Performance Analysis of Finite-Buffered Multistage Interconnection Networks with a General Traffic Pattern*

T. Lin
University of California, Los Angeles
Computer Science Department

L. Kleinrock
University of California, Los Angeles
Computer Science Department

Abstract

We present an analytical model for evaluating the performance of finite-buffered packet switching multistage interconnection networks using blocking switches under any general traffic pattern. Most of the previous research work has assumed unbuffered, single buffer or infinite buffer cases, and all of them assumed that every processing element had the same traffic pattern (either a uniform traffic pattern or a specific hot spot pattern). However, their models cannot be applied very generally. There is a need for an analytical model to evaluate the performance under more general conditions.

We first present a description of a decomposition & iteration model which we propose for a specific hot spot pattern. This model is then extended to handle more general traffic patterns using a transformation method. For an even more general traffic condition where each processing element can have its own traffic pattern, we propose a superposition method to be used with the iteration model and the transformation method. We can extend the model to account for processing elements having different input rates by adding weighting factors in the analytical model.

An approximation method is also proposed to refine the analytical model to account for the memory characteristic of a blocking switch which causes persistent blocking of packets contending for the same output ports. The analytical model is used to evaluate the uniform traffic pattern and a very general traffic pattern "EFOS". Comparison with simulation indicates that the analytical model is very accurate.

*Supported by a grant from DARPA MDA 903-87-C-0663

1 Introduction

Packet-switching Multistage Interconnection Networks (MIN) have been proposed for applications in multiprocessor systems for interconnecting a large number of processing elements (PE) and memory modules (MM), and also for fast packet switching various packets to their destination ports [21]. The performance analysis of MIN's thus becomes an important issue.

A considerable amount of performance analysis has been reported on clocked, packet-switched multistage interconnection networks. Example networks are the Banyan [5] and Omega [13] networks. Most of the previous research was limited to only unbuffered or infinite buffered cases with a uniform traffic pattern or a particular hot spot traffic pattern. Dias and Jump [4] and Jenq [7] analyzed the single buffer case under a uniform traffic pattern. Kim and Garcia [8] analyzed a single buffered network with nonuniform traffic patterns. Kruskal and Snir [9], [10], [11] analyzed the performance of banyan networks with infinite buffer sizes under uniform and a particular hot spot traffic pattern. Szymanski and Shaikh [19], Yoon et al. [23] all presented an analytical model to analyze a finite buffered MIN under a uniform traffic pattern only. Willick and Eager [22] also analyzed the infinite buffered case; their model is able to analyze any given general traffic pattern except that they need a special purpose analysis for each non-uniform pattern. Theimer et al. [20] proposed a technique for modelling the persistent blocking behavior of a single buffered network with a uniform traffic pattern.

In this paper, we present an analytical model for clocked, packet-switched multistage interconnection networks built with 2x2 blocking switches. The model is based on a decomposition and iteration analysis of the Markov chain representing the output queue in each switch. The goal is to provide a simple analytical model which can be applied to evaluate the performance of multistage interconnection networks with

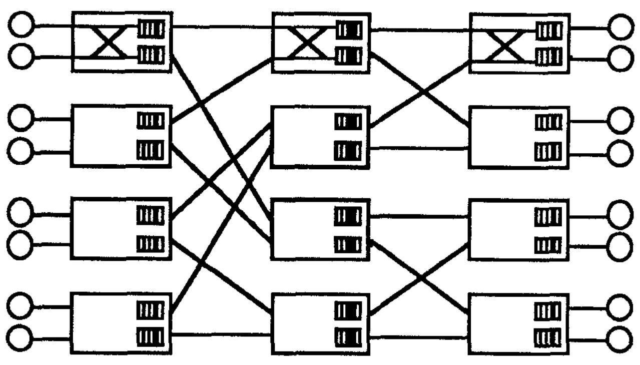

Figure 1: The 3 stage Banyan network with buffers at output ports of each switch

arbitrary switch sizes, arbitrary buffer sizes, arbitrary input rates for each processing element and any general traffic pattern.

