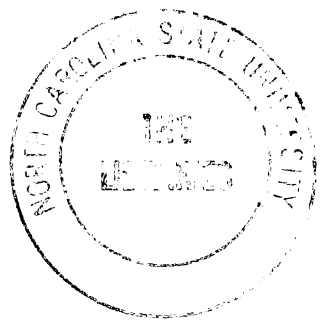
Approximate Analysis of a Shared-Medium ATM Switch Under Bursty Arrivals and Nonuniform Destinations

Atef O. Zaghloul

Harry G. Perros



Center for Communications and Signal Processing
Department of Computer Science
North Carolina State University

TR-93/2
February 1993

Approximate Analysis of a Shared-Medium ATM Switch under Bursty Arrivals and Nonuniform Destinations

Atef O. Zaghloul [1] and Harry G. Perros [2]

[1] IBM Corporation
Department C15
RTP, N.C. 27709
U S A

[2] Department of Computer Science, and
Center for Communications and Signal Processing
North Carolina State University
Raleigh, NC 27695-8206
U S A

Abstract

In this paper, we present an approximate analysis of a generic shared-medium ATM switch with input and output queueing. Input traffic is assumed to be bursty and is modelled by an Interrupted Bernoulli Process (IBP). Three different bus service policies are analyzed: Time Division Multiplexing (TDM), Cyclic, and Random. The output links may have constant or geometric service time. The analysis is based on the notion of decomposition whereby the switch is decomposed into smaller sub-systems. First, each input queue is analyzed in isolation after we modify its service process. Subsequently, the shared medium is analyzed as a separate sub-system utilizing the output process of each input queue. Finally, each output queue is analyzed in isolation. The results from the individual sub-systems are combined together through an iterative scheme. This method permits realistic system characteristics such as limited buffer size, asymmetric load conditions, and nonuniform destinations to be taken into consideration in the analysis. The model's accuracy is verified through simulation.

1.0 Introduction

The Asynchronous Transfer Mode (ATM) is the adopted transfer mode solution for broadband ISDN. ATM must be capable of efficiently multiplexing a large number of highly bursty sources, such as voice, video, and large file transfer. As a result, there is an increasing interest in the performance analysis of ATM based networks and in particular in ATM switches. There are several ATM switch architectures that have been proposed recently in the literature (see for instance Cidon et al [1], and Turner [15]). ATM switch architectures can be classified into three classes: ***shared-memory, shared-medium, and space-division*** . Tobagi [13] gives a detailed review of the different switch architectures and their implementation issues. In this study, we are interested in the analysis of the shared-medium architecture. In a shared-medium switch, all packets arriving on the input links are routed to the output links over a common high-speed medium such as a parallel bus. Each of the output links is capable of receiving all packets transmitted on the bus. The PARIS system described by Cidon et al [1] and the ATOM switch proposed by Suzuki et al [11], are two examples of such a switch architecture. One of the main advantages of the shared-medium architecture is the simplicity with which the multicast function is achieved since a transmitted packet over the shared-medium can be received by all output links simultaneously. Thus, the need to send multiple copies is eliminated. Also, priority among input links can be easily implemented within the bus arbitration policy. A shared-medium packet switch can provide packet queueing at the input links, the output links, or both the input and the output links (see Karol et al [6] and Pattavina [9]).

A queueing model of a generic shared-medium packet switch with input and output queueing is shown in figure 1. The input queues are attached to the shared bus and contend for access when they have one or more packets to transmit. The order in which input queues are served is determined by the bus service policy. In order to model such a system accurately, it is essential to take into consideration system characteristics such as finite buffer capacities. bursty input traffic, nonsymmetric load conditions, and nonuniform destinations. Cells arriving to a full input queue will be lost. Furthermore, if one of the output queues is full, the flow of customers destined to it may be momentarily stopped. This situation is commonly referred to as blocking. Because of the finite buffer capacities and blocking, closed-form solutions are difficult to

obtain. As a result, such queueing networks are typically analyzed approximately using the notion of decomposition.

In this paper, we analyze the shared-medium switch shown in figure 1 under three bus service policies: Time division multiplexing (TDM), Cyclic, and Random selection. The switch is analyzed approximately using the notion of decomposition. First, each input queue is analyzed in isolation after we modify its service process. Next, we analyze the shared bus as a separate sub-system utilizing the departure process from each input queue. Finally, we analyze each output queue in isolation. Each sub-system is analyzed numerically as a Markov chain. The results from individual sub-systems are combined together through the means of an iterative algorithm. The accuracy of the approximation algorithm is verified via simulation.

The shared-medium switch, under cyclic and random selection bus service policies, can be viewed as a finite capacity discrete-time polling system with blocking. The literature contains hundreds of papers on the subject of polling systems (see Takagi [12]). However, most of the studies reported were for one-message buffer systems or infinite buffer systems. Very few papers have dealt with finite buffer polling systems in discrete-time or in continuous-time. Jou, Nillson, and Lai [5] analyzed a discrete-time polling system with finite buffer capacity under bursty arrival process. Their approach gives an upper bound for the cell loss probability. Tran-Gia [14] gives an approximate algorithm for polling systems with finite capacity and non-exhaustive service. The analysis is based on the technique of the imbedded Markov chain and the evaluation of discrete convolution operations taking advantage of fast convolution algorithms, e.g., the Fast Fourier Transform. Ibe and Trivedi [4] considered finite-population and finite-capacity polling systems. Generalized stochastic Petri nets are used to describe the system. The major drawback of this method is the large state space of the system. Ganz and Chlamtac [2] presented an approximate solution to slotted communication systems in continuous time with finite population and finite buffer capacity. Similarly, very little has been reported in the literature on discrete-time queueing networks with blocking. Gershwin [3] analyzed a discrete-time queueing network with blocking that represents a synchronized production assembly system. It is often referred to as the transfer line model. The transfer line model has been analyzed by decomposing the network to sub-systems each consisting of two successive servers and the buffer in-between. Recently, Morris and Perros [7] developed an approximation algorithm for the analysis of a buffered Banyan ATM switch. The switch is analyzed by decomposing it into the individual switching elements. Further references

for the analysis of multi-stage interconnection networks can be found in [7]. A sub-system of this switch was analyzed approximately in [16]. See Perros [10] for a comprehensive review of queueing networks with blocking.

The paper is organized as follows. In section 2, a description of the system under study is presented. Section 3 covers the analysis of the three sub-systems. Numerical results are presented and compared against simulation results in section 4. Finally, conclusions are given in section 5.

2.0 The Queueing model under study

In this paper, we consider a single-stage N x N shared-medium switch with queue size (input and output) equal to m as shown in figure 1. Cells may be queued before switching at the inputs, if the output queue is full, as well as after switching at the outputs. A cell at the top of input queue i is destined for output queue j with probability d_{ij}, where $\sum_{j=1}^{N} d_{ij} = 1$, $i = 1, 2, ..., N$. Each input queue containing one or more packets arbitrates for the bus by activating the ***bus-request*** signal. A bus arbiter is used to select the input queue to be served next according to the bus service policy. Once an input queue has been granted the bus, only one cell is forwarded to its destination output buffer. If the output buffer is full, then the cell will be blocked. The blocked cell remain at the top of its input queue until the queue is granted the bus again. At that time, the cell will be forwarded to output queue j with probability d_{ij}.

The bus bandwidth is N times the speed of a single input link, where N is the number of input links. The arrival process to each input queue is slotted, with a slot size equal to the transmission time of an ATM cell. The bus is also slotted, with a slot size equal to 1/N of the transmission time of an ATM cell. For instance, if N = 6, then there are 6 bus slots within the boundary of each arrival slot as shown in figure 2. Three different bus service policies are considered in this paper: Time Division Multiplexing (TDM), cyclic order, and random selection. These service policies are described in detail in section 3.1. We assume zero switch-over time which means that when the bus completes serving the current queue, it switches instantaneously and starts serving the next queue. This is normally accomplished by separating the control signals, which are used for bus arbitration, and data signals, which are used for data transmission. This scheme permits arbi-

tration for the bus to be performed in parallel to data transmission; thus, justifying the zero switch-over assumption. Finally, the service time of each output link consists of n arrival slots, where n is geometrically distributed with parameter B_p. The servers of the N output links are synchronized. That is, they all begin and end a slot at the same time.

