}} + \frac{\beta_i}{\rho_i}$ |
| Regulated FIFO Server | $\omega - (1-\pi_n)\cdot I_{\max}$ | $\omega - (1-\pi_n)\cdot I_{\max} + \frac{\beta_i}{\rho_i}$ | $2\cdot\omega - 2\cdot(1-\pi_n)\cdot I_{\max} + \frac{\beta_i}{\rho_i}$ |

where $\omega = \sum_{k=1}^{n} \beta_k$ $\quad \pi_0 = 0$ and $\pi_i = \sum_{k=1}^{i} \rho_k$, i = 1, 2, ..., n $\quad I_{\max} = \max_{k=1}^{n}(I_k)$

**Table 1. Worst Case Delays by Independent Methods**

### 5.1. Performance Metrics

We are interested in two kinds of metrics: the *admission probability* and the *relative error*. By admission probability, we meant the probability that a hard real-time connection can be admitted to the ATM switch. Recall that a hard real-time connection is admitted if and only if its worst case delay can meet its deadline constraint without violating the deadline constraints of connections that are already in the system. Obviously, the method to derive the worst case delay will impact the admission probability.

To obtain the admission probability, we used the discrete event simulation model to simulate the ATM switch. The simulation program is written in C++ programming language and run in a SUN/Solaris environment. There are altogether five system parameters for this simulation: $\lambda$ is defined as the mean of inter-arrival rate between connections; $\mu$ is the mean of the life-time of each connection; $\rho$ and $\beta$ are as defined in our previous sections, and lastly, D is the relative deadline for a connection. For each connection, $\rho$, $\beta$, and D are chosen from uniform distribution with D having its minimum value of $\beta/\rho$. The inter-arrival time and the life time of connections are chosen from exponential distributions. Hence, we have our estimated utilization, $U = \lambda\, \rho/\mu$.

We started our simulation with no connection in the switch. For each connection arrival, the worst case delays with a particular method were calculated. The system then decided to accept or reject the connection. To eliminating the warm-up effect, we let the system to run into a steady state before collecting the data. In particular, we let the system run through at least U/ρ number of connections before recording the data.

For each run of the simulation, we estimate the admission probability, $P_{admission}$ as follows:

(5.1) $$P_{admission} = \frac{N_{acceptance}}{\left(N_{acceptance} + N_{rejection}\right)}$$

where $N_{acceptance}$ is the number of acceptance and $N_{rejections}$ is the number of rejections.

Another statistical data we are interested in is the *relative error*. Let the worst case delays of a connection derived by the integrated method and independent method be $d^{integrate}$ and $d^{independent}$, respectively. The relative error, *e*, is defined as follows:

(5.2) $$e = \frac{d^{\text{independent}} - d^{\text{integrated}}}{d^{\text{integrated}}}.$$

Here, we set up another simulation experiment in order to collect statistics on the relative error. In this experiment, for each particular utilization, *U*, we randomly generated *N* connections such that $U = \sum_{i=1}^{N} \rho_i$. For every connection in the set, we computer the worst case delay with both the independent and integrated methods. We repeated the simulation 500 times, and obtained the average of the relative errors for each method.

## 5.2. Performance Results

Figures 7.a and 7.b show the admission probability according to the integrated method and the independent method for regulated priority driven server and regulated FIFO server. For both schedulers: as the total utilization increases, the admission probabilities for both methods decrease. That is, some connections will be rejected so as to keep the guarantee given to the previous connections. The rate of dropping in admission probability increases as the total utilization increases. When the servers are totally utilized, Figure 7.a and 7.b show that the admission probability for the integrated method drops to around 0.7 whereas the admission probability for the independent method drops to 0.65.

More importantly, from both Figures 7.a and 7.b, we observe that the integrated method is consistently superior to the independent method in terms of admission probability. That is, we will always admit more real-time connections to the system if the worst case delay is derived from the integrated method, no matter which scheduling policy the ATM switch has adopted.

Figures 8.a and 8.b show the average relative error for the regulated priority driven output port controller and the regulated FIFO output port controller, respectively. From both figures, we notice that the relative error is always positive and increases as the total utilization increases. This means that the worst case delays derived by our integrated method are always tighter than that by the independent method for both regulated priority driven server and regulated FIFO server.