This paper is organized as follows. Section 2 describes the model assumptions and the approach. The decomposition and iteration method is introduced with a unique routing probability matrix to represent the steady state traffic flow. A uniform traffic pattern and a special hot spot pattern are discussed using this matrix. A transformation method is introduced in Section 3 to allow any general traffic pattern to be mapped onto the routing matrix to represent the steady state flow. A superposition method is also introduced to allow each processing element to have its own traffic pattern. The resulting memory referencing pattern is even more general; as an example we analyze the Non-Uniform Traffic Spots (NUTS) pattern [12]. A weighting factor is used with the mapping process to handle the case when processing elements are allowed to have different input rates. An approximation method is introduced in section 4 to account for the "memory" characteristics inherent in the blocking scheme. Results are verified through simulation in section 5. Conclusions are given in section 6.

2 Analytical Model for Uniform Traffic and Hot Spot Traffic Patterns

2.1 Architecture Description and Assumptions

In this paper, the interconnection network we consider is a clocked, packet-switched finite-buffered Banyan network made up of 2x2 switches, each of which has buffers of finite size K at their output ports (see Figure 1). There are N processing elements and N memory modules interconnected by the n-stage (i.e. $N = 2^n$) interconnection network. All the operations of input and output take place at the end of each cycle. The interconnection network accepts requests from the input nodes (processing elements), then routes them to the output nodes (memory modules). Responses to these requests are returned from the output nodes through the interconnection network in the reverse direction to the original requesting nodes. The "forward" network and "backward" network are distinct, but are identical in topology. It is sufficient to discuss the delay and throughput performance of the forward network only.

Each packet generated at the processing elements carries an address tag with a number of bits equal to the number of stages of the interconnection network. The address tag is a binary representation of the destination address. It is then fed into the first stage of the network. The first stage switch examines the first (i.e. most significant) bit of the address tag; if it is a 0, the packet is routed to the queue at the upper output port. If the first bit is a 1, the packet is routed to the queue at the lower output port (see Figure 1). The packet then waits in the queue until its turn to be served. The routing process repeats in each stage to choose a proper output port. A blocking switch is assumed in which if a head-of-queue packet cannot go to the next stage due to a full buffer or a "contention failure" for a single available position, it stays at the current queue and waits for the next cycle to try again. The blocking phenomenon has an implied memory characteristic in that a blocked packet will attempt to reach the same output port again. This memory characteristic makes the analytical modelling difficult. We shall discuss an approximation method to be incorporated into the analytical model to model this memory characteristic in later sections.

When two packets from different queues in the same stage contend for the same output queue in the next stage, a contention occurs. If there are more than two spaces available at the output queue, the switch is assumed to be fast enough to accept both packets in one cycle. If there is only one space available, a packet is randomly chosen to fill up this space; the other packet is then "blocked" and stays at the original queue. However, if no space is available in the next stage (i.e. the queue at the output port is full), then both packets are blocked.

Packets are assumed to be of the same length (i.e. fixed size packets). A packet is generated by each processing element independently with probability q in each cycle. All processing elements are assumed to have this identical bernoulli input process. This assumption is later relaxed by using weighting factors to allow each processing element to have its own input rate q_j, $1 \leq j \leq N$. We assume that there is no buffer space at the processing elements. After being generated, a packet is discarded if it cannot be delivered to the first stage of the interconnection network either due to a full buffer or a contention failure. Discarded packets are not re-submitted. A packet, once accepted by the network, is never discarded inside the network. The input process is independent of the discarding process. (An extension of the current model to allow blocked packets to be stored in a finite-sized queue or an infinite queue is underway.) An important performance measure is the total time a packet spends in the network. Time delay is meaningful only for those packets accepted into the network. The probability of acceptance, another performance measure, is the probability that a packet is accepted into the network after it is generated. The normalized throughput is simply the probability of acceptance multiplied by the input rate. Current work also includes an extension to the case of multiple packet generation.