The arrival process to each of the input queues is assumed to be bursty and it is modelled by an Interrupted Bernoulli Process (IBP). That is, the incoming link into an input queue is slotted. Each slot is long enough to contain one cell. An arriving slot may or may not contain a cell. In an IBP, we have a geometrically distributed period during which no arrivals occur (idle state), followed by a geometrically distributed period during which arrivals occur in a Bernoulli fashion (active state). Given that the process is in the active state at slot t, it will remain in the active state during the next slot t + 1 with probability p, or it will change to the idle state with probability 1-p. If the process is in the idle state at slot t, it will remain in the idle state during the next slot t + 1 with probability q, or it will change to the active state with probability 1-q. During the active state, a slot contains a cell with probability α. The quantity α is also known as the peak bandwidth, i.e., the rate of arrivals during the active period. In [8], the average arrival rate, i.e., the probability that any slot contains a cell, is calculated as:

$$\rho = \frac{\alpha(1-q)}{2-p-q} \quad ,$$

and the squared coefficient of variation, C^2, of the time between successive arrivals is

$$C^2 = 1 + \alpha\left[\frac{(1-p)(p+q)}{(2-p-q)^2} - 1\right] .$$

Because of the finite buffer space at the input links, a cell arriving to a full input queue is lost. However, once a cell has been received, it will not be lost within the switch. In this paper, we assume that α is equal to 1 for each arrival process.

3.0 The analysis

The shared-medium switch described in the previous section is analyzed approximately by decomposing it into smaller sub-systems. First, each input queue is analyzed separately after we revise its service process. This revised service process which we will call the *effective service time* consists of three components. The possible delay due to bus contention, the actual bus transmission time (one bus slot), and the blocking delay due to a full destination queue. Having analyzed each input queue, we proceed to analyze a sub-system that represents the bus. The arrival process to this bus sub-system from each input queue i is approximated by an IBP, is obtained from the attempted departure process of input queue i. Finally, each output queue is analyzed in isolation. The attempted departure process out of the bus sub-system serves as the input process for the output queues.

Each sub-system is analyzed numerically as a Markov chain. However, in order to analyze each sub-system, information is needed from the other sub-systems. This information is updated through an iterative scheme. The algorithm iterates until convergence. Sections 3.1 and 3.2 describe the analysis of the input queues and the bus respectively. The analysis of the output queues is presented in section 3.3. Finally, a summary of the algorithm is given in section 3.4.

3.1 Analysis of the input queues

As stated earlier, each input queue is analyzed in isolation after we revise its service process. This revised service process is described in sections 3.1.1, 3.1.2, and 3.1.3 for TDM, Cyclic, and Random selection policies respectively. In section 3.1.4 we derive the cell loss probabilities for the input links.

3.1.1 TDM

Time Division Multiplexing (TDM) is the simplest bus allocation scheme. In TDM each bus slot i is preassigned to input queue i. Each input queue is only allowed to transmit during the bus slot assigned to it. If input queue i is empty, bus slot i is wasted since other busy input queues are not allowed to use it. The slot

assignments follow a predetermined pattern that repeats itself periodically as shown in figure 2. Each such period is called a bus cycle. Once, a cell reaches the head of an input queue, it waits for its slot number and then it is forwarded to the output buffer, if the output buffer is not full. If the output buffer is full, then the cell will be blocked and will wait for its slot number in the next bus cycle. We note that this blocking mechanism is different from the first-blocked-first-unblocked mechanism that has been typically used in continuous-time queueing networks with blocking. The blocking probability Pb_i for input queue i is calculated from the steady-state probabilities of the output queues in section 3.3.

The state of an input queue is fully described by the state vector $X = \{x_0, x_1, x_2\}$ where x_0 takes the following values: 0 if the input process is in the idle state, 1 if the input process is in the active state. Variable x_1 represents the number of cells in the queue, and it takes the values 0, 1, 2, 3, ... ,m, where m is the maximum queue capacity. Variable x_2 represents the present bus slot number and it takes the values: 1, 2, 3, ... , N. The stationary probability vector π of the input queues is obtained by solving the system of linear equations $\pi A = \pi$, where A is the transition probability matrix. In this paper, π was obtained for all three bus service policies using the Gauss-Seidel method.

3.1.2 Cyclic service

In this policy, the busy input queues are served in a cyclic fashion. The service policy in limited, i.e. only one cell is transmitted from the input queue being served before the server (the bus) switches to the next input queue. Once the present input queue has been served, the server proceeds in cyclic order until it finds an input queue which is non-empty. We assume zero switch-over time. Due to this cyclic policy, once a cell has reached the front of an input queue, it may be blocked for 0, 1, 2, ..., N-1 bus slots before it is granted the bus. This leads to representing the effective service time of each of the input queues by the phase-type structure shown in figure 3. The upper path in figure 3 represents the event that a cell at the top of input queue i will be transmitted immediately across the bus without delay. In this event, the effective bus service time is equal to one bus cycle. The probability of this event is denoted by r_{i1} for input queue i. The other paths in the figure represent the events that the cell at the top of input queue i finds one or more busy input queues already waiting for the bus. In this case, the cell from input queue i will have to wait for its turn in the polling cycle. This waiting time varies from 1 bus slot to N-1 bus slots. For example, the second path in

the figure represents the event where the effective bus service time is equal to 2 bus slots. During the first slot the cell is blocked at its respective input queue. During the second slot the cell is transmitted across the bus. The probabilities, $r_{i1}, r_{i2}, \ldots, r_{iN}$ are derived from the bus sub-system discussed in the section 3.2. If the destination output buffer is full, then the cell will be blocked. The blocked cell remains at the top of input queue i until the queue is granted the bus again. At that time, the cell will attempt to enter a destination queue j with probability d_{ij}.

The state of an input queue is fully described by the state vector $X = \{x_0, x_1, x_2, x_3\}$ where x_0 takes the following values: 0 if the input process is in the idle state, 1 if the input process is in the active state. Variable x_1 represents the number of cells in the node, and it takes the values 0, 1, 2, 3, ... ,m, where m is the maximum queue capacity. Variable x_2 represents the current bus slot number (1, 2, ..., N) which is used as a counter to keep track of the arrival slot boundaries. Variable x_3 represents the number of remaining service bus slots (1,2, ..., N).

3.1.3 Random selection

In this service policy the bus arbiter will select one input queue among the busy ones randomly. All busy input queues have equal probability of being selected. For instance, if there are N busy input queues at selection time, each busy input queue is selected with probability 1/N. On the other hand, if there is only busy input queue, it will be selected with probability 1. This policy could easily be modified to give one or more input queues a higher probability of being selected. Because we have random selection, when a cell reaches the front of input queue i, it may be selected (i. e. granted the bus) with probability c_i. This selection probability c_i , obviously, is a function of the number of busy input queues at selection time. The probabilities, $c_1, c_2, \ldots, c_N$ are derived from the bus sub-system discussed in the section 3.2. If the output buffer is full, then the cell will be blocked. The blocked cell remains at the top of its input queue until the queue is granted the bus again. As a result, one bus slot is wasted.

The state of an input queue is fully described by the state vector $X = \{x_0, x_1, x_2, x_3\}$ where x_0 takes the following values: 0 if the input process is in the idle state, 1 if the input process is in the active state. Variable x_1 represents the number of cells in the node, and it takes the values 0, 1, 2, 3, ... ,m, where m is the maximum queue capacity. Variable x_2 represents the current bus slot number (1, 2, ..., N). Variable x_3 indicates

whether the input queue is in service (granted the bus) or not and takes the values: 0 if the queue is not in service, 1 if the queue is in service.

3.1.4 The cell loss probability

Because of the finite buffer space at the input links, arriving cells to a full input queue will be lost. This loss probability P_L is calculated from the input queue stationary probability vector as follows. Let G be the set of states of the input queue in which the arrival process is in the idle state, H be the set of states of the input queue in which the arrival process is in the active state, F_I be the set of the states of the input queue in which the queue is full and the arrival process is in the idle state, and F_A be the set of the states of the input queue in which the queue is full and the arrival process is in the active state. Then, we have

$$P_L = \frac{\sum_{X \in F_I} P(X)(1-q) + \sum_{X \in F_A} P(X)p}{\sum_{X \in G} P(X)(1-q) + \sum_{X \in H} P(X)p}$$

where $P(X)$ is the steady-state probability that the input queue is in state X.

3.2 Analysis of the bus

The bus is analyzed as a separate sub-system. This sub-system is modeled as a single server, representing the bus, with a hypothetical queue that contains one customer from each busy input queue. Customers present in this hypothetical queue are served according to the bus service policy, i. e. TDM, Cyclic, or Random. After a cell from input queue i receives service equal to one bus slot, another cell from input queue i may join the hypothetical queue with probability p_i. On the other hand, a cell from input queue i which is not in the system during the present bus slot may arrive to the hypothetical queue with probability $1 - q_i$. The values for p_i and q_i for each of the input queues are obtained from the steady-state probability vector of the input queues as follows.

