Approximate Analysis of a Discrete-Time Polling System with Bursty Arrivals

Atef O. Zaghloul

Harry G. Perros

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

TR-92/23
December 1992

Approximate Analysis of a Discrete-Time Polling System with Bursty Arrivals

Atef O. Zaghloul and Harry G. Perros

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 discrete-time polling system that consists of N input links and a single output link. Each of the input links has a finite buffer to receive incoming traffic. Traffic is assumed to be bursty and is modelled by an Interrupted Bernoulli Process. A shared medium such as a high speed bus is used to forward incoming traffic to the output link in a cyclic fashion. The output link has a geometric service time. The analysis is based on the notion of decomposition whereby the polling system 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, the 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 and asymmetric load conditions to be taken into consideration in the analysis. The models accuracy is verified through simulation.

1.0 Introduction

Due to the increased interest in ATM based networks, several ATM switch architectures have been proposed and analyzed recently in the literature (see for instance Cidon et al [1], Morris and Perros [6], Nilsson, Lai, and Perros [7], and Tobagi [11]). One of these architectures is the ***shared-medium***. In a shared-medium switch, all packets arriving on the input links are forwarded 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 [9], are two examples of such a switch architecture. A shared-medium packet switch with N input links and M output links could be modeled as a polling system as shown in Figure 1. The N input links 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 arbitration or polling scheme of the system under study. In order to model such a system accurately, it is essential to take into consideration system characteristics such as the finite buffer capacity, bursty input traffic, and nonsymmetric load conditions. Because of the finite buffer space at the input links, arriving cells 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 will analyze a sub-system of the shared-medium architecture shown in figure 1. The first sub-system to be analyzed is shown in Figure 2. This sub-system could be viewed as a ***finite capacity discrete-time polling system.*** The second sub-system is shown in Figure 3 and it could be viewed as a ***finite capacity discrete-time polling system with blocking.*** We will refer to the two sub-systems by ***model 1*** and ***model 2*** respectively. Both sub-systems are analyzed approximately using the notion of decomposition. The polling system is decomposed into smaller sub-systems. First, each of the input queues 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 the output queue in isolation (model 2 only). Each sub-system is analyzed numerically as a Markov chain. The results from individual subsystems are combined together through the means of an iterative algorithm which is repeated until a convergence criterion is met. The approximation algorithm is verified via simulation.

The literature contains hundreds of papers on the subject of polling systems (see Takagi [10]). 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. They assumed symmetric work load, zero switchover time, and limited-1 service order. Rather than analyzing the individual input queues, they analyzed the aggregate queue length distribution which represents the total number of cells in the polling system. Under the assumption that the cell occupancy is uniformly distributed among the input queues, they computed the weighting of the cell occupancy configuration which could cause cell loss using the multiple urn model. Subsequently, from this weighting, the aggregate queue length distribution and the mean delay were calculated. Their approach gives an upper bound for the cell loss probability. Tran-Gia [12] gives an approximate algorithm for polling systems with finite capacity and nonexhaustive 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. Two numerical algorithms are presented; one for exponential service time distribution and another for general service time distribution. Ibe and Trivedi [4] considered finite-population and finite-capacity polling systems. Generalized stochastic Petri nets are used to describe the system. A stochastic Petri net package called the SPNP is used to a) generate the state space, and then b) solve the system of linear equations. Poisson arrivals and exponentially distributed service times are assumed. They were able to analyze symmetric and asymmetric polling systems with different types of service order. 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 with finite population and finite buffer capacity. By using restricted occupancy urn models and partial state description, they constructed a solution which consists of a set of linear equations whose number is a linear function of the total number of buffers in the system.

The polling system analyzed in this paper can also be seen as a queueing network with blocking. A queueing network with blocking is a network of finite capacity queues. The queues may be arbitrarily interconnected, and the queueing network may be open or closed. The flow of customers through a queue may be momentarily stopped if a destination queue becomes full. The blocking and unblocking of a customer is governed by a set of rules called the blocking mechanism. Various blocking mechanisms have been proposed and studied in the literature; among them, blocking-after-service, blocking-before-service, and repetitive service. See Perros [8] for a comprehensive review. 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 is a tandem configuration consisting of M finite capacity queues. The first queue is assumed to be saturated, that is, there is always at least one customer waiting to be served. All servers begin and end service at the same time. The duration of a service is constant and it is taken as the unit time. A customer requires a deterministic amount of service at each server which is equal to one unit of time. If we assume that the servers never breakdown, then there will be no customers waiting in queues 2 to

M. Variability in the service is introduced by allowing a server to break down with some probability. 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 [6] developed an approximation algorithm for the analysis of a buffered Banyan ATM switch. The model is capable of handling bursty input traffic and non-uniform destination distributions. The entire switch is viewed as a discrete-time queueing network with blocking where each node represents a switching element. The switch is analyzed by decomposing it into the individual switching elements. Each switching element is then analyzed independently of the other switching elements. The departure process of the first stage becomes the arrival process of stage two and so on. Three different methods for characterizing the arrival process to an input buffer of a switching element were presented. The parameters of the arrival and service processes of the switching elements were obtained using an iterative scheme. Further references for the analysis of multi-stage interconnection networks can be found in [6].

The paper is organized as follows. In section 2, a description of the two queueing models under study is presented. Section 3 covers the analysis of both queueing models. Numerical results are presented and compared against simulation results in section 4. Finally, conclusions are given in section 5.

2.0 The queueing models under study

As mentioned earlier, there are two systems that are analyzed in this paper. Both models consist of N input links, each with a finite buffer, and a single output link. A shared medium such as a high speed bus is used to forward incoming traffic to the output link in cyclic fashion. Only one customer can use the bus at a time. Once an input queue has been granted the bus, only one customer is transmitted across it. Input queues are served in a cyclic order (i.e. 1, 2, 3,..., N, 1, 2, ...) by the bus. This system is modelled as a cyclic polling system with limited service policy (model 1) and it is shown in Figure 2.

In model 2, the output link service time is geometrically distributed with probability B_p and as a result, a finite capacity output buffer has been added as shown in Figure 3. If the output buffer is full, customers are forced to wait in their respective input buffers until a space becomes available at the output buffer. Once a customer reaches the head of the input queue, it will wait for its turn in the polling cycle. At that time it will be forwarded to the output buffer if there is at least one empty space. If the output buffer is full, then the customer will be blocked and it will wait for its turn during the next polling cycle.

We analyze these two models within the context of ATM. In particular, the two models are discrete-time queueing models, in which the bus is slotted, with a slot size equal to the transmission time of an ATM cell.