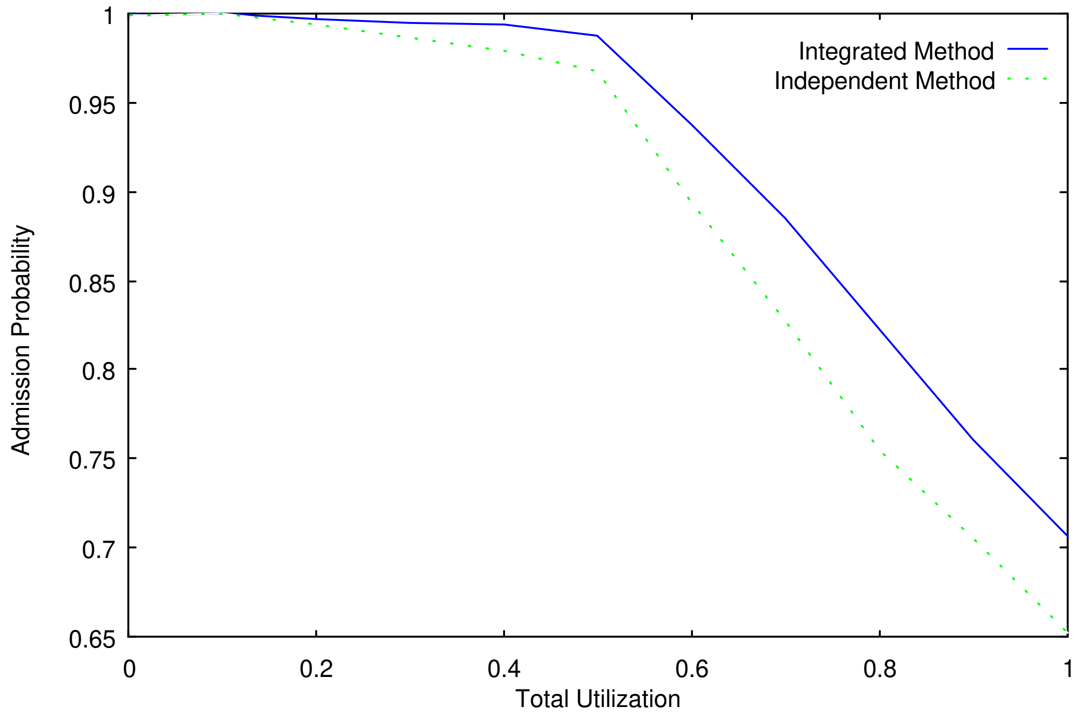



**Figure 7.a . Admission Probability --- Regulated Priority Driven Output Port Controller**

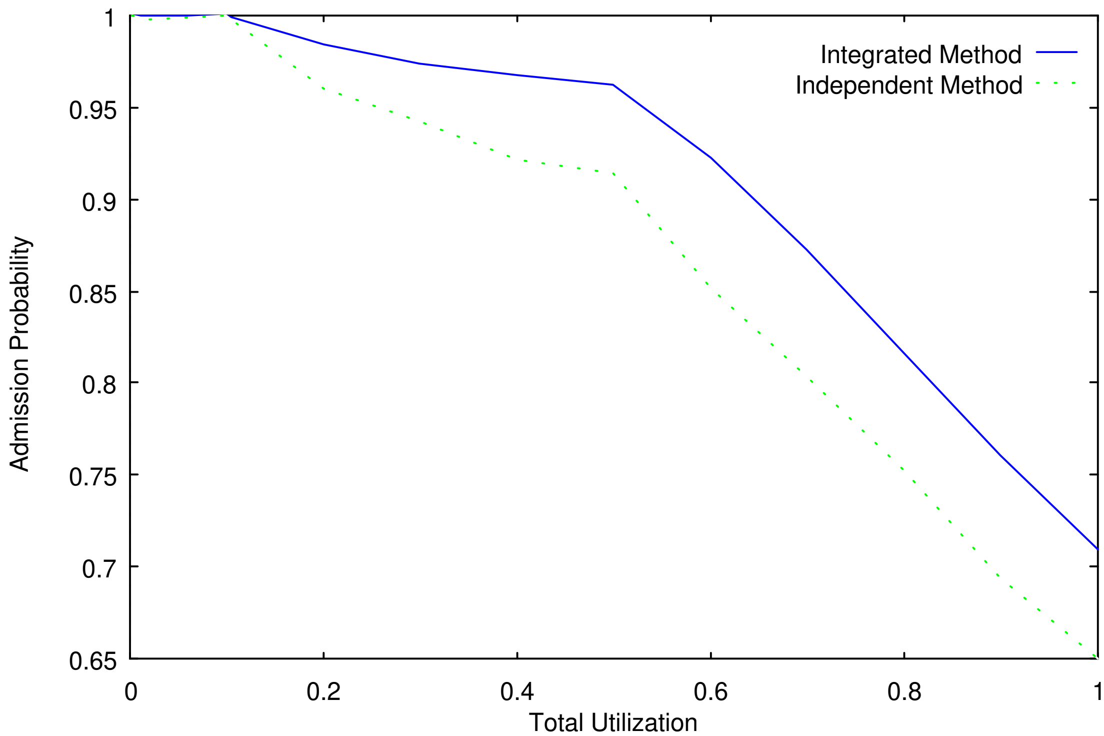



**Figure 7.b. Admission Probability --- Regulated FIFO Output Port Controller**

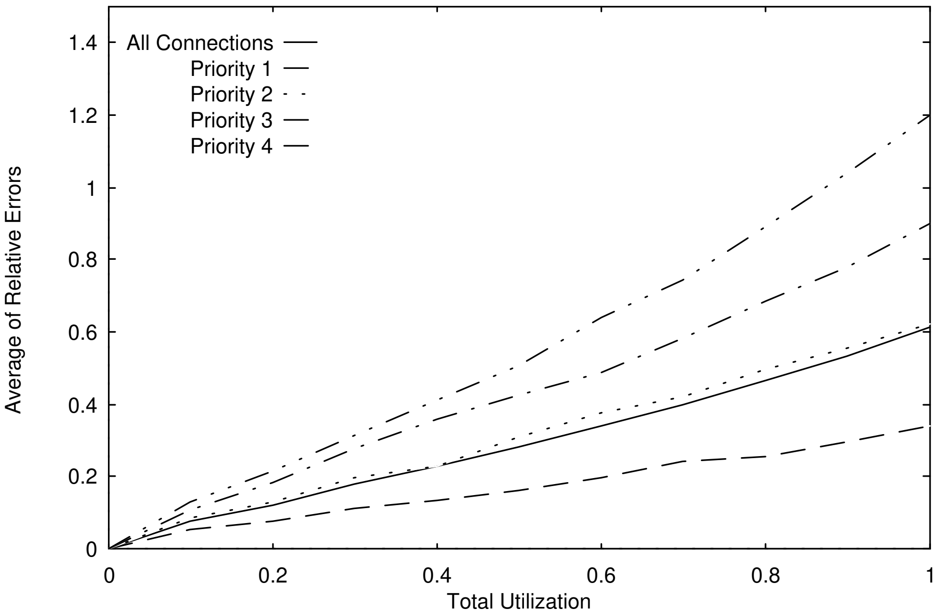



**Figure 8.a. Average Relative Error --- Regulated Priority Driven Output Port Controller**

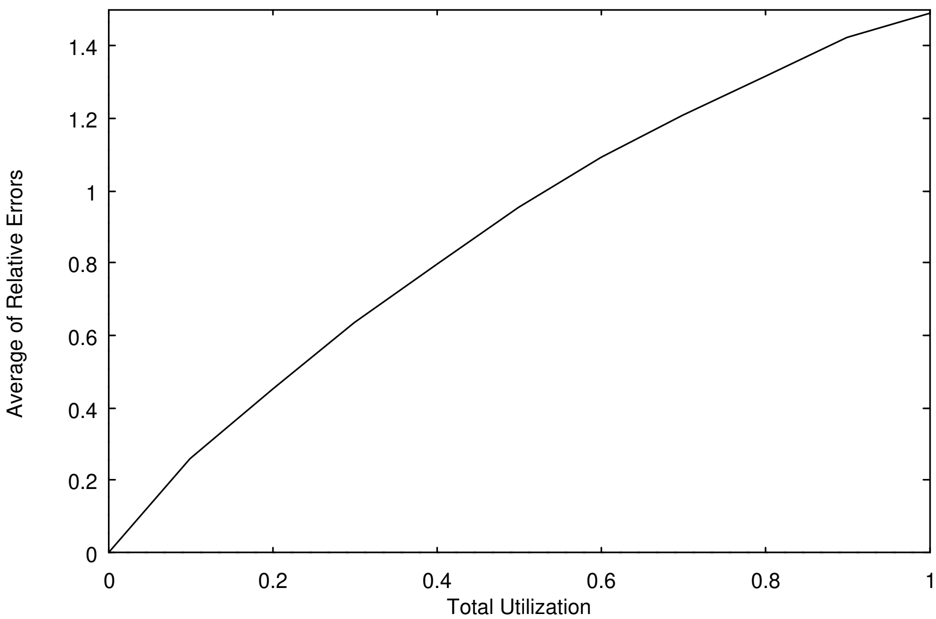



**Figure 8.b. Average Relative Error --- Regulated FIFO Output Port Controller**

For the regulated priority driven output port controller, connections are assigned 5 different priorities where priority 0 is the highest and priority 4 is the lowest. Generally speaking, from Figure 8.a we observed that the lower the priority, the larger the relative error with respect to the integrated method. In particular, the relative error for priority 0 is always zero because for the highest priority connections, the delays for the independent method and the integrated method are the same. However, relative errors for the lowest priority connection increase consistently, and reach 1.2 when the total utilization is equal to 1. For all connections, the average relative error increases with the total utilization. The error maximizes at 0.6 when the total utilization is equal to 1.

Under a regulated FIFO server, all connections are under the same priority. Nevertheless, from Figure 8.b, a similar observation is obtained. That is, as the total utilization increases, the relative error increases. However, the regulated FIFO server may now have the relative error maximized at 1.5 when the total utilization is equal to 1.

## 6. Summary and Future Work

In summary, we have done a worst case delay analysis for a regulated ATM switch. We present an efficient and accurate method to derive the worst case delay by utilizing the inverse of the arrival and service functions. We have shown that our proposed integrated method is superior to the independent method in terms of cell delays, admission probabilities, and relative errors at any utilization.