The states of input queue i are divided into three groups. Those states from which it is possible to have a departure will be called ***departure states***. Those states from which it is not possible to have a departure because of an empty queue will be called *idle states*. The third group consists of all the states in which the input queue has at least one customer; however, a departure is not possible. These states are called ***wait states***. The probability p_i can then be obtained as the probability that input queue i will be in a departure state or a wait state in the next slot, given that it is currently in a departure state. The probability q_i can be obtained as the probability that input queue i will be in one of the idle states in the next slot, given that it is currently in one of the idle states. Let $P(X)$ be the steady-state probability that the input queue is in state X, and let $t(X \rightarrow X')$ be the transition probability from state X to state X'. Then, we have

$$p_i = \frac{\sum_{X \in D} \left(P(X) \sum_{X' \in D,\, W} t(X \rightarrow X') \right)}{\sum_{X \in D} P(X)} \quad \text{and} \quad q_i = \frac{\sum_{X \in I} \left(P(X) \sum_{X' \in I} t(X \rightarrow X') \right)}{\sum_{X \in I} P(X)}$$

where D denotes the set of departure states, I denotes the set of idle states, and W denotes the set of wait states. Since the departure process is approximated by an IBP with $\alpha_i = 1$, this simplifies the task of selecting the appropriate departure, idle, and wait states as shown below.

1. TDM

   - Departure states: $x_1 > 0$ and x_2 (bus slot number) equal to the input queue number. Wait states: $x_1 > 0$ and $x_2 \neq$ input queue number. Idle states: $x_0 = x_1 = 0$.

2. Cyclic

   - Departure states: $x_1 > 0$ and $x_3 = 1$. Wait states: $x_1 > 0$ and $x_3 > 1$. Idle states: $x_0 = x_1 = 0$.

3. Random

- Departure states: $x_1 > 0$ and $x_3 = 1$. Wait states: $x_1 > 0$ and $x_3 = 0$. Idle states: $x_0 = x_1 = 0$.

For Cyclic and Random policies, the state of the bus sub-system is described by the state vector (W;S), where W denotes the input queues which have requested to use the bus. These are the queues that contain one or more cells waiting to be transmitted. The variable S represents the input queue number that is presently being served. For a system with 3 input queues numbered 1,2,3, W takes on the values (0), (1), (2), (3), (1,2), (1,3), (2,3), and S takes on the values 1, 2, and 3. For the TDM case, the state of the bus is described by the state vector (W;B,S) where W denotes the input queues that have requested the bus, B denotes the current bus slot number (B = 1,2, ..., N), and S denotes the state of the server (i.e. the bus), idle or busy. The bus sub-system is analyzed as a Markov chain using the Gauss-Seidel method.

Next, we derive the ***effective bus service time*** distribution for each input queue. The effective bus service time consists of the possible delay due to bus contention plus the actual bus transmission time (one bus slot). Note that the effective service time, mentioned earlier in the paper, consists of the effective bus service time plus the blocking delay due to a full destination queue. In the TDM case, the time it takes for an input queue to be granted the bus is deterministic since it is independent of the state of the other input queues. This is because, every input queue is served for exactly one bus slot in every bus cycle. In Cyclic and Random bus service policies, however, the time it takes for an input queue to be granted the bus is dependent upon the state of the other input queues. The input queue effective bus service time for the cyclic and random bus allocation policies are derived in the next two sections.

3.2.1 Effective bus service time - cyclic case

For the cyclic bus service policy, the effective bus service time is calculated by marking an arriving cell and following it until it departs. The distribution of the time that a cell spends in the system is viewed as the first passage time distribution between pairs of states (j,k) where j belongs to those states from which an arrival is permitted, and k belongs to those states where the cell is in service. The distribution of the first passage time between arbitrary pairs of states j and k is defined by,

$$f_{jk}^{(n)} = P(Y_n = k, Y_{n-1} \neq k, \ldots, Y_1 \neq k | Y_0 = j), n = 1,2,\ldots. \quad (1)$$

where Y_n is the state of the system at transition n. The effective bus service time distribution for each input queue is calculated from the steady-state probabilities as follows. First, we aggregate the bus states into three groups as they pertain to each of the input queues.

- S_i = set of states in which a cell from input queue i is in service.
- Q_i = set of states in which a cell from input queue i is requesting service, i.e. the cell is waiting in the bus queue.
- R_i = set of states in which a cell from input queue i is not in the system.

The probability that the effective bus service time for a cell from input queue i is n bus slots is equal to the first passage time $f_{jk}^{(n)}$ where j is the ***arrival state*** and k is the ***service state.*** In our model, the arrival states for input queue i are those states from which an arrival is possible, i.e. the union of the set of states S_i and R_i. The service states for input queue i are those states in which a cell from input queue i is in service, i.e. the set of states S_i. Note that all intermediate states between the arrival states and the service states, belong to the set Q_i.

The probability $f_i^{(n)}$ that the effective bus service time for input queue i is n bus slots is calculated from equation (1) by considering all arrival states and all service states for input queue i. Let $P(Y)$ be the steady-state probability that the bus is in state Y, and $\lambda_i(Y)$ be the probability that in the next slot there will be an arrival from input queue i given that the bus is in state Y. Furthermore, let $\iota(Y \rightarrow Y')$ be the transition probability from state Y to state Y' and $\iota^n(Y \rightarrow Y')$ be the nth power of the transition probability from state Y to state Y'. Then, the probability $f_i^{(n)}$ that the effective bus service time for input queue i is n bus slots is:

$$f_i^{(1)} = \frac{\sum_{Y \in S_i, R_i} \left(P(Y)\lambda_i(Y) \sum_{Y' \in S_i} \iota(Y \rightarrow Y') \right)}{\sum_{Y \in S_i, R_i} P(Y)\lambda_i(Y)}$$

$$f_i^{(2)} = \frac{\sum_{Y \in S_i,\, R_i} \left(P(Y)\lambda_i(Y) \sum_{Y' \in Q_i} \iota(Y \to Y') \sum_{Y'' \in S_i} \iota(Y' \to Y'') \right)}{\sum_{Y \in S_i,\, R_i} P(Y)\lambda_i(Y)}$$

and for n > 2,

$$f_i^{(n)} = \frac{\sum_{Y \in S_i,\, N_i} \left(P(Y)\lambda_i(Y) \sum_{Y' \in Q_i} \iota(Y \to Y') \sum_{Y' \in Q_i} \iota^{n-2}(Y' \to Y') \sum_{Y'' \in S_i} \iota(Y' \to Y'') \right)}{\sum_{Y \in S_i,\, R_i} P(Y)\lambda_i(Y)}$$

Note that $\lambda_i(Y) = p_i$ when $Y \in S_i$, and $\lambda_i(Y) = 1 - q_i$ when $Y \in R_i$. The transition probabilities $\iota(Y \to Y')$ are obtained readily from the bus transition matrix since the bus states are aggregated into three groups for each input queue as it was explained earlier. Finally, the above first passage probabilities $f_i^{(1)}, f_i^{(2)}, \ldots, f_i^{(n)}$ are substituted for the branching probabilities $r_{i1}, r_{i2}, \ldots, r_{in}$ shown in figure 3.

3.2.2 Effective bus service time - random selection case

In random selection bus service policy, the service time of input queue i is geometrically distributed with probability c_i. This is the probability that input queue i will be selected (granted the bus) during the next bus slot given that input queue i is busy. Note that the probability of being selected is dependent upon how many other input queues are arbitrating for the bus. Furthermore, the probability of selection is equally distributed among the busy queues.

The probability that input queue i will be selected to be served during the next bus slot can be directly obtained from the steady-state probabilities of the bus sub-system. Let $P(Y)$ be the steady-state probability that the bus is in state Y, and $\lambda_i(Y)$ be the probability that in the next slot there will be an arrival from input queue i given that the bus is in state Y. Let $t(Y \rightarrow Y')$ be the transition probability from state Y to state Y'. Then, the probability of selecting input queue i is

$$c_i = \frac{\sum_{Y} \left(P(Y)\lambda_i(Y) \sum_{Y' \in S_i} t(Y \rightarrow Y') \right)}{\sum_{Y} P(Y)\lambda_i(Y)}$$

where $\lambda_i(Y) = p_i$ when $Y \in S_i$, $\lambda_i(Y) = 1$ when $Y \in Q_i$, and $\lambda_i(Y) = 1 - q_i$ when $Y \in N_i$.