The arrival process to each input queue is also slotted, with a slot size equal to the bus slot. The slot boundary of each arrival process lies somewhere in between two successive bus slot boundaries as shown in Figure 4. We assume zero switch-over time which means that when the server 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 arbitration for the bus to be performed in parallel to data transmission. Thus, justifying the zero switch-over time assumption. Furthermore, if the system is empty, i.e. no customers are being served, during the present slot, the polling order will reset to queue number 1. This could easily be changed to a random selection among the N queues or other polling schemes.

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 assumed to be equal to the bus cycle. A 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 [7], 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 queues, a cell arriving to a full input queue is lost. However, once a cell has been received, it will not be lost within the queueing system. In this paper, we assume that α is equal to 1 for each arrival process.

3.0 The analysis

The polling systems described in the previous section are analyzed approximately by decomposing them 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 two components. The actual bus transmission time, which is equal to one bus slot and is independent of the load at the other queues, plus the possible delay caused by bus contention which is dependent on the load of the other queues. 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 with $\alpha = 1$, and it is obtained from the departure process of input queue i. The third sub-system (model 2 only) consists of the output queue. The attempted departure process out of the sub-system representing the bus serves as the input process for the output queue. This attempted departure process is also approximated by an IBP with $\alpha = 1$.

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 queue (queueing model 2 only) 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

Let us first consider queueing model 1. As mentioned earlier, in order to analyze each of the input queues in isolation, we need to modify its original service process. The effective service time for each of the input queues consists of two components, the actual bus transmission time (equal to 1 slot), and the delay caused due to bus contention. Because we have a cyclic polling system, once a cell has arrived to the front of the queue, it may be blocked for 0, 1, 2, ..., N-1 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 5. The upper path in Figure 5 represents the event that a customer at the top of input queue i will be transmitted immediately across the bus without delay. In this event, the effective service time is equal to one bus cycle. The probability of this event is denoted by r_1. The other paths in the figure represent the events that the customer from input queue i finds one or more customers from the other input queues already waiting to be transmitted. In this case, customer i will have to wait for its turn in the polling cycle. This waiting time varies from 1 bus cycle to N-1 bus cycles, where N is the number of input queues. For example, the second path in the figure represents the event where the effective service time is equal to 2 bus slots. During the first slot the customer is blocked at its respective input queue. During the second slot the customer is transmitted across the bus. The probabilities, $r_1, r_2, \ldots, r_N$ are derived from the bus sub-system discussed in the next section.

In queueing model 2, once a cell reaches the head of an input queue i, it will wait for its turn in the polling cycle. At that time it will be forwarded to the output buffer if there is at least one empty space. If the output buffer is full, then the cell will be blocked and will wait for its turn during the next polling cycle. We note that this blocking mechanism is different from the first-blocked-first-unblocked mechanism that has been typically used in queueing networks with blocking-after-service. This leads to representing the effective service time for each of the input queues by the phase-type structure shown in Figure 6. Note that this service process for input queue i is the same as the service process for queueing model 1 with the following exception. After a customer completes his service, he may depart from the input queue with probability $1 - Pb_i$, or he may receive another service with probability Pb_i, where Pb_i is the blocking probability due to a full output queue. This blocking probability is calculated from the steady-state probability of the output queue in section 3.3.

The state of an input queue is fully described by a state vector that keeps track of the input process (idle or active), the number of customers in the node (0, 1, 2, 3, ... , m), and the service phase of the server (1, 2, 3, ... , N). The state vector is denoted $X = \{x_0, x_1, x_2\}$ 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 node, and it takes the values 0, 1, 2, 3, ... ,m, where m is the maximum queue capacity. Variable x_2 represents the number of remaining service slots, and it takes the values: 1 if the cell is being transmitted across the bus, 2 if the cell will exit the system after 2 slots, and so on. The stationary probability vector π of this queue can be obtained by solving the system of linear equations $\pi T = \pi$, where T is the transition probability matrix. In this paper, π was obtained using the Gauss-Seidel method.

As stated earlier, 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.

$$P_L = \frac{\sum_{x_2} [P(0, m, x_2)(1-q) + P(1, m, x_2)p]}{\sum_{x_1, x_2} [P(0, x_1, x_2)(1-q) + P(1, x_1, x_2)p]}$$

Where $P(x_0, x_1, x_2)$ is the steady-state probability that the input queue is in state (x_0, x_1, x_2).

The next step is to characterize the departure process from each input queue. This departure process becomes the offered arrival process to the bus. The departure process from each input queue i is approximated by an IBP with parameters p_i, q_i, and α_i. We assume $\alpha_i = 1$. The the values for p_i and q_i are obtained as follows. The states of the input queue 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 (i.e. the customer is not in the last phase of the service process). 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

$$p_i = \frac{\sum_{X \in D} P(X) \left[\sum_{X' \in D,\, W} t(X \rightarrow X') \right]}{\sum_{X \in D} P(X)}$$

and

$$q_i = \frac{\sum_{X \in I} P(X) \left[\sum_{X' \in I} t(X \rightarrow X') \right]}{\sum_{X \in I} P(X)}$$

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. In this model, any state of the input queue from which it is possible to have a departure is a departure state. All states in which the input queue is empty are idle states. Thus, the set of departure states, D, consists of any state $X = \{x_0, x_1, x_2\}$ where $x_1 > 0$ and x_2

equal to 1. The set of wait states, W, consists of any state where $x_1 > 0$ and $x_2 > 1$. For the idle states, we use only one state in which the queue is empty and the arrival process is in the idle state, i.e. $I = \{0,0,0\}$.

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 in round-robin fashion. After a customer from input queue i receives service equal to one slot (i.e. it departs), another customer from input queue i may join the hypothetical queue with probability p_i. This is the probability that a departing customer from input queue i leaves a nonempty queue. On the other hand, a customer from input queue i who is not in the system during the present 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 were obtained as described in the previous section. (We note that the common bus model could be viewed as a discrete-time machine interference model with the bus representing the repairman and the input links representing the machines).

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 following values: (0), (1), (2), (3), (1,2), (1,3), and (2,3). For example the vector (1,2; 3) represents a bus state during which customer 3 is in service and customers 1 and 2 are waiting in the queue , i.e. requesting the use of the bus. In this system, all servers (the bus and the output link) are synchronized so that they start and end service at the same time. The service time is assumed to be equal to one slot. At the beginning of each service slot only one customer is selected among the input queues that have requested the bus. At the end of the slot, this customer will be transferred to the output queue provided that there is at least one available space. This sub-system is analyzed as a Markov chain using the Gauss-Seidel method.

The next step is to derive the effective service time distribution for each input queue. This is calculated by marking an arriving customer and following him until he departs. The distribution of the time that a customer 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 customer is in service.