Each processing element has a memory module referencing pattern. A referencing pattern is the set of probabilities with which a packet accesses the various memory modules. All previous work assumes that processing elements have the same referencing pattern. We shall allow PE's to have their own traffic pattern in Section 3. The memory module is assumed to be fast enough to accept 1 packet per cycle from switches at the last stage. This fast memory module assumption implies that there is no blocking at the last stage since a dedicated link connects 1 memory module to the output queue (see Figure 1). A slower memory module (e.g. 2 cycles to accept a packet) will have a severe effect on the performance of the network. Extension to slower memory models is underway.

2.2 Routing Model

In the real world, the packets are routed according to their destination address. However, in order to analyze the network analytically, an abstract flow model that can be used in an analytical model must be established that at least faithfully reflects the steady state flow situation in the network. We propose a routing matrix $r_{i,j}$, $1 \leq i \leq n, 1 \leq j \leq N$ where $r_{i,j}$ is the routing probability of the jth input port in stage i. A packet entering a switch will be routed either to the upper output queue with probability $r_{i,j}$ or to the lower output queue with probability $1 - r_{i,j}$. To simulate a uniform traffic pattern, we simply let all $r_{i,j}$ be 0.5. With equal probability of choosing output queues, no memory module is preferred. A special hot spot pattern can be created by letting all $r_{i,j}$ be an identical value greater than 0.5. For instance, by letting all $r_{i,j}$ be 0.8 in a 10 stage network, 10.7% ($= .8^{10}$) of the total traffic will go to memory module 0 in a 1024-node network with 2.7% of the traffic going to the second highest referenced memory modules (all memory modules with a single 1-digit in their address tag) and other fractions of traffic to the other memory modules. The advantage of this routing model is that by changing the value of $r_{i,j}$ with proper mappings from real traffic patterns, we can evaluate any general traffic pattern. We leave the general $r_{i,j}$ to be discussed in Section 3. Throughout this section, all $r_{i,j}$ are assumed to have the same value, $r_{i,j} = r$.

2.3 Analysis

The proposed approximate analytical approach employs a decomposition and iteration strategy. The real interconnection network is in fact a network of finite-buffered queues with blocking. The dependency among queues, caused by the blocking from stage to stage, makes the exact analysis intractable. We shall use a similar approximation technique as that applied in tandem queues with blocking [2], [3] and [17] where approximate analyses are used. The approximation method is to decompose each queue in the tandem configuration with assumed input rates and blocking conditions. The exact Markov chain is then solved to find the corresponding input rates and blocking conditions. Each decomposed queue is analyzed, then the whole process is repeated until it converges, if it is to have steady state. The concept of using this decomposition and iteration approximation method in analyzing the finite-buffered Banyan network is very similar to that of tandem queues except that instead of a single input source and a single output queue for each queue in the tandem configuration, the interconnection network has 2 input sources and 2 output queues for each queue in the network (except the last stage queue where only 1 output sink is presented, namely, the memory module). Therefore, when we solve for the equivalent input rates and blocking conditions for a decomposed queue, we consider the combined input from 2 input sources and the combined probability of blocking from the 2 output queues. The approach is as follows :

Let $Q_{i,j}$ represent the jth queue in stage i and $P_{i,j}(k)$ be the steady state probability that there are k packets in the queue $Q_{i,j}$. Let $Q_{i-1,j1}$ and $Q_{i-1,j2}$ be the two input sources from stage i-1 that feed $Q_{i,j}$. Let $X[i]$ be the probability that there are i packets destined to $Q_{i,j}$ from its two input sources. In the following, we solve for the equivalent input rates for a queue $Q_{i,j}$ which is located at output port 0 :