3.2.3 The bus attempted departure process

Next, we characterize the attempted departure process from the system bus. We use the attempted departure process rather than the actual departure process. The actual departure process is associated with departure instances where a cell leaves the input queue. The attempted departure process is associated with all instances of service completion independent of whether the cell leaves or gets blocked and is forced to receive another service. The aggregate attempted departure process from the bus is split into N different IBP processes which serve as the offered arrival processes to the output queues. The aggregate attempted departure

process is approximated by an IBP with parameters p_d, q_d, and α_d. We set $\alpha_d = 1$. The values for p_d and q_d are obtained using the same method as the case of p_i and q_i obtained in section 3.1.5. The states of the bus are divided into two groups; those states from which it is possible to have an attempted departure (active states), and those states from which it is not possible to have an attempted departure (idle states). The probability p_d can then be obtained as the probability that the bus will be in one of the active states in the next slot, given that it is currently in an active state. Similarly, the probability q_d can be obtained as the probability that the bus will be in one of the idle states in the next slot, given that it is currently in one of the idle states. Let $P(Y)$ be the steady-state probability that the bus is in state Y, and $t(Y \rightarrow Y')$ be the transition probability of state Y to state Y'. Then, we have

$$p_d = \frac{\sum_{Y \in A} \left(P(Y) \sum_{Y' \in A} t(Y \rightarrow Y') \right)}{\sum_{Y \in A} P(Y)} \quad \text{and} \quad q_d = \frac{\sum_{Y \in I} \left(P(Y) \sum_{Y' \in I} t(Y \rightarrow Y') \right)}{\sum_{Y \in I} P(Y)}$$

where the set of active states A consists of any state where the bus sub-system has one or more customers in the hypothetical queue.

The next step is to split the aggregate attempted departure process out of the bus sub-system into N separate processes which in turn become the arrival processes to the output queues. Let p_j, q_j, and α_j be the IBP parameters associated with the arrival process to output queue j. In this analysis, we assume that $p_j = p_d$ and $q_j = q_d$, for $j = 1, 2, ..., N$. The parameter α_j is calculated by equating throughputs at the output of the input queues and the input of the output queues. Let T_i be the throughput of of input queue i and d_{ij} be the routing probability that a cell from input queue i is destined for output queue j. Let T_j be the throughput of the traffic destined for output queue j. Then, we have

$$\alpha_j = \frac{T_j(2 - p_d - q_d)}{(1 - q_d)} \qquad \text{for } j = 1, 2, \ldots, N$$

where $T_j = \sum_{i=1}^{N} T_i(1 - Pb_i)d_{ij}$. The blocking probability Pb_i for input queue i is derived in the following section.

3.3 Analysis of an output queue

The arrival process to each output queue is modeled by an IBP as discussed above. The service process is assumed geometric. The state of the output queue is described by a state vector that keeps track of the state of the input process (idle or active), the number of customers in the queue, and the bus slot number. The output queue state vector is denoted $Z = \{z_0, z_1, z_2\}$ where z_0 takes the following values: 0 if the input process is in the idle state, 1 if the input process is in the active state. The variable z_1 represents the number of cells in the node, and takes the values 0, 1, ... , m where m is the queue size. The variable z_2 represents the current bus slot number, and takes the values 1, 2, ... , N where N is the number of input queues. The output queue is analyzed as a Markov chain using the Gauss-Seidel method.

Next, the blocking probability for each input queue is calculated. The blocking probability Pb_i for input queue i is the probability that given that the bus has been granted to input queue i, the cell at the top of input queue i is blocked from entering its chosen destination output queue. Let Pb_{ij} be the probability that output queue j is full, given that the bus has been granted to input queue i and the cell at the top of input queue i is destined for output queue j. Then, $Pb_i = \sum_{j=1}^{N} d_{ij}Pb_{ij}$, for i = 1, 2, ..., N. Pb_{ij} is obtained as follows. Let F_j be the set of all states in which output queue j is full. Recall that S_i is the set of states of the bus in which a customer from input queue i is in service. Then, we have

$$Pb_{ij} = \frac{\sum_{Z \in F_j} \left(P(Z) \sum_{Y} P(Y) \sum_{Y' \in S_i} t(Y \rightarrow Y') d_{ij} \right)}{\sum_{Y} P(Y) \sum_{Y' \in S_i} t(Y \rightarrow Y') d_{ij}}$$

3.4 Summary of the algorithm

The switch is analyzed by decomposing it into smaller sub-systems. First, each input queue is analyzed in isolation after its service process has been modified. Subsequently, the bus is analyzed as a separate sub-system utilizing the output process of each input queue. Finally, each output queue is analyzed as a separate sub-system. The approximation algorithm can be summarized as follows:

Step 1. For each input queue, initialize the service time distribution r_{i1}, r_{i2}, ... , r_{iN} (Cyclic case), the selection probability c_i, i = 1, 2, ..., N, (random selection case), and the blocking probability Pb_i.

Step 2. Analyze numerically each input queue. Calculate new values for p_i and q_i that characterize the departure process from each input queue i.

Step 3. Analyze numerically the common bus. Calculate the effective service time distribution for each input queue (r_{i1}, r_{i2}, ... , r_{iN} for TDM, and c_i for random selection). These new values are used in the next iteration. Calculate new values for p_j, q_j, and α_j that characterize the arrival process to each output queue.

Step 4. Analyze numerically each output queue. Calculate the queue length distribution and new values for the blocking probability Pb_i of each input queue.

Step 5. Repeat steps 2, 3, and 4 until a convergence test criterion has been met. The convergence test compares the queue length distribution of each input queue from one iteration to the next.

Step 6. Calculate performance measures such as queue length distribution, system throughput, and cell loss probabilities.

4.0 Numerical results

The approximation algorithm described in the previous section was implemented on an IBM 3090. The following performance measures were obtained for validation purposes: mean queue length (input and output queues), switch throughput, blocking probability, and cell loss probability. The approximate results were compared to results obtained by simulation. A representative sample of these comparisons is given for a 6x6 switch with queue size (input and output) equal to 32. The blocking probability at the output links B_p was set to 0.1. Confidence intervals for the simulation results were small and, therefore, they were not plotted.

Figures 4 to 7 have been obtained assuming a uniform destination distribution. That is, the routing probability d_{ij} was equal to 1/6 for all cells traversing the switch. In general, the results for the cyclic and the random selection policies were similar. Therefore, in figures 4 to 7, the comparisons are made between either Random selection and TDM, or Cyclic and TDM. Figure 4 gives the mean input queue length for TDM and cyclic bus service policies under symmetric arrivals and uniform destinations. The arrival process is an IBP with $\alpha = 1$, such that $\rho = 0.7$ and $C^2 = 100$. We observe that under the cyclic bus service policy, the mean input queue lengths are equal. On the other hand, under TDM, the mean input queue length increases as the input queue number is increased from 1 to 6. This is due to the fact that if two or more input queues are waiting to transmit a cell to the same output queue which has only one empty space, then the input queue with the lowest queue number will transmit successfully, whereas the other queues will be blocked. Since, the state of the output queue does not change within a bus cycle, the input queue blocking probability will increase as the input queue number is increased, e.g. input queue 6 will have the largest blocking probability. The results show that, due to blocking, TDM policy has a fairness problem. There is a big difference, nearly 7-fold, in the mean queue length between input queue 1 and input queue 6. The

maximum, minimum, and average relative errors for the results in figure 4 are 8.1%, 1.9%, and 4.8% respectively. Figures 5 and 6 show the blocking probability (the probability that the cell at the top of queue i will be blocked due to a full destination queue) and the cell loss probability for each input queue for the same example considered in figure 4. Again, under TDM, we note that input queue 6 suffers the most in terms of cell loss probability. We note that the fairness problem can be solved by rotating the bus slot assignments within bus cycles (i.e. (1, 2, 3, 4, 5, 6), (2, 3, 4, 5, 6, 1), (3, 4, 5, 6, 1, 2), ... etc.). The maximum, minimum, and average relative errors for results in figure 5 are 14.0%, 3.9%, and 9.8% respectively and for results in figure 6 are 14.1%, 2.3%, and 9.4% respectively. Figure 7 gives the mean queue length of each input queue for TDM and Random bus service policies under asymmetric IBP arrivals with parameters: $\rho_1 = 0.1,\ \rho_2 = 0.2,\ \rho_3 = 0.3,\ \rho_4 = 0.4,\ \rho_5 = 0.5,\ \rho_6 = 0.6,$ and $C_i^2 = 100$ for $i = 1,2,...,6$. We note that random selection is considerably superior to TDM since the selection is made among only the busy input queues. Thus, bus slots are not wasted on empty input queues as in TDM. The maximum, minimum, and average relative errors for the results in figure 7 are 14.5%, 0.9%, and 5.2% respectively.