As for future work, based on the delay computation method we have developed, we can extend our model to include networks of ATM switches. By this extension, we can analyze the end-to-end delays in an ATM network. On the other hand, instead of using a two-piece linear function to describe the arrival traffic as we did in this report, one may like to consider other forms of arrival functions in order to investigate the tradeoff between the complexity of traffic constraint function and the performance gain.

## 7. References


[CHA94] C. S. Chang, "Stability queue length, and delay of deterministic and stochastic queueing networks," *IEEE Trans. Automat. Contr.,* vol. 39, no.5, pp. 913-931, May 1994.

[CRU91]R. L. Cruz, "A calculus for network delay, Part I & II," *IEEE Trans .Inform. Theory,* vol.37, no.1, pp.114-141, Jan. 1991.

[CRU92] R. L. Cruz, "Service burstiness and dynamic burstiness measures: A framework," *J. High Speed Networks,* vol. 1, no.2, pp.105-127, 1992.

[CRU93] R. L. Cruz, and M. C. Chuah, "Consequences of rate control in exhaustive service polling systems," in *Proc.27th Ann. Conf. Inform. Sci. Syst.,* Johns Hopkins University, Mar. 1993, pp.119-125.

[CRU94] R. L. Cruz and H. Liu, "End-to-end queuing delay in ATM networks," *J. High Speed Networks,* vol.3, no.4, pp.413-427, 1994.

[CRU95] R. L. Cruz, "Quality of service guarantees in virtual circuit switched networks," *IEEE. J. Select.Areas Commun.,* vol.13, no.6, pp.1048-1056.

[FER90] D. Ferri and D. C. Verma, " A scheme for real-time channel establishment in wide-area networks, " *IEEE J. Select. Areas Commun.,* vol. SAC-8, no.3, pp. 368-379, Apr. 1990.

[GEO96] L. Georgiadis, R. Guerin, V. Peris and K. N. Sivarajan, "Efficient network QoS provisioning based on per node traffic shaping," *Proc. IEEE Infocom'96*, March 1996.

[HLU88] M. G. Hluchyj and M. J. Karol, " Queueing in high performance packet switching, " *IEEE J. Select. Areas Commun.,* vol. 6, no. 9, pp. 1587-1597, Sept. 1988.

[HAN95a] C. Han, C. Hou, and K. Shin, "On slot allocation for time-constrained messages in DQDB networks," *Proc. of IEEE INFOCOM'95*, pp. 1164-1171, Boston, April 1995.

[HAN95b] C. Han and K. Shin, "Real-time communication in FieldBus multiaccess networks," *Proc. of IEEE RTAS'95*, pp. 86-95, May 1995.

[HAN97] C. Han, C. Hou, and K. Shin, "On slot allocation for time-constrained messages in dual-bus networks," to appear in *IEEE Trans. on Computers*, July 1997.

[KNI95] E. Knightly and H. Zhang, "Traffic characterization and switch utilization using deterministic bounding interval dependent traffic models," in *Proc. IEEE INFOCOM'95,* Apr. 1995, pp. 1137-1145.

[KUR92] J. Kurose, "On computing per-session performance bounds in high-speed multi-hop computer networks," in *Proc. ACM Sigmetrics, Perform.'92,* 1992.

[LEE96] C. Lee, K. Yoshida, C. Mercer and R. Rajkumar, "Predictable Communication Protocol Processing in Real-Time Mach", *Proc. of RTAS'96*, June 1996.