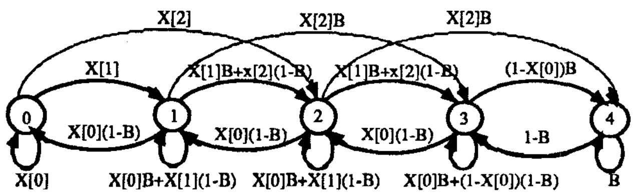

Figure 2: Markov chain of a queue $Q_{i,j}$ extracted from the network where the state variable represents the number of packets in that queue

$$X[1] = 2r(1-r)(1-P_{i-1,j1}(0))(1-P_{i-1,j2}(0)) +$$
$$r[P_{i-1,j1}(0)(1-P_{i-1,j2}(0)) + P_{i-1,j2}(0)(1-P_{i-1,j1}(0))]$$
$$X[2] = [r(1-P_{i-1,j1}(0))] \cdot [r(1-P_{i-1,j2}(0))]$$
$$X[0] = 1 - X[1] - X[2] \tag{1}$$

The first term in the $X[1]$ equation corresponds to the case that both queues in stage i-1 are not empty, and one chooses output port 0 with probability r and the other chooses another output port with probability $1 - r$. The second term corresponds to the case that one queue in the previous stage is empty, and the other is not. The non-empty one chooses the output port 0 with probability r. The summation of probabilities in both cases represents the probability that only one input packet feeds the queue. The $X[2]$ equation represents the case when both queues in the previous stage are not empty and they both choose output port 0 with probability r.

Regarding the equivalent blocking condition, let $B_{i,j}$ be the probability that a packet in the jth queue in stage i is blocked at the end of the cycle. Let $C_{i,j}$ be the probability that the jth queue in stage i is blocking a packet in stage $i-1$. Let $Q_{i+1,j1}$ and $Q_{i+1,j2}$ be the two output queues of $Q_{i,j}$ and let $Q_{i,l}$ be the queue that feeds both $Q_{i+1,j1}$ and $Q_{i+1,j2}$. Then the equivalent blocking condition for queue $Q_{i,j}$ is as follows :

$$B_{i,j} = r \cdot C_{i+1,j1} + (1-r) \cdot C_{i+1,j2}$$
$$C_{i+1,j1} = P_{i+1,j1}(K) + \frac{r}{2} \cdot (1 - P_{i,l}(0)) \cdot P_{i+1,j1}(K-1) \tag{2}$$

The first term in the $B_{i,j}$ equation represents the case when the packet at the head of queue $Q_{i,j}$ chooses $Q_{i+1,j1}$ with probability r and is blocked by $Q_{i+1,j1}$. The second term represents the other case when the packet chooses $B_{i+1,j2}$ and is blocked. There are two situations in which a queue blocks a packet in the preceding stage : firstly, when the queue is full, and secondly, when the queue has only one more space and a contention from $Q_{i,l}$ wins the arbitration.

Given a set of initial values for the variables of the network, we "extract" queue $Q_{1,1}$ from the network (with the equivalent input rates and blocking conditions as exist in the network) as an independent queue. The Markov chain for this queue is then solved to get new values for the state probabilities. A sample Markov chain for the queue $Q_{i,j}$ with buffer size 4 is shown in Figure 2 where B represents the blocking probability $B_{i,j}$. We repeat this process for other queues in the first stage, in the order $Q_{1,2}, Q_{1,3}, ...Q_{1,N}$. Using these new state probabilities as the new input rates, we repeat the same process for all queues in the second stage in the order $Q_{2,1}, Q_{2,2}, ...Q_{2,N}$. This process is repeated for all stages. Now we have a new set of values for the network variables which can be used to compute the new input rates and blocking probabilities. This new set of values is used in the next iteration to compute another set of new values, etc.. The iteration process is repeated until the difference between two consecutive iterations is below 10^{-6}.