The distribution of the first passage time between arbitrary pairs of states j and k is defined by,

$$f_{jk}^{(n)} = P(Y_n = k, Y_{n-1} \neq k, \ldots, Y_1 \neq k \mid Y_0 = j),\; n = 1,2,\ldots \qquad (3)$$

where Y_n is the state of the system at transition n. The effective 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 customer from input queue i is in service.
- Q_i = set of states in which a customer from input queue i is requesting service, i.e. the customer is waiting in the bus queue.
- N_i = set of states in which a customer from input queue i is not in the system.

The probability that the effective service for a customer 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 N_i. The service states for input queue i are those states in which a customer 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 service time for input queue i is n bus slots is calculated from equation (3) 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 $t(Y \rightarrow Y')$ be the transition probability from state Y to state Y'. Then, the probability $f_i^{(n)}$ that the effective service time for input queue i is n bus slots is:

$$f_i^{(1)} = \frac{\sum_{Y \in S_i, N_i} P(Y)\lambda_i(Y)\left[\sum_{Y' \in S_i} t(Y \rightarrow Y')\right]}{\sum_{Y \in S_i, N_i} P(Y)\lambda_i(Y)}$$

$$f_i^{(2)} = \frac{\sum_{Y \in S_i, N_i} P(Y)\lambda_i(Y)\left[\sum_{Y' \in Q_i} t(Y \rightarrow Y') \sum_{Y'' \in S_i} t(Y' \rightarrow Y'')\right]}{\sum_{Y \in S_i, N_i} P(Y)\lambda_i(Y)}$$

and for n > 2,

$$f_i^{(n)} = \frac{\sum_{Y \in S_i, N_i} P(Y)\lambda_i(Y)\left[\sum_{Y' \in Q_i} t(Y \rightarrow Y') \sum_{Y' \in Q_i} t^{n-2}(Y' \rightarrow Y') \sum_{Y'' \in S_i} t(Y' \rightarrow Y'')\right]}{\sum_{Y \in S_i, N_i} P(Y)\lambda_i(Y)}$$

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

The next step is to characterize the departure process from the system bus. In this analysis, we use the attempted departure process rather than the actual departure process. This attempted departure process becomes the offered arrival process for the output queue. This attempted departure process is approximated by an IBP with parameters p_d, q_d, and α_d. We assume that $\alpha_d = 1$. The the values for p_d and q_d are obtained using the same method as in the case of the input queues described in section 3.1. However, since we are characterizing the attempted departure process, the states of the bus are divided into only 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. 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

$$p_d = \frac{\sum_{Y \in A} P(Y)\left[\sum_{Y' \in A} t(Y \rightarrow Y')\right]}{\sum_{Y \in A} P(Y)}$$

and

$$q_d = \frac{\sum_{Y \in I} P(Y) \left[\sum_{Y' \in I} t(Y \rightarrow Y') \right]}{\sum_{Y \in I} P(Y)}$$

Where the set of active states A consists of any state $Y = (W;S)$ with $S \neq 0$, and the set of idle states I consists of the state (0, ..., 0; 0).

3.3 Analysis of the output queue

The arrival process to the output queue is modeled as an IBP with $\alpha = 1$. 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), and the number of customers in the queue. The output queue state vector is denoted $Z = \{z_0, z_1\}$ 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 output queue is analyzed as a Markov chain using the Gauss-Seidel method.

The next step is to calculate the blocking probabilities for each of the input queues. The blocking probability for input queue i is the probability that the output queue is full given that the bus is about to serve a customer from input queue i. This probability is approximated by the probability that the output queue is full given that an arrival from input queue i occurs. Let F be the set of all states in which the output queue 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 and N_i is the set of states of the bus in which a customer from input queue i is not in the system. Then we have

$$Pb_i = \frac{\sum_{Z \in F} P(Z) \left[\sum_{Y \in S_i} P(Y) + \sum_{Y \in N_i} P(Y)(1 - q_i) \right]}{\sum_{Y \in S_i} P(Y) + \sum_{Y \in N_i} P(Y)(1 - q_i)} .$$

3.4 Summary of the algorithm

The polling system is analyzed by decomposing it into smaller sub-systems. First, each of the input queues is analyzed separately after its service process has been modified. Consequently, the bus is analyzed as a separate sub-system utilizing the output process of each input queue. Finally, the 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_1, r_2, ... , r_N, and the blocking probability Pb_i (queueing model 2 only).

Step 2. Analyze numerically each of the input queues. 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 of the input queues. This new service time distribution is used in step 2 in the next iteration. Calculate new values for p_d and q_d that characterize the attempted departure process from the common bus.

Step 4. Analyze numerically the output queue. Calculate the queue length distribution. This is used to update the blocking probability Pb_i in the next iteration. This step is needed for queueing model 2 only.

Step 5. Repeat steps 2, 3, and 4 until the convergence test criteria have been met. The convergence test compares the service time distribution ($r_1, r_2, \ldots, r_N$). for each input queue from one iteration to the next.

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

4.0 Numerical results

The approximation algorithm described in the previous section was implemented on an IBM 3090. The main performance measures obtained by the model were the queue length distribution, throughput, and cell loss probability. The approximate results were compared to results obtained by simulation. A representative sample of these comparisons is given for a polling system with six input queues and $B_p = 0.1$. Since model 1 is a subset of model 2, we show results for model 2 only. Results presented in this section were obtained for a system with one queue size for all queues (input and output). Confidence intervals for the simulation results are very small and therefore, are not plotted.

Figures 7 to 10 have been obtained assuming symmetric load. Under symmetric load, the statistics for all input queues is the same, and therefore we will present the data for one input queue only. Figure 7 gives the mean input and output queue lengths for a polling system with queue size equal to 16 as a function of the arrival rate ρ. We note that there is a good agreement between the approximation results and the simulation data. Maximum, minimum, and average relative errors for results in figure 7 are 12.5%, 0.0%, and 6.4% respectively. Figure 8 gives the output queue length distribution for a system with queue size equal 8 under highly bursty traffic ($C^2 = 200$). We note that the queue length distribution is bi-modal. The two peaks correspond to the case where the queue is empty and the case where the queue is full. This is due to the bursty arrival process. Maximum, minimum, and average relative errors for results in figure 8 are 18.0%, 0.0%, and 9.4% respectively (note that the large relative errors are not indicative of the algorithm's accuracy since they correspond to very small approximation and simulation values). Figure 9 gives cell loss probability for a polling system with two queue sizes, 8 and 16, as function of C^2. We observe that as the burstiness of the arrival process increases (higher values for the squared coefficient of variation), the relative error increases; however, as long as C^2 is below 100, the relative error remains below 15 %. Maximum, minimum, and average relative errors for results in figure 9 are 14.0%, 0.0%, and 7.3% respectively. Figure 10 gives system throughput for a polling system with queue size equal to 16 as a function of C^2 The aggregate arrival rate is 0.6 ($\rho = 0.1$ per input link). We note that the system throughput drops sharply as C^2 is increased from 1 to 60. As C^2 continues to increase, the system throughput stabilizes around 0.5; however, cell loss probability will continue to increase as C^2 increases beyond 60 as shown in figure 9. We note that there is a good agreement between the approximation results and the simulation data. Maximum, minimum, and average relative errors for results in figure 10 are 2.4%, 0.0%, and 1.4% respectively.