Results in figures 8 to 11 were obtained under a nonuniform destination distribution. In particular, for each input queue 30% of the traffic was destined to output queue 1 and the remainder of the traffic was uniformly distributed among the rest of the output queues, i.e. output queues 2 to 6 get 14% each. This type of traffic pattern is referred to in the literature as the ***hot-spot*** pattern. Figures 8 to 10 give the mean queue length of output queue 1 (hot output) and output queue 6 (a non-hot output) for the three bus service policies, TDM, Random, and Cyclic respectively. The arrival process to each input queue was an IBP with $C^2 = 50$ and ρ increasing from 0.1 to 0.5. Note that when ρ is equal to 0.5, the average rate of arrivals destined for output 1 (hot output) is equal to 0.9 which is the output link capacity. For all three bus service policies, although not shown, the input queues were virtually empty. This is due to the fact that the bus speed is equal to the sum of the input link capacities and as a result, most of the queueing within the switch occurs in the output queues. We note that there is a good agreement between the approximation results and the simulation data. The maximum, minimum, and average relative errors for the results in figure 8 are 8.1%, 2.6%, and 4.9%, and for the results in figure 8 are 14.5%, 6.1%, and 10.6% respectively. Maximum, minimum, and average relative errors for results in figure 8 are 14.8%, 4.9%, and 10.7% respectively. Figure 11 gives the blocking probability for each input queue under asymmetric arrival process

($\rho_1 = 0.1$, $\rho_2 = 0.2$, $\rho_3 = 0.3$, $\rho_4 = 0.4$, $\rho_5 = 0.5$, $\rho_6 = 0.6$, and $C_i^2 = 100$ for $i = 1,2,...,6$) and nonuniform traffic destination distribution (the same destination distribution mentioned above). This pattern gives wide variation of traffic crossing the switch. Under TDM bus service policy, the blocking probability increases as the input queue number is increased from 1 to 6 which is expected. On the other hand, under cyclic service, the blocking probability is highest for the input queue with the lowest arrival rate and vice versa. This can be explained, intuitively, by the fact that the heavily utilized input queues keep the hot spot output full most of the time. Thus, blocking probabilities of the input queues with low arrival rates is increased. The maximum, minimum, and average relative errors for the results in figure 8 are 16.7%, 3.3%, and 11.6% respectively.

In general, the accuracy of the algorithm is good. It was observed that if the input or output queue size is small (i.e. less than 16) the results are not very good when C^2 is high (i.e. greater than 100). When the queue size is large (i.e. equal to or greater than 32) the accuracy is good for values of C^2 up to 150.

5.0 Conclusions

In this paper, we presented an approximate analysis of a generic shared-medium ATM switch under realistic system characteristics such as limited buffer size, asymmetric load conditions, and nonuniform destinations. Three different bus service policies were analyzed: Time Division Multiplexing (TDM), Cyclic, and Random.

The analysis is based on the notion of decomposition whereby the switch is decomposed into smaller sub-systems. First, each input queue is analyzed in isolation after we modify its service process. Subsequently, the shared medium is analyzed as a separate sub-system utilizing the output process of each input queue. Finally, each output queue is analyzed in isolation. The results from the individual sub-systems are combined together through an iterative scheme. The model's accuracy is verified through simulation. It was shown that it has an acceptable accuracy.

References

1. I. Cidon, I. Gopal, G. Grover, and M. Sidi, "Real-Time Packet Switching: A Performance Analysis", IEEE JSAC Vol. 6 NO. 9, December 1988.

2. A. Ganz, and I. Chlamtac, "A Linear Solution to Queueing Analysis of Synchronous Finite Buffer Networks", IEEE Trans. on Comm., Vol. 38, NO. 4, April 1990.

3. S. Gershwin, "An efficient decomposition algorithm for unreliable tandem queueing systems with finite buffers", First Int. Workshop on Queueing Networks with blocking, North Holland, 1989.

4. O. Ibe and K. Trivedi, "Stochastic Petri Net Models of Polling Systems", IEEE JSAC, Vol.8 NO. 9, December 1990.

5. F. Jou, A. Nilsson, and F. Lai, "The Upper Bounds for Delay and Cell Loss Probability of Bursty ATM Traffic in a Finite Capacity Polling System", Second Int. Workshop on Queueing Networks with blocking, RTP, North Carolina, May 1992.

6. M. Karol, M. Hluchyj, and S. Morgan, "Input versus output queueing on a space-division packet switch", IEEE Trans. Comm., VOL. 35, NO. 12, December 1987.

7. T. Morris and H. Perros, "Performance analysis of a multi-buffered Banyan ATM switch under bursty traffic", INFOCOM 92, Florence, Italy.

8. A. Nilsson, F. Lai, H. Perros, "An Approximate analysis of a bufferless NxN synchronous Clos ATM switch" in: Cohen and Pack (eds.), Queueing, Performance, and Control in ATM, North-Holland, 1991, 39-46.

9. A. Pattavina, "Performance evaluation of ATM switches with input and output queueing", International Journal of Digital and Analog Communication Systems, VOL. 3, 1990.

10. H.G. Perros, "Approximation algorithms for open queueing networks with blocking", in Takagi (ed.), Stochastic Analysis of Computer and Communication Systems, North-Holland, 1990, 451-498.

11. H. Suzuki, H. Nagano, T. Takeuchi, and S. Iwasaki, "Output-buffer switch architecture for asynchronous transfer mode", Int. Conf. on Communications, Boston, MA, June 1989.

12. H. Takagi, "Queueing Analysis of Polling Models: An Update", in Takagi (ed.), Stochastic Analysis of Computer and Communication Systems, North-Holland, 1990, 267-318.

13. F. Tobagi, "Fast Packet Switch Architectures for Broadband Integrated Services Digital Networks", Proceedings of the IEEE, Vol. 78, NO. 1, January 1990.

14. P. Tran-Gia, "Analysis of Polling Systems with General Input Process and Finite Capacity ", IEEE Trans. on Comm., VOL.40, NO.2, February 1992.

15. J. Turner, "Design of a broadcast packet switching network", IEEE Trans. Comm., Vol. 36, NO. 6, June 1988.

16. A. Zaghloul and H. Perros, "Approximate analysis of a discrete-time polling system with bursty arrivals", IFIP workshop on "Modelling and Performance Evaluation of ATM technology", Martinique, January 1993.