The performance measures that are of interest are the probability of acceptance, the normalized throughput and the average time delay. There are two ways to calculate the probability of acceptance. If we sum the output rate over all output ports and divide it by the total input rate, we get the probability of acceptance :

$$PA_{out} = \frac{\sum_{i=1}^{N}[1 - P_{n,i}(0)]}{N \times q} \tag{3}$$

The total output rate over the input rate is the probability of acceptance at the output port. From the input port, we solve for the probability that a packet generated at the PE's is discarded due to a full buffer or a contention failure at the first stage. This discarding probability is $B_{0,j}$, which can be solved for using equation (2). Hence,

$$PA_{in} = 1 - B_{0,j} \tag{4}$$

Both values, although solved in different ways, should be equal when the MIN reaches steady state. (This can be used to test for the correctness of the model.) The normalized throughput is found by multiplying the probability of acceptance by the input rate. We apply Little's result to calculate the average time delay of a packet. When the network reaches steady state, we take the sum of the mean queue size for the whole network using the steady state probabilities of queue size of each queue. Given the throughput and the average number of customers in the system, the average time delay can be solved for by applying Little's result.

2.4 Results

The analytical results of a finite-buffered multistage interconnection network under a uniform traffic pattern and a hot spot pattern were shown in [14]. For various traffic loads and various network sizes (from a single stage to a 9-stage Banyan), the improvement of the probability of acceptance by adding buffers (from unbuffered to buffer size 8) were shown for both the uniform traffic pattern and various hot spot traffic patterns. Another experiment was shown for a 9-stage, 8-buffered Banyan where the average busy buffer size was calculated in each stage. By varying the offered load and the degree of hot spots ($0.5 \leq r \leq 1.0$), we can see how the average busy buffer size grows according to different situations. The average time delay for a 9-stage, 8-buffered Banyan network was shown with various offered loads and various degrees of hot spots. A tree build-up time, defined as the time for the saturated tree [18] to build up, was discussed and a method for calculating the upper bound was shown using the analytical model. The analytical model can also be extended [14] to analyze the performance of the combining network suggested in the NYU Ultra computer project [6] and [18]. The effectiveness of a combining switch was analyzed against various offered loads and queue sizes for a 9-stage combining network. It was shown that the combining switch works only when there is a "hot" memory cell inside the hot spot memory module.

3 Analytical Model for General Traffic Conditions

In this section, we consider very general traffic conditions where not only a general traffic pattern is allowed, but also each processing element can have its own traffic pattern and its own input rate. The basic decomposition and iteration model remains as the main modeling approach. The additional analysis which is needed for general traffic conditions is reduced to finding the proper representation of the steady state traffic flows in terms of the routing probabilities $r_{i,j}$.

3.1 Model Assumptions

We still consider a clocked, finite-buffered multistage interconnection network as discussed in the previous section. All assumptions made in section 2 remain the same except :

- Any general memory referencing pattern is allowed.

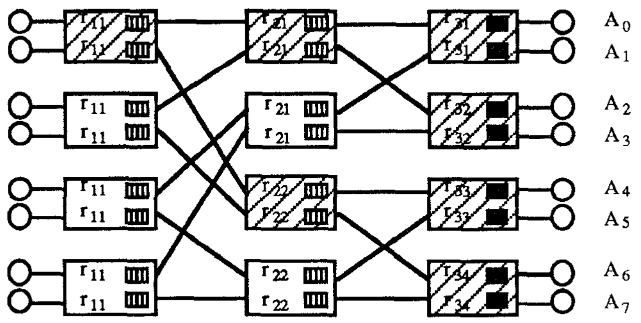

Figure 3: A General memory referencing pattern shown in terms of accessing probabilities A_j

- Each processing element can have its own referencing pattern.
- Each processing element can have its own input rate.

These three different assumptions represent different levels of general traffic patterns. We shall discuss the modeling approaches for these three different assumptions in the next subsections.

3.2 Identical General Traffic Patterns for the Processing Elements

A traffic pattern that can be in any form implies that the $r_{i,j}$'s in the routing matrix no longer have the same value r as discussed in the previous section. The approach to model this general traffic pattern is to find a mapping scheme that transforms the given referencing pattern into a set of $r_{i,j}$'s which reflects the steady state traffic flow in the network.