Figures 11 to 13 have been obtained assuming a polling system with queue size equal to 32 under asymmetric arrival process ($\rho_1 = 0.025$, $\rho_2 = 0.05$, $\rho_3 = 0.075$, $\rho_4 = 0.10$, $\rho_5 = 0.125$, $\rho_6 = 0.15$). Figure 11 shows mean input queue lengths for two values of C^2, 20 and 200. Maximum, minimum, and average relative errors for results in figure 11 are 9.0%, 0.0%, and 4.1% respectively. Figure 12 gives cell loss probability for each input link. Maximum, minimum, and average relative errors for results in figure 12 are 4.4%, 0.2%, and 2.5% respectively. Figure 13 gives the throughput for each input link. Maximum, minimum, and average relative errors for results in figure 13 are 1.4%, 0.0%, and 0.2% respectively. We note that there is a good agreement between the approximation results and the simulation data for the three performance measures: mean queue length, cell loss probability, and throughput.

In general, the accuracy of the algorithm is good. It was observed that if the queue size is small (i.e. less than 8) the results are not very good when C^2 is high (i.e. greater than 100). When the queue size is large (i.e. greater than 32) the accuracy is good for values of C^2 up to 200.

5.0 Conclusions

In this paper we presented an approximation analysis of a discrete-time polling system that consists of N input links and a single output link. Each of the input links has a finite buffer to receive incoming traffic. A shared medium such as a high speed bus is used as a central server to forward incoming traffic to the output link in a cyclic fashion. In queueing model 1, the output link has deterministic service process. In queueing model 2, the output link has geometric service process.

The analysis is based on the notion of decomposition where the polling system is decomposed into smaller sub-systems. First, each of the input queues is analyzed numerically in isolation after its service process has been revised. Subsequently, the bus is analyzed as a separate sub-system utilizing the output process of each of the input queues. Finally, the output queue is analyzed in isolation (queueing model 2 only). The results from the sub-systems are combined together through an iterative scheme.

This approximation method permits realistic system characteristics, such as a limited buffer size and asymmetric load conditions, to be taken into consideration. The accuracy of the approximation algorithm 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. T. Morris and H. Perros, "Performance analysis of a multi-buffered Banyan ATM switch under bursty traffic", INFOCOM 92, Florence, Italy.

7. A. Nilsson, F. Lai, H. Perrros, "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.

8. 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.

9. 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.

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

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

12. P. Tran-Gia and T. Raith, "Performance Analysis of Finite Capacity Polling Systems with Nonexhaustive Service", Performance Evaluation 9 (1988), pp. 1-16, North-Holland.