[LIZ97] C. Li, R. Bettati and W. Zhao, "Delay computation and priority assignment for ATM networks with static priority scheduling," *Technical report*, Dept. Computer Science, Texas A&M University, 1997.
[LIE96] J. Liebeherr, D. Wrege, and D. Ferrari, "Exact admission control in networks with bounded delay services," *IEEE/ACM Trans. Networking*, vol.4, no. 6, pp. 885-901, Dec. 1996.
[MUK93] Mukherjee, Tripathi, Saha, and Saksena, "A Bandwidth Allocation Scheme for Time Constrained Message Transmission on A Slotted Ring Lan," *Proc. of RTSS'93*, December 1993.
[NEW92] P. Newman, "ATM technology for corporate networks," *IEEE Comm. Magazine,* pp. 90-101, Apr. 1992.
[PAR93] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single node case," *IEEE/ACM Trans .Networking,* vol.1, no.3, pp. 344-357, June 1993.
[PAR94] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the multiple node case," *IEEE/ACM Trans. Networking,* vol.2, no.2, pp. 137-150, Apr. 1994.
[RAH95] A. Raha, S. Kamat, and W. Zhao, "Guaranteeing end-to-end deadlines in ATM networks," In *Proc. 15th IEEE International Conference on Distributed Computing Systems,* June, 1995.
[RAH96] A. Raha, S. Kamat, and W. Zhao, "Admission control for hard real-time connections in ATM networks," in *Proceedings of IEEE INFOCOM'96,* pp. 111-119, Mar. 1996.
[ROO94] R. Rooholamini, V. Cherkassky and M. Garver, "Finding the right ATM switch for the market," *IEEE Computer,* vol. 27, no. 4, pp. 16-18, Apr. 1994.
[SAH94] Saha, Tripathi, Saksena, and Mukherjee, "On Guaranteed Delivery of Time-Critical Messages in DQDB", *Proc. of INFOCOM '94*, March 1994.
[WRE96] D. Wrege, E. Knightly, H. Zhang and J. Liebeherr, "Deterministic delay bounds for VBR video in packet-switching networks: fundamental limits and practical trade-offs," *IEEE/ACM Trans. Networking,* vol. 4, no. 3, June, 1996.
[WRE97] D. E. Wrege and J. Liebeherr, "A near-optimal packet scheduler for QoS networks, " *Proc. IEEE INFOCOM'97,* March 1997.
[YAT93] D. Yates, J. Kurose, D. Towsley and M. G. Hluchyj, "On per-session delay distributions and the call admission problem for real-time applications with QoS requirements," In *Proc. ACM SIGCOMM'93,* pp. 2-12, Sept. 1993.
[ZHA94] H. Zhang and D. Ferrari, "Rate-controlled service disciplines," *J. High Speed Networks,* vol. 3, no. 4, pp. 389-412, 1994.
[ZHE96] Q. Zheng and K. G. Shin, "On the ability of establishing real-time channels in point-to-point packet-switched networks, " *IEEE Trans. Communications*, 19996.

# Appendix A.[Ω]

**Lemma 3.2.** Let S(), the minimum available service function for a connection at a server, be in the form of (3.8). The maximum arrival function of the connection, F(), is given as follows:

(A.1) $$F(X) = \begin{cases} X & X \le X^*, \\ \rho X + \beta & t > X^*, \end{cases}$$

where

(A.2) $$X^* = \beta/(1-\rho).$$

If the server is sequential and normal and S() is weakly convex with respect to ρ, then the worst case delay of the cells from the connection is given by

(A.3) $$d = \begin{cases} s(X^*) - X^* & \text{if} \quad X^* > S(X'), \\ \dfrac{\beta}{\rho} + X' - \dfrac{S(X')}{\rho} & \text{else;} \end{cases}$$

where s() is the inverse function of S(), X* is defined in (A.2) and X' is a convex point with respect to ρ in S().

**Proof**. We will use Lemma 3.1 to derive the worst case delay. Let s() and f() be the inverse functions of S() and F(), respectively. They are given by

(A.4) $$\text{f(c)}=\begin{cases} c & 0 \le c \le X^*, \\ \dfrac{c}{\rho} - \dfrac{\beta}{\rho} & c > X^*. \end{cases}$$

(A.5) $$\text{s(c)}=\begin{cases} 0 & c = 0, \\ \dfrac{1}{\mu_k} c + X_k - \dfrac{1}{\mu_k} S(X_k) & S(X_k) < c \le S(X_{k+1}) \text{ and } k = 0, 1, ..., m-1, \\ \dfrac{1}{\mu_m} c + X_m - \dfrac{1}{\mu_m} S(X_m) & c > S(X_m). \end{cases}$$

Define function g(c) = s(c) - f(c). Then, by Lemma 3.1,

(A.6) $$d = \max_c g(c).$$

Now let us find out the maximum point of g(c) and thus compute the worst case delay.

**Case 1**. $X^* > S(X')$. For c ≤ X*, the slope of g(c) is $1/\mu_k - 1 = (1-\mu_k)/\mu_k \ge 0$. That is, g(c) is non-decreasing in (0, X*). For c > X*, c > S(X') because $X^* > S(X')$. Due to the fact that S(X) is a weakly convex function, the slope of S(X) is no less than ρ when X > X'. Thus, the slope of s(c) is no more than 1/ρ when c > S(X'). At the same time, the slope of f() is 1/ρ. Thus, for c > X*, the slope of g() is non-positive. That is, g(c) is non-increasing when c > X*. Because functions S() and F() are continuous, s() and f() are also continuous. Function g(c) has its maximum at c = X*. The worst case delay of cells from the connection is then given by

(A.7) $$d = \max_c g(c) = g(X^*) = s(X^*) - X^*.$$

**Case 2.** $X^* \le S(X')$. Due to the fact that S(X) is a weakly convex function, the slope of S() is no more than ρ when X ≤ X' but no less than ρ when X > X'. Thus, the slope of s() is no less than 1/ρ when c ≤ S(X') but no

[Ω] Appendices A and B contain the proofs of lemmas used in this report.

more than 1/ρ when c > S(X'). For c < X*, the slope of g(c) is $1/\mu_k - 1 = (1-\mu_k)/\mu_k \geq 0$. For $X^* \leq c < S(X')$, the slope of g() is $1/\mu_k - 1/\rho \geq 1/\rho - 1/\rho = 0$. That is, g() is non-decreasing in (0, S(X')). For c ≥ S(X'), the slope of g() is $1/\mu_k - 1/\rho \leq 1/\rho - 1/\rho = 0$. That is, g() is non-increasing after S(X'). Thus, g() is maximized at c = S(X') and hence the worst case delay is given by

$$\begin{aligned} d &= \max_c g(c) = g(S(X')) \\ &= s(S(X')) - f(S(X')) \\ &= \frac{\beta}{\rho} + X' - \frac{S(X')}{\rho}. \end{aligned} \tag{A.8}$$

Q.E.D.