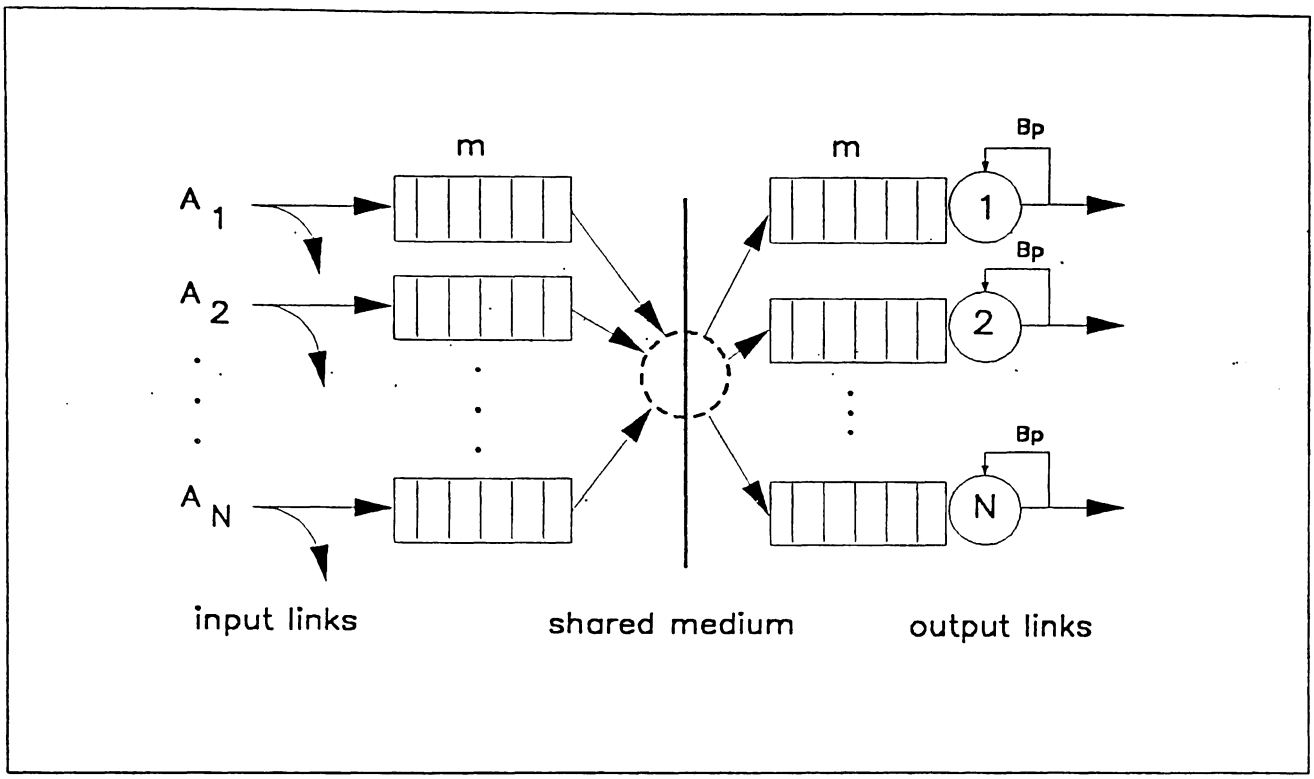


Figure 1. Queueing model of a shared-medium switch

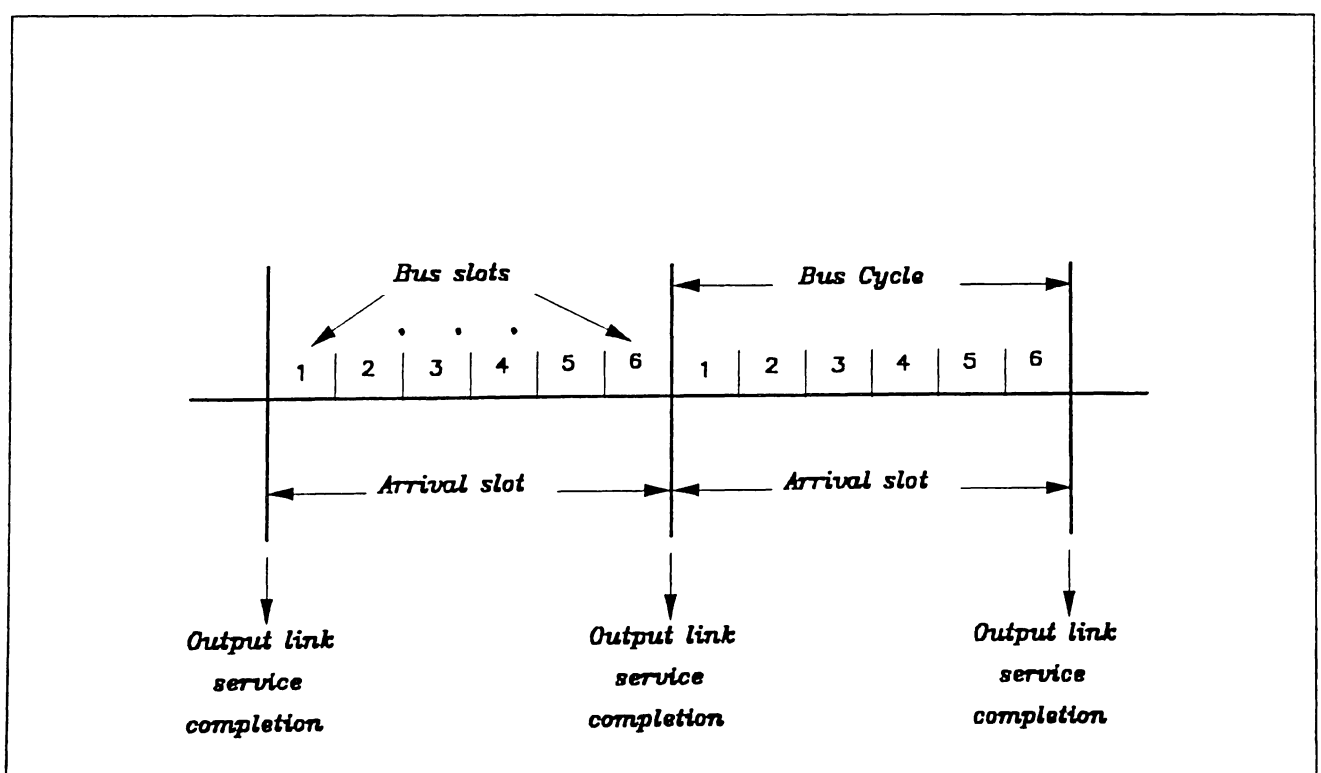


Figure 2. Bus slots and arrival slots

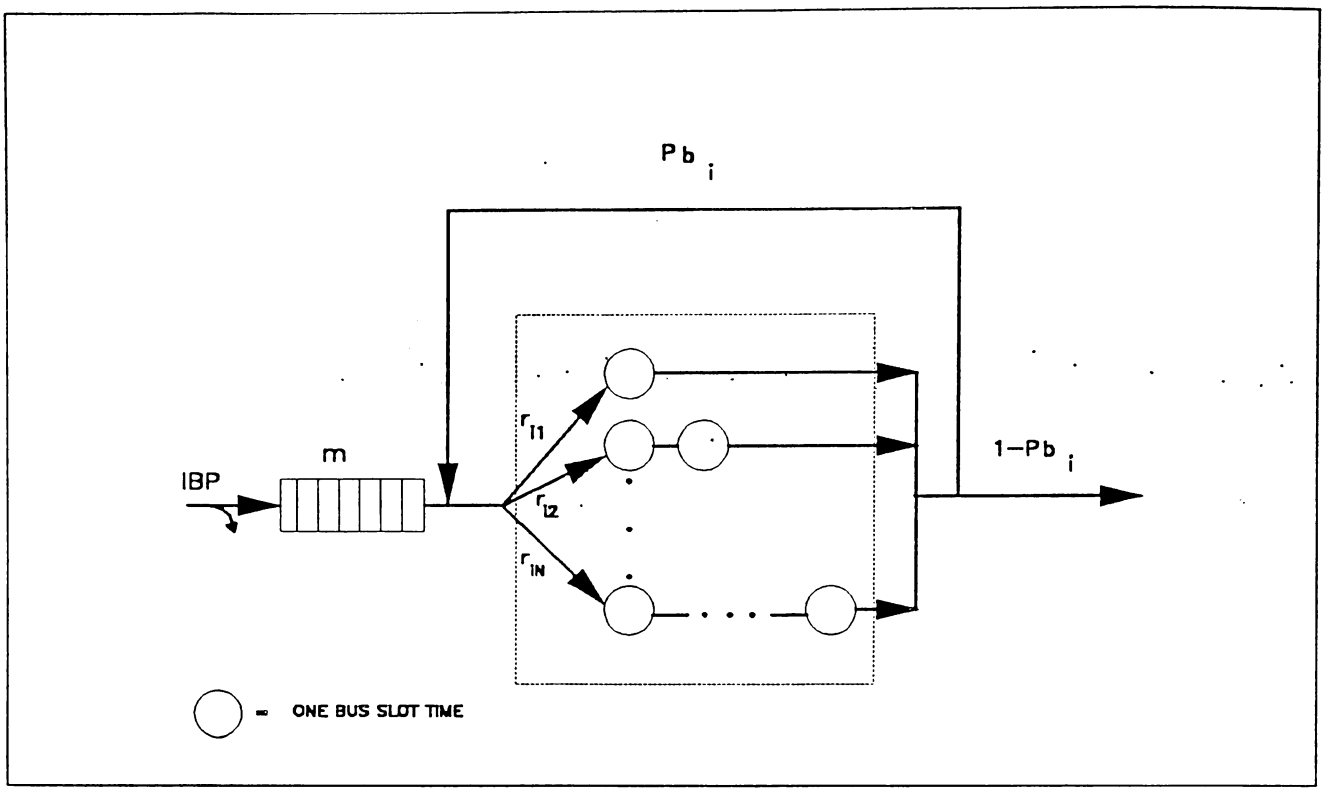


Figure 3. Effective service time of an input queue (Cyclic case)