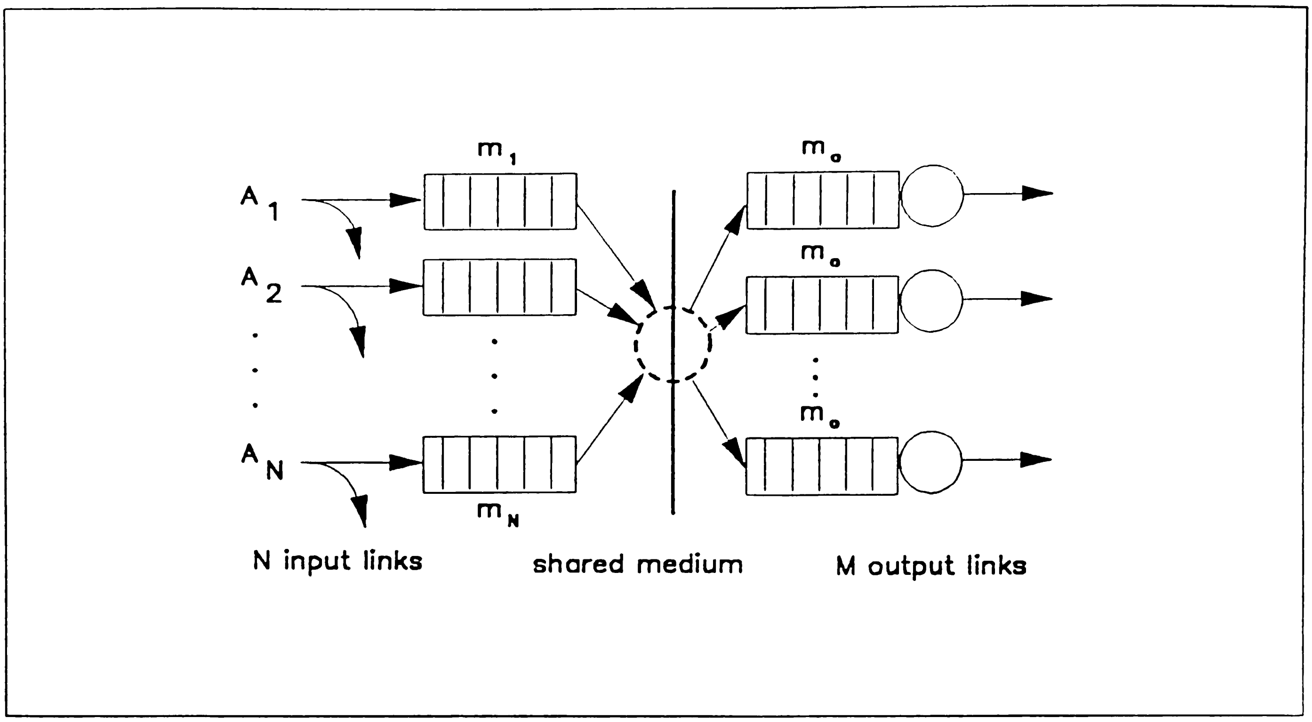

Figure 1. Queueing model of a shared-medium switch

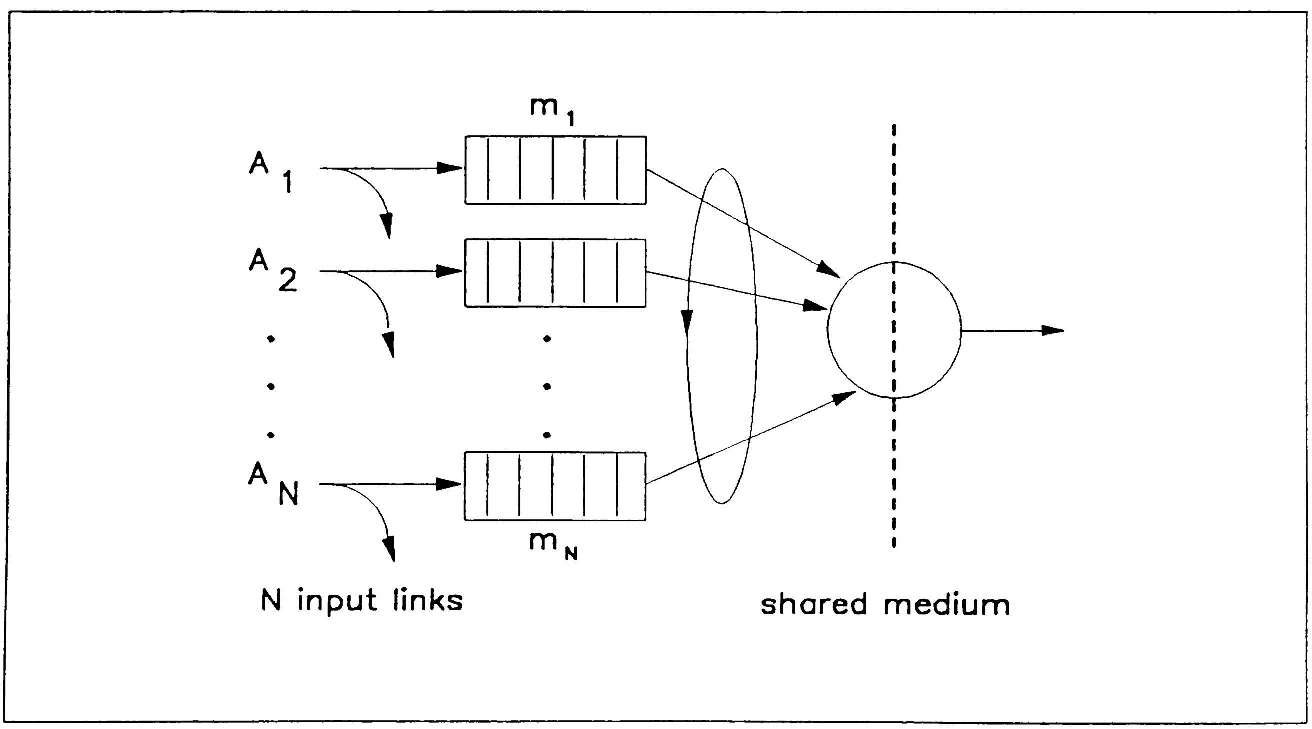

Figure 2. Queueing model 1

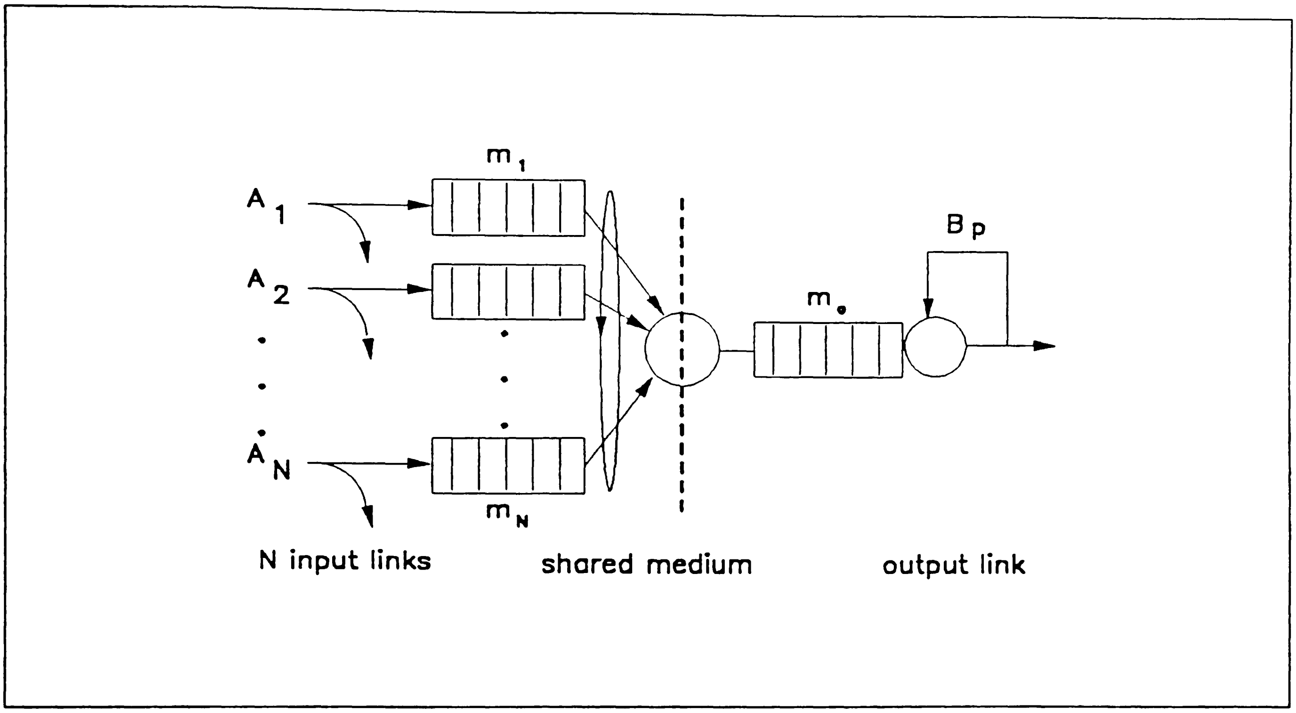