**Lemma 4.1.** For the regulated priority driven server, the minimum available service function of connection $M_i$ is given by

$$S_i(I) = \begin{cases} 0 & 0 \leq I \leq \theta_i, \\ \rho_i (I - \theta_i) & I > \theta_i; \end{cases} \tag{A.9}$$

where

$$\theta_i = \frac{\sum_{k=1}^{i-1} \beta_k}{1 - \sum_{k=1}^{i-1} \rho_k}. \tag{A.10}$$

**Proof**. We derive the minimum available service function of $M_i$ with priority driven scheduler at first. Let $S_i^{P.D}(I)$ be the minimum available service function for connection $M_i$ with the priority driven scheduler. With the priority driven scheduler, the cells of high priority connections will be transmitted first. Consider a busy interval of (t, t+I) where t is the starting point of the busy interval. Let $S_i^{H.P.}(I)$ be the maximum service time spent on all high priority connections in the time interval. Clearly, $S_i^{H.P.}(I)$ is given by

$$S_i^{H.P.}(I) = \min(I, \sum_{k=1}^{i-1} F_k(I)). \tag{A.11}$$

It is easy to verify that when I ≤ $\theta_i$ where $\theta_i$ is defined in (A.10), I ≤ $\Sigma_{k=1..i-1} F_k(I)$, and when I > $\theta_i$, I > $\Sigma_{k=1..i-1} F_k(I)$. Thus, we can re-write (A.11) as follows:

$$S_i^{H.P.}(I) = \begin{cases} I & I \leq \theta_i, \\ \sum_{k=1}^{i-1} (\beta_k + \rho_k I) & I > \theta_i; \end{cases} \tag{A.11'}$$

where θi is given in (A.10).

Then, in the same interval (t, t+I), the minimum available service time available for connection $M_i$ will be the service time left by serving higher priority connections. Thus, $S_i^{P.D}(I)$, the minimum available service function for connection $M_i$ with the priority driven scheduler is given by

$$S_i^{P.D.}(I) = I - S_i^{H.P.}(I) = \begin{cases} 0 & I \leq \theta_i, \\ (1 - \sum_{k=1}^{i-1} \rho_k) I - \sum_{k=1}^{i-1} \beta_k & I > \theta_i; \end{cases} \tag{A.12}$$

At the leaky bucket regulator, $S_i^{L.B.}$ (I), the minimum available service for connection $M_i$ is given by

(A.13) $$S_i^{L.B.}(I) = \rho_i I.$$

Now, we are ready to derive the minimum available service function $S_i(I)$ for the compound server. We consider two cases: $I \le \theta_i$ and $I > \theta_i$. For $I \le \theta_i$, due to $S_i^{P.D.}(I) = 0$, $S_i(I) = 0$.

For $I > \theta_i$, the service rate at priority driven scheduler is $1 - \sum_{k=1}^{i-1} \rho_k > \rho_i$. Thus, the leaky bucket is the bottleneck of service. Hence, the service rate of $S_i(I)$ is $\rho_i$, and

(A.14) $$S_i(I) = \rho_i (I - \theta_i).$$ Q.E.D.

**Lemma 4.3.** $S_i^{FIFO}(I)$ can be expressed in piece-wise form as follows:

(A.15)
$$S_i^{FIFO}(I) = \begin{cases} 0 & I = 0, \\ \eta_j \cdot (I - F(I_j)) + S_i^{FIFO}(F(I_j)) & 0 \le j < n \text{ and } F(I_j) < I \le F(I_{j+1}), \\ \eta_n \cdot (I - F(I_n)) + S_i^{FIFO}(F(I_n)) & I > F(I_n); \end{cases}$$

where

(A.16)
$$\eta_j = \begin{cases} \dfrac{1}{\varphi_j} & 0 \le j \le i-1, \\ \dfrac{\rho_i}{\varphi_i} & i \le j \le n; \end{cases}$$

where $\varphi_j$ (j = 0, 1, 2, …, n) is defined in (4.9).