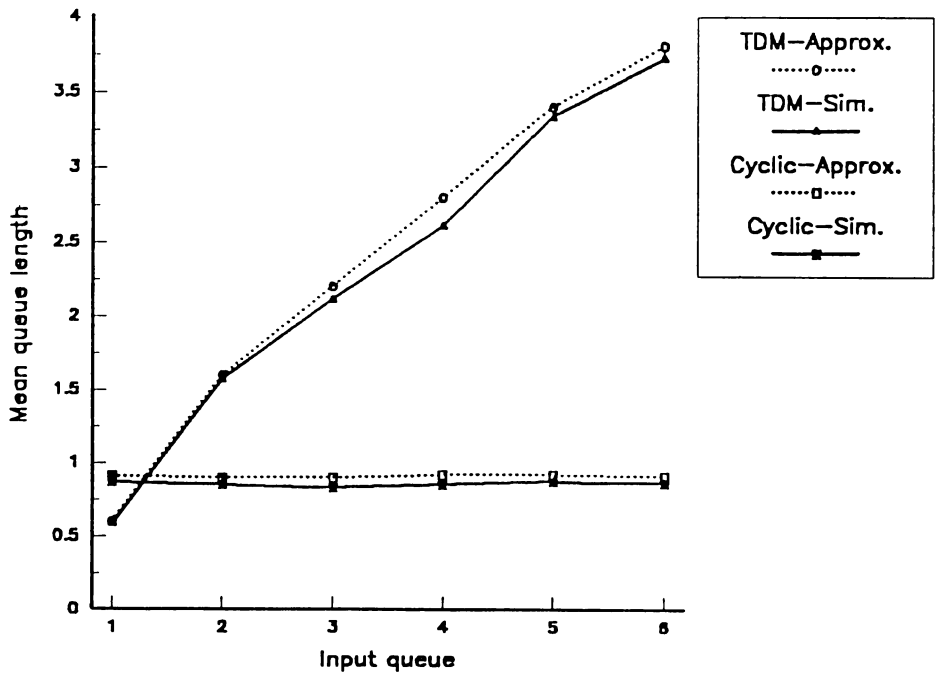


Figure 4. Mean input queue length: uniform destinations, $\rho = 0.7$, $C^2 = 100$