Figure 3. Queueing model 2

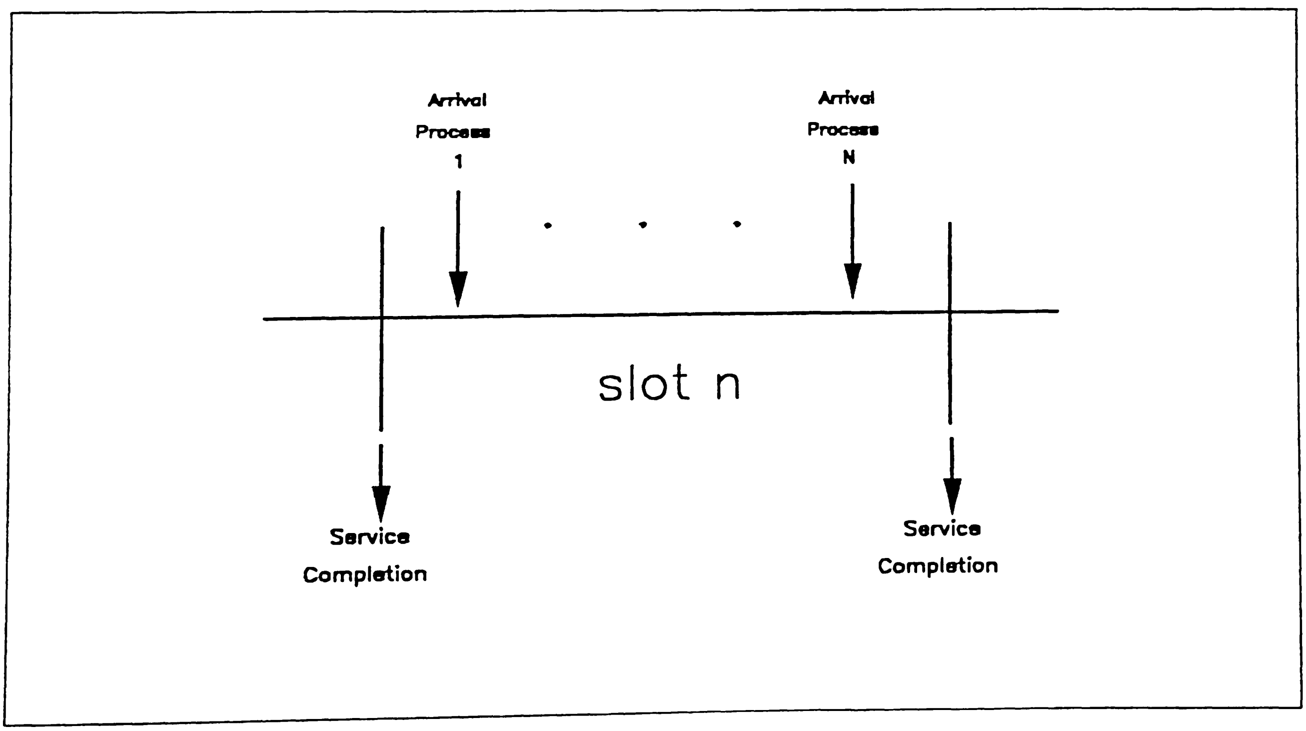

Figure 4. Arrival and departure instants.

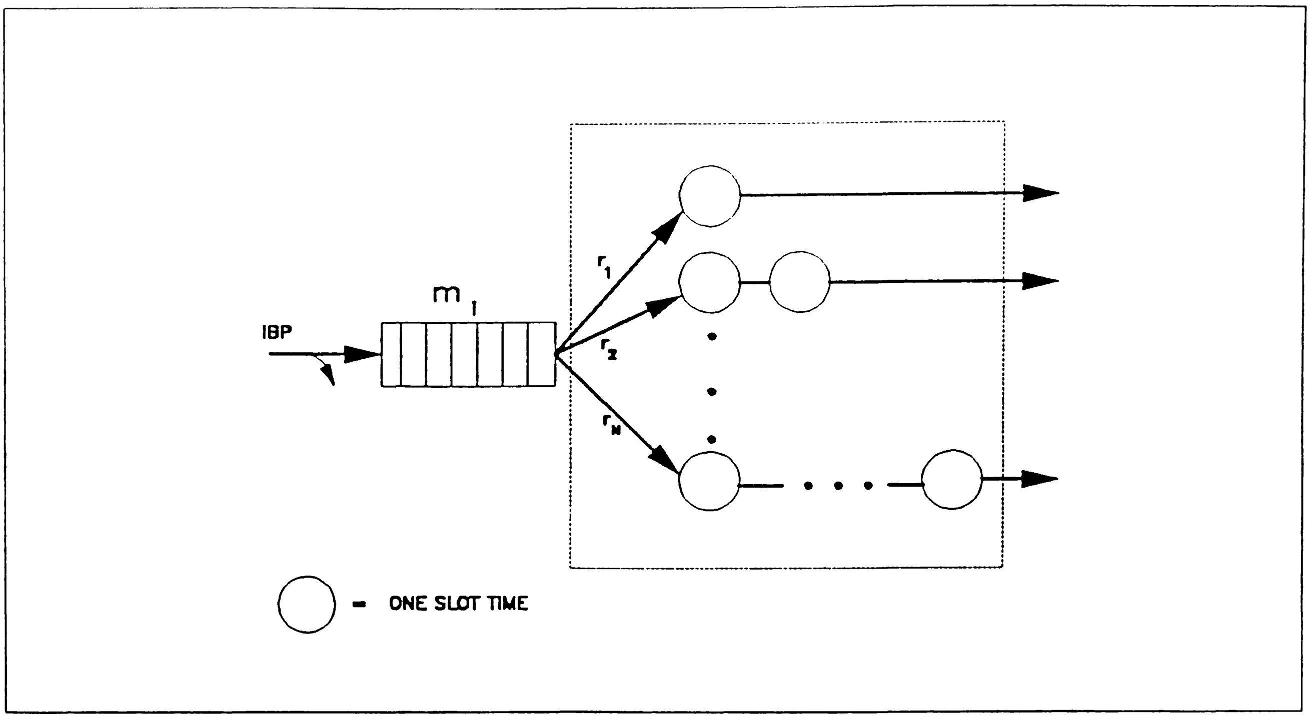

Figure 5. The effective service time of an input queue (model 1)