**Proof.** Recall that by Definition 4.2, with the FIFO scheduler, F(I) is the maximum arrival function of all n connections. Let the inverse function of F(I) be f(). We first show the following result:

(A.17) $$S_i^{FIFO}(I) = F_i(f(I)).$$

Let t + I be a time instance in a busy interval where t is the starting point of busy interval. Given that the server is normal and sequential and that we are only interested in the worst case delay, we can assume that cell arrivals of connections follow their maximum arrival functions. By the definition of busy interval, $F(I) \ge I$. At time t + I, the total number of cells which have been served is I for this busy interval because the server can serve one cell per unit of time. Because this is an FIFO server, these I cells must have arrived at or before t + u where F(u) = I. That is, u = f(I). Among these I cells, the share for connection $M_i$ is $F_i(u)$. Hence,

(A.18) $$S_i^{FIFO}(I) = F_i(u) = F_i(f(I)).$$

Furthermore, from (4.8) we can solve the inverse function of F() as follows:

(A.19)
$$f(c) = \begin{cases} 0 & c = 0, \\ \dfrac{1}{\varphi_j}(c - F(I_j)) + I_j & F(I_j) < c \le F(I_{j+1}), j = 0, 1, ..., n-1, \\ \dfrac{1}{\varphi_n}(c - F(I_n)) + I_n & I > F(I_n). \end{cases}$$

Then, substituting (2.1) into (A.18), we have

(A.20) $$S_i^{FIFO}(I) = F_i(f(I)) = \begin{cases} f(I) & f(I) \le I_i, \\ \beta_i + \rho_i f(I) & f(I) > I_i; \end{cases}$$

$$= \begin{cases} f(I) & I \le F(I_i), \\ \beta_i + \rho_i f(I) & I > F(I_i). \end{cases}$$

Using (A.20), we can observe for j = 0, 1, …, n,

(A.21) $$S_i^{FIFO}(F(I_j)) = F_i[f(F(I_j))] = F_i(I_j) = \begin{cases} I_j & j \le i, \\ \beta_i + \rho_i I_j & j > i. \end{cases}$$

Now, we are ready to derive (A.15). For $I \le F(I_i)$, substituting (A.19) into (A.20), we have

(A.22) $$S_i^{FIFO}(I) = \begin{cases} 0 & I = 0, \\ \eta_j (I - F(I_j)) + I_j & F(I_j) < I \le F(I_{j+1}) \text{ and } j = 0, ..., i-1; \end{cases}$$

$$= \begin{cases} 0 & I = 0, \\ \eta_j (I - F(I_j)) + S_i^{FIFO}(F(I_j)) & F(I_j) < I \le F(I_{j+1}) \text{ and } j = 0, ..., i-1; \end{cases}$$

where $\eta_j$ is defined in (A.16) and $S_i^{FIFO}(F(I_j)) = I_j$ as shown in (A.21). For $I > F(I_i)$, substituting (A.19) into (A.20), we have, $F(I_j) < I \le F(I_{j+1})$ and $j = i, \ldots, n-1$,

(A.23) $$\begin{aligned} S_i^{FIFO}(I) &= \beta_i + \rho_i f(I) \\ &= \beta_i + \eta_j (I - F(I_j)) + \rho_i I_j \\ &= \eta_j (I - F(I_j)) + S_i^{FIFO}(F(I_j)), \end{aligned}$$

where $\eta_j$ is defined in (A.16) and $S_i^{FIFO}(F(I_j)) = \beta_i + \rho_i I_j$ as shown in (A.21). For the last segment (i.e., $I > F(I_n)$), the expression can be obtained in a similar manner. Q.E.D.

## Appendix B

Here, we derive the worst case delays at the FIFO server, the priority driven servers, and the regulator by using the independent methods. These delays have been listed in Table 1. We copy it as follows:

| | Delay at the Scheduler | Delay at the Regulator | The Total Delay |
|---|---|---|---|
| Regulated Priority Driven Server | $\theta_i + I_i \cdot \frac{\pi_{i-1}}{1-\pi_{i-1}}$ | $\theta_i + I_i \cdot \frac{\pi_{i-1}}{1-\pi_{i-1}} + \frac{\beta_i}{\rho_i}$ | $2 \cdot \theta_i + 2 \cdot I_i \cdot \frac{\pi_{i-1}}{1-\pi_{i-1}} + \frac{\beta_i}{\rho_i}$ |
| Regulated FIFO Server | $\omega - (1-\pi_n) \cdot I_{\max}$ | $\omega - (1-\pi_n) \cdot I_{\max} + \frac{\beta_i}{\rho_i}$ | $2 \cdot \omega - 2 \cdot (1-\pi_n) \cdot I_{\max} + \frac{\beta_i}{\rho_i}$ |

where $\omega = \sum_{k=1}^{n} \beta_k$  $\pi_0 = 0$ and $\pi_i = \sum_{k=1}^{i} \rho_k$, $i = 1, 2, \ldots, n$  $I_{\max} = \max_{k=1}^{n}(I_k)$

**Table B.1. Worst Case Delays by Independent Methods**

We will derive the results of Table B.1 in the following sub-sections:

**B.1. Delay at priority driven scheduler**

$S_i^{P.D}(I)$, the minimum available service function of connection $M_i$ is given in (A.12). $S_i^{P.D}(I)$ is a weakly convex function with respect to ρ and $\theta_i$ is a convex point. Due to $I_i > S(\theta_i) = 0$, by Lemma 3.2, the worst case delay of connection $M_i$ at the priority driven scheduler is

(B.1) $$d_i^{P.D.} = s_i^{P.D.}(I_i) - I_i,$$

where $s_i^{P.D}()$ is the inverse function of $S_i^{P.D}()$. From (A.12), we can solve $s_i^{P.D}()$ as follows:

(B.2) $$s_i^{P.D}(c) = \frac{1}{1-\pi_{i-1}} c + \theta_i,$$

where $\pi_{i-1} = \sum_{k=1,\ldots,i-1} \rho_k$. Substituting (B.2) into (B.1), we have

(B.3) $$d_i^{P.D.} = s_i^{P.D.}(I_i) - I_i = \theta_i + I_i \frac{\sum_{k=1}^{i-1} \rho_k}{1 - \sum_{k=1}^{i-1} \rho_k} = \theta_i + I_i \frac{\pi_{i-1}}{1-\pi_{i-1}}.$$

**B.2. Delay at FIFO scheduler**

In the FIFO scheduler, cells from all connections will be served in the order of first in and first out. The worst case delays of cells from all connections are the same. The maximum arrival function of all connections is given in (4.8) and the minimum available service function for all connections is

(B.4) $$S^{FIFO}(I) = I.$$

$s^{FIFO}(c)$, the inverse functions of $S^{FIFO}(I)$, is given by, for c $\geq 0$, (B.4’) $$s^{FIFO}(c) = c\,.$$

We would like to use Lemma 3.1 to drive the worst case delay at FIFO scheduler. We need to consider g(c) = $s^{FIFO}(c) - f(c)$ where f(c) is the inverse function of the maximum arrival function of all connections given in (A.19). Let $I_{max} = \max_{k=1,\ldots,n}(I_k)$. For $c \le F(I_{\max})$, the slope of f(c) is $1/\varphi_j < 1$ and the slope of $s^{FIFO}(c)$ is 1. So, the slope of g(c) is positive. That is, g(c) is increasing in $(0, F(I_{\max}))$. For $c > F(I_n)$, the slope of g(c) is $1 - 1/\varphi_n = (\varphi_n - 1)/\varphi_n < 0$, that is, g(c) is decreasing. Hence, g() is maximized at c = $F(I_{\max})$. By Lemma 3.1, we have

(B.5)
$$\begin{aligned} d_i^{FIFO} &= d^{FIFO} \\ &= \max_c(\mathrm{g(c)}) \\ &= s^{FIFO}(F(I_{\max})) - f(F(I_{\max})) \\ &= F(I_{\max}) - I_{\max} \\ &= \omega - (1-\pi_n)\cdot I_{\max}; \end{aligned}$$

where $\omega = \sum_{k=1}^{n} \beta_k$, $\pi_n = \sum_{k=1}^{n} \rho_k$, and $I_{\max} = \max_{k=1}^{n}(I_k)$.

**B.3**. **Delay at leaky bucket regulator**

We will first consider the case when the compound server is a regulated priority driven server. Let $F_i^{L.B.}(I)$ be the maximum arrival functions of cells from connection $M_i$ at the entry of the leaky bucket. In [CRU91], Cruz proved the following:

(B.6)
$$F_i^{L.B.}(I) = F_i\ (I + d_i^{P.D.}) = \begin{cases} I & I \le I_r, \\ \beta_i + \rho_i (I + d_i^{P.D.}) & I > I_r; \end{cases}$$

where $d_i^{P.D}$ is given in (B.3) and

(B.7)
$$I_r = \frac{\beta_i + \rho_i d_i^{P.D.}}{1-\rho_i}.$$

$S_i^{L.B.}()$, the minimum available service function as defined in (A.13), is weakly convex with respect to $\rho_i$. Any point I (I > 0) is a convex point with respect to $\rho_i$. Then, by Lemma 3.2, we can compute the worst case delay at the leaky bucket server:

(B.8)
$$d_i^{L.B.} = s_i^{L.B.}(I_r) - I_r = d_i^{P.D.} + \frac{\beta_i}{\rho_i}$$

where we use I = $I_r$ as a convex point. Substituting (B.3) into (B.8), we have

(B.9)
$$d_i^{L.B.} = \theta_i + I_i \frac{\pi_{i-1}}{1-\pi_{i-1}} + \frac{\beta_i}{\rho_i}.$$

The worst case delay of the cells at the leaky bucket regulator with a regulated FIFO server can be derived in a similar manner. The only difference is that when we drive (B.9) from (B.8), the delay of the FIFO scheduler should be used. That is, instead of (B.3), (B.5) should be substituted into (B.8) to obtain (B.9).