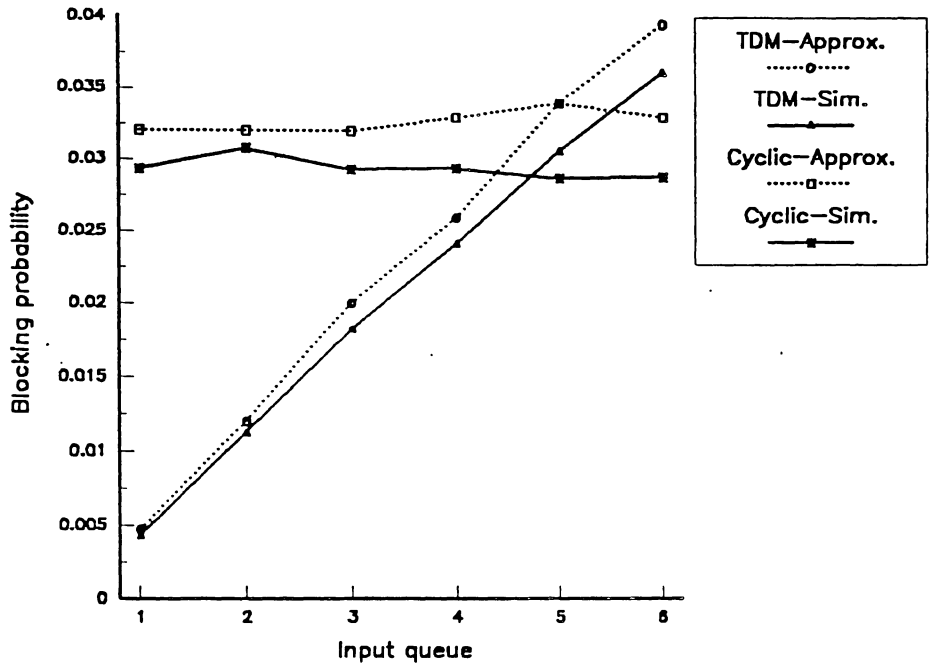


Figure 5. Blocking probability: uniform destinations, $\rho = 0.8$, $C^2 = 100$

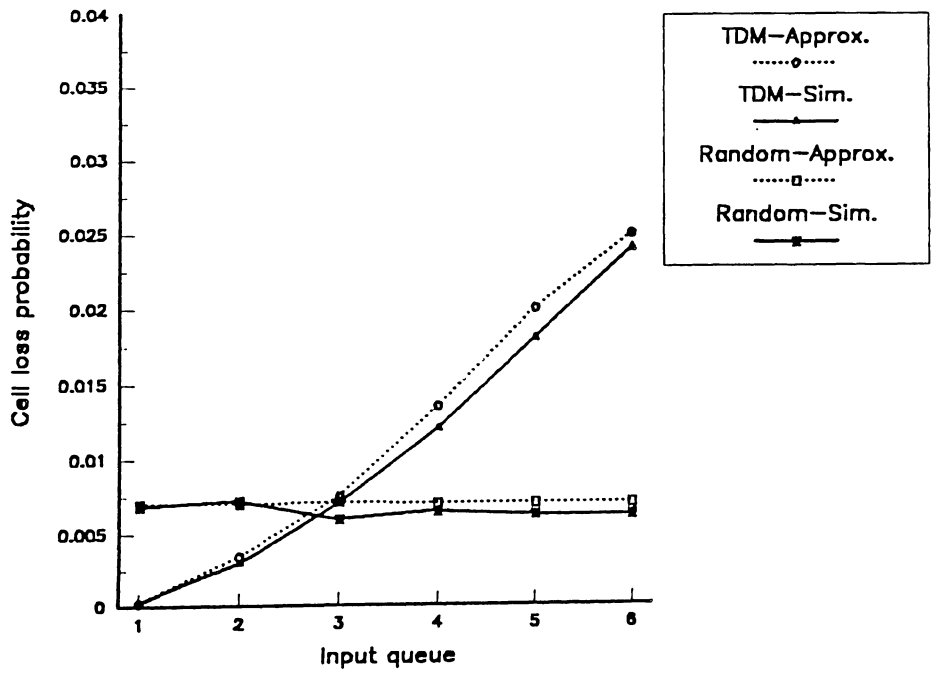


Figure 6. Cell loss probability: uniform destinations, $\rho = 0.8$, $C^2 = 100$

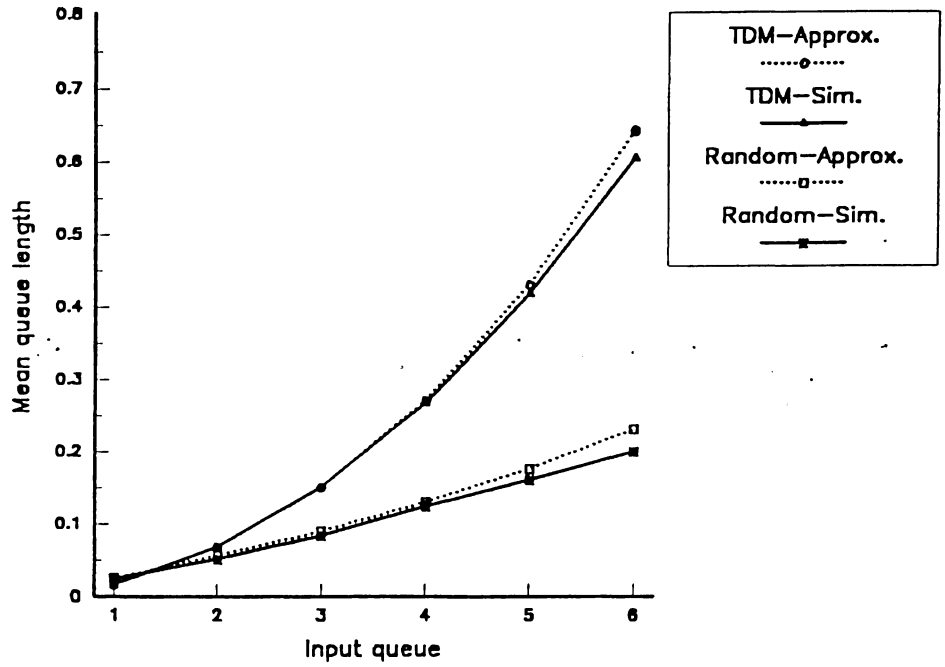


Figure 7. Mean queue length: asymmetric arrivals, uniform destinations

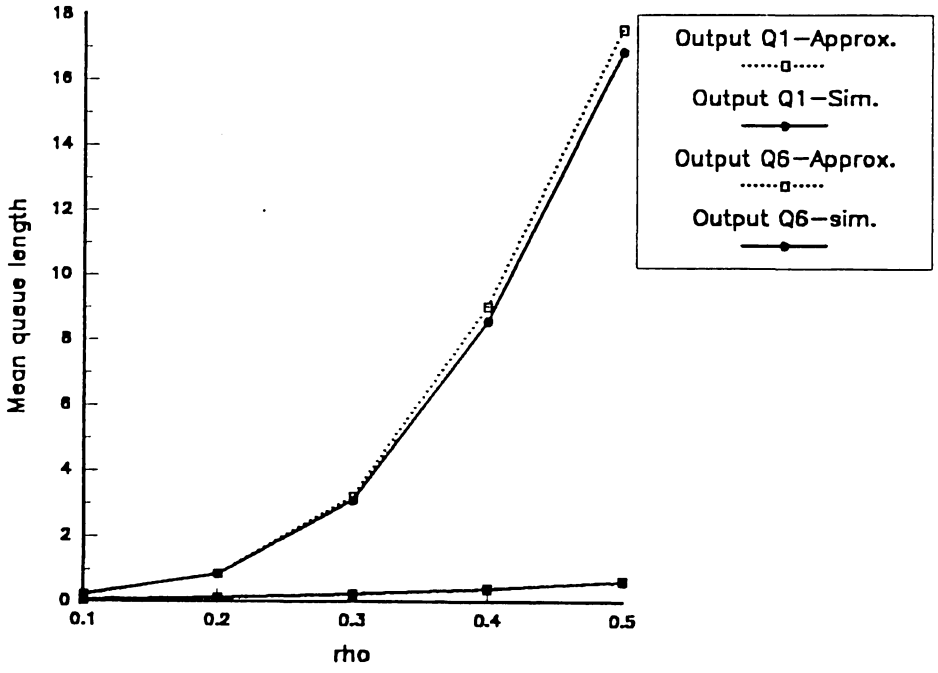


Figure 8. Mean queue length: TDM, nonuniform destinations

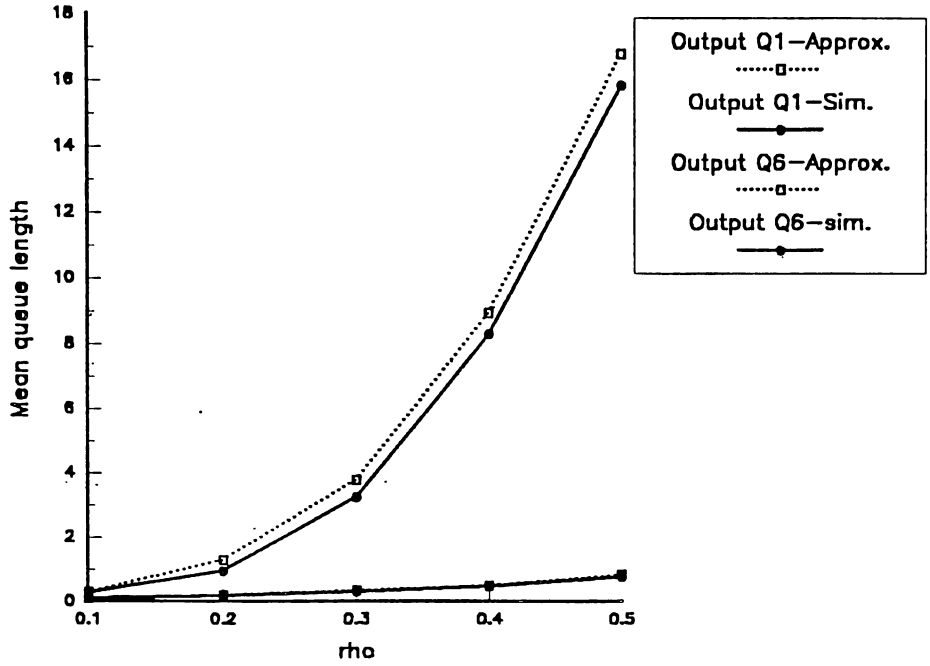


Figure 9. Mean queue length: Random selection, nonuniform destinations

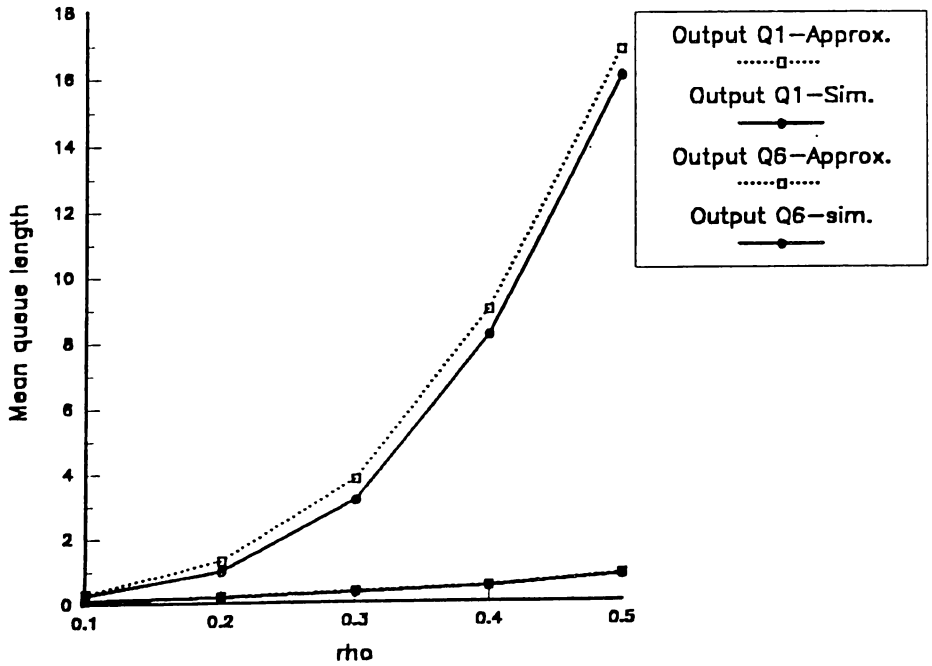


Figure 10. Mean queue length: Cyclic, nonuniform destinations

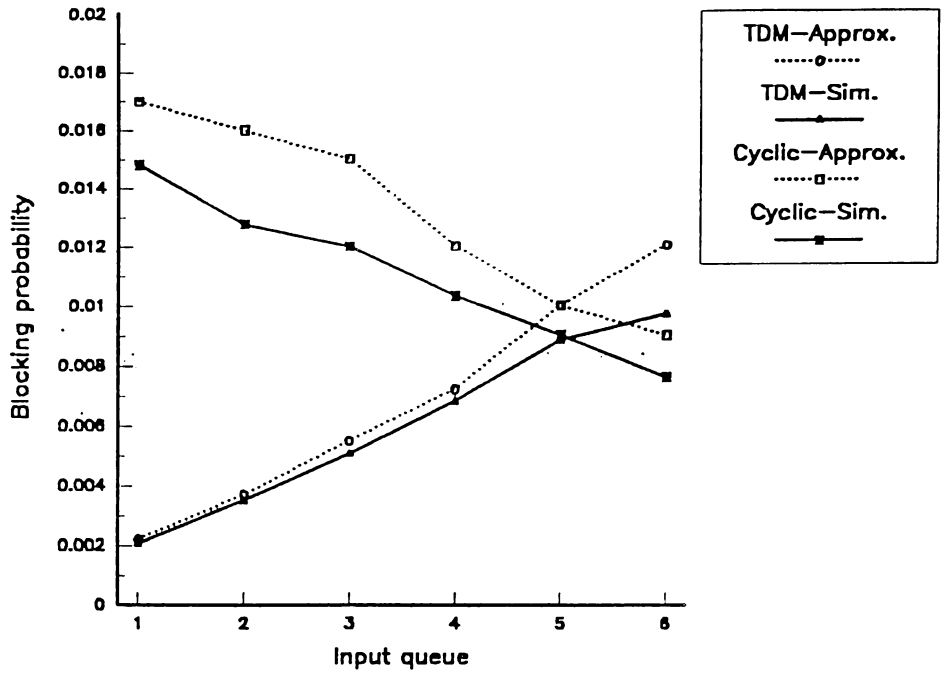


Figure 11. Blocking probability: asymmetric arrivals, nonuniform destinations