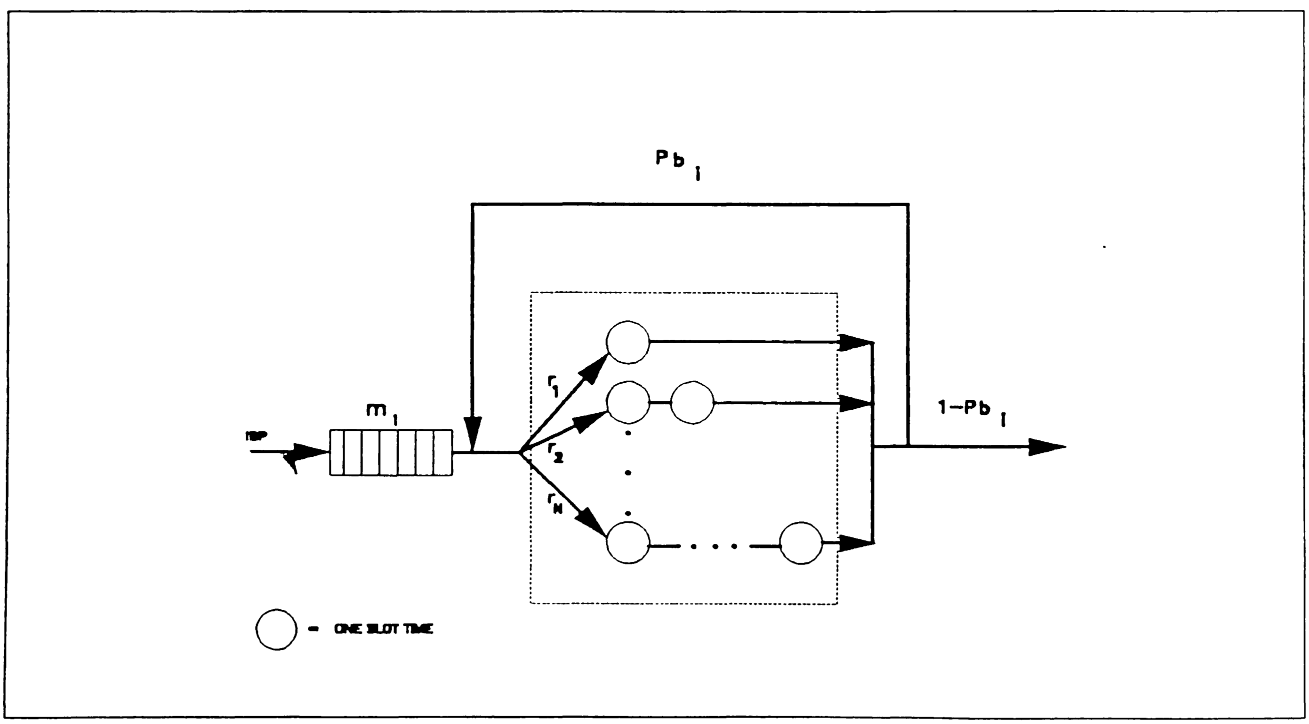

Figure 6. Effective service time of an input queue (model 2)

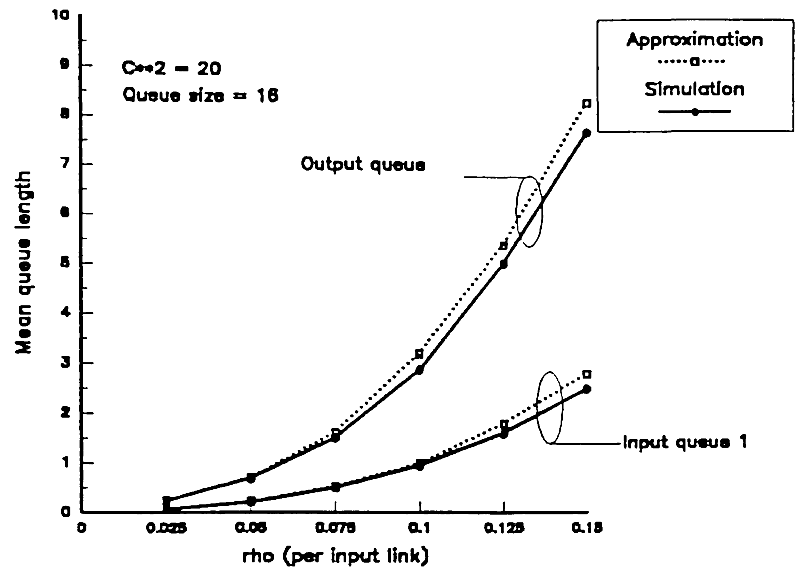

Figure 7. Mean queue length vs ρ (symmetric case)

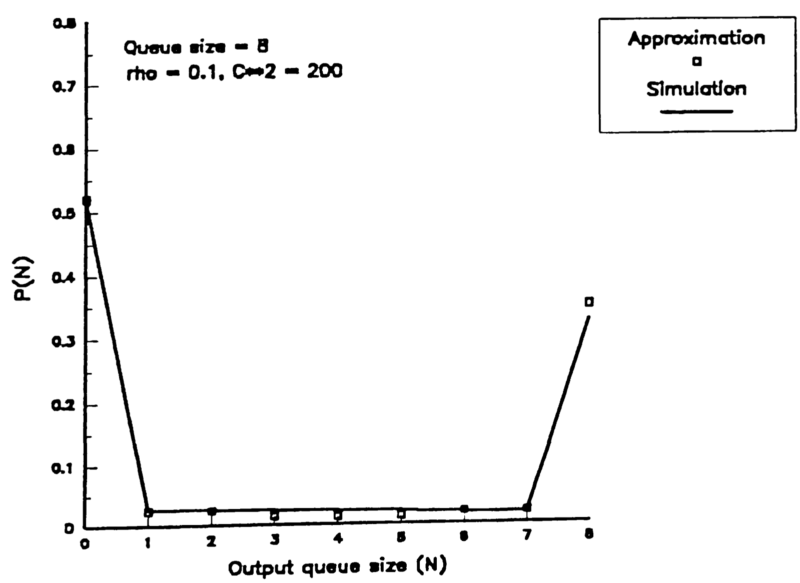

Figure 8. Output queue length distribution (symmetric case)

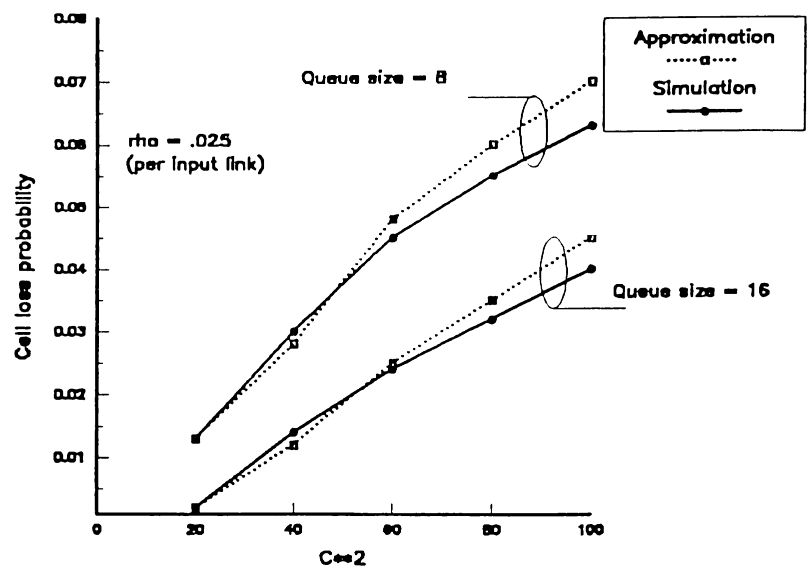

Figure 9. Cell loss probability vs C^2 (symmetric case)

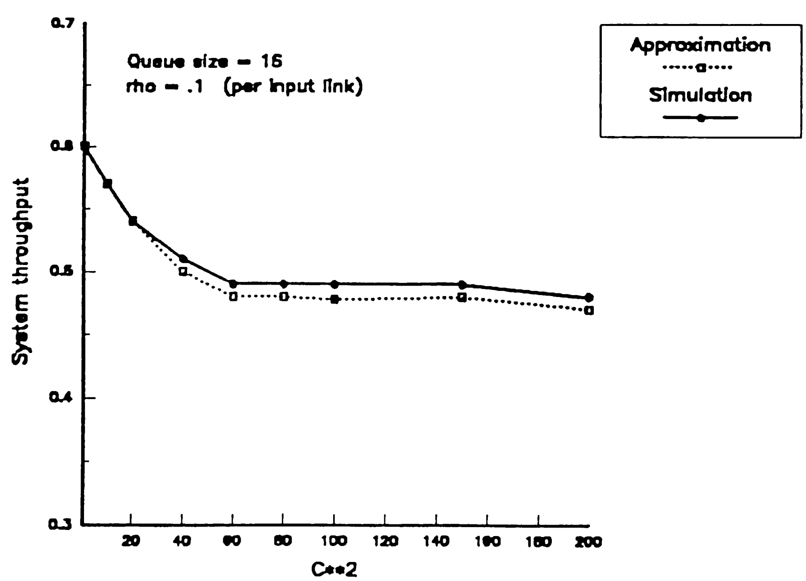

Figure 10. System throughput vs C^2 (symmetric case)

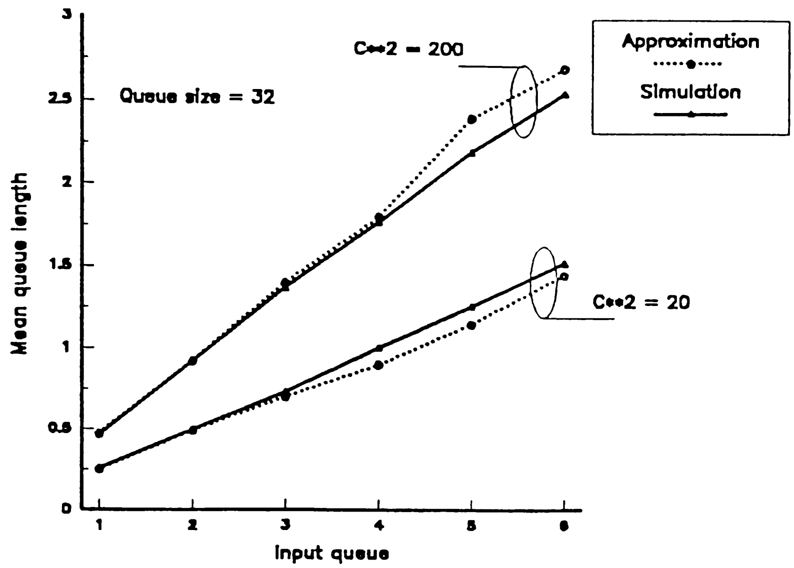

Figure 11. Mean input queue length (asymmetric case)

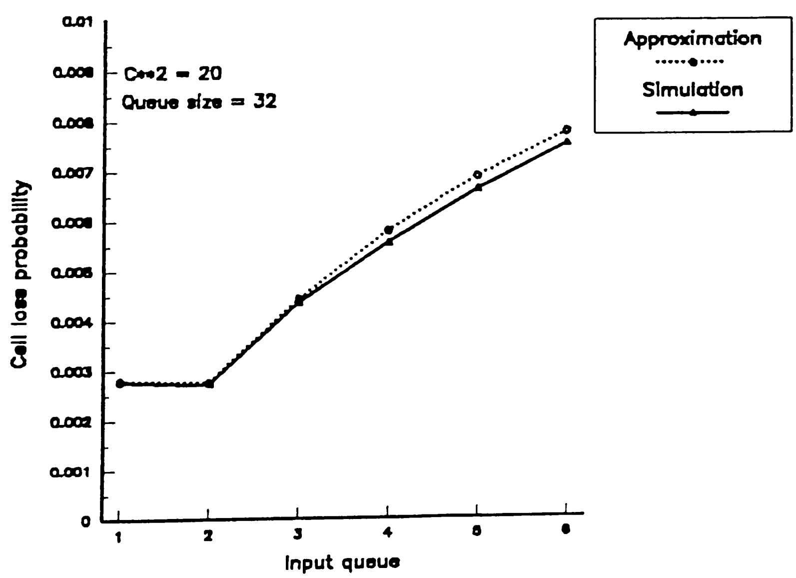

Figure 12. Cell loss probability (asymmetric case)

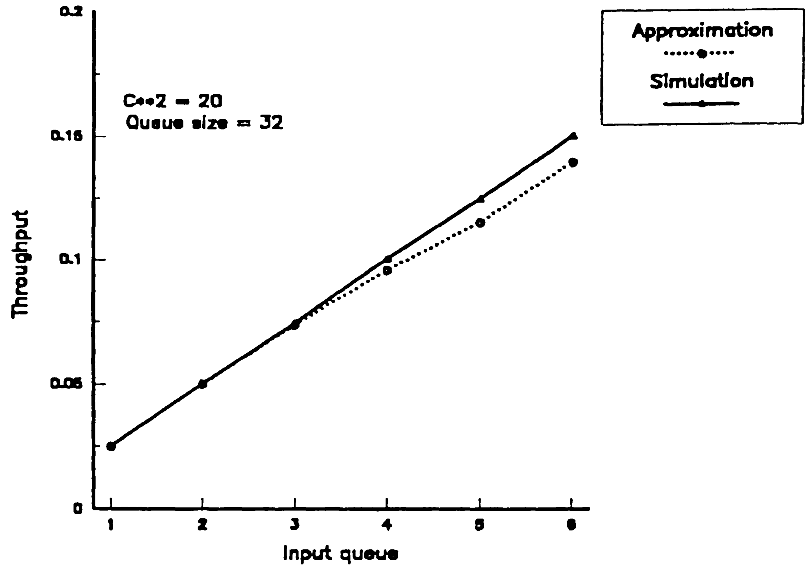

Figure 13. Throughput for each input link (asymmetric case)