Let us take a 3 stage Banyan network as an example, as shown in Figure 3. Since we assume that all processing elements have the identical general traffic pattern, we only discuss the transformation method for one processing element. If there exists a steady state referencing pattern, we can represent it in terms of destination accessing probabilities A_j, the probability that a new packet generated by a processing element chooses memory module j as its destination. Consider a packet generated by processing element 0 and observe the path it takes as it travels through the network to access the memory modules. A packet chooses memory module 0 with probability A_0 which equals $r_{11} \cdot r_{21} \cdot r_{31}$. Similarly, a packet chooses memory module 1 with probability $A_1 = r_{11} \cdot r_{21} \cdot (1 - r_{31})$. Using these two equations, we find r_{31} in terms of A_0 and A_1.

$$r_{31} = \frac{A_0}{A_0 + A_1}$$

The other routing Probabilities can be found in a similar way :

$$r_{32} = \frac{A_2}{A_2 + A_3}$$
$$r_{33} = \frac{A_4}{A_4 + A_5}$$
$$r_{34} = \frac{A_6}{A_6 + A_7}$$
$$r_{21} = \frac{A_0 + A_1}{A_0 + A_1 + A_2 + A_3}$$
$$r_{22} = \frac{A_4 + A_5}{A_4 + A_5 + A_6 + A_7}$$
$$r_{11} = \frac{A_0 + A_1 + A_2 + A_3}{A_0 + A_1 + A_2 + A_3 + A_4 + A_5 + A_6 + A_7}$$

Since there is only one traffic pattern for all processing elements, this routing probability set is valid for all other processing elements.

This transformation method is simply a mapping of the memory referencing pattern onto a set of routing probabilities. We incorporate the transformation method into our analytical model approach in section 2. When we calculate the equivalent input rates and blocking probabilities, we replace the value r in equations (1)-(2) with the proper $r_{i,j}$ values. Thus, the decomposition and iteration model we proposed in section 2 can be extended to analyze a multistage interconnection network with any general traffic pattern.

3.3 Different General Traffic Patterns for Processing Elements

For more generality, we allow each processing element to have its own general traffic pattern. In practice, it is quite possible that the traffic requirement for each processing element is different from the others. Let us assume that they all have the same input rate. The modelling approach for this case is to find a proper set of routing probabilities $r_{i,j}$ which reflects the combined traffic flow in steady state. A superposition method is proposed, in addition to the use of the transformation method, to find this proper set of routing probabilities. We first apply the transformation method to all processing elements to transform their general memory referencing patterns into N sets of routing matrices. Since all processing elements have the same input rate, we simply take the mean value of these N sets of routing probabilities to find a routing matrix that reflects the combined traffic flows in steady state. This routing matrix is used in the decompostion and iteration model. Thus, the analytical model can be extended to analyze the case where each processing element has its own traffic pattern.

3.4 Different Input Rate and Different Traffic Patterns for the Processing Elements

In many cases, the processing elements might have different packet input rates. This presents the most general traffic condition. The modelling approach, again, is reduced to finding the proper representation of the routing matrix.

A slower source contributes less to the steady state flows. Hence we must determine a weighting factor for each source to reflect its contribution to the steady state traffic. The ideal weighting factor is the input rate of each source. When we take the mean of the N routing matrices, we then weight their contribution according to their input rates. If we incorporate this weighting factor, the model can be extended to analyze the case where each processing element has its own traffic pattern and its own input rate.

3.5 Results

Since the proposed analytical model employs several approximate methods, it is important to study how these approximations affect the model accuracy. There are two approximations in the modelling approach :

- decomposing a queue from a network of queues with blocking into an independent queue.
- using a general routing matrix to model the steady state flows.

The first approximation is obvious since dependent queues are decomposed into equivalent independent queues and solved individually. Some accuracy is lost because our model neglects the dependency and coupling among the queues. The second approximation allows packets to choose their output ports every cycle independently according to the routing probabilities, instead of the real-world address tag. This renewal routing choice allows a blocked packet to choose a different output port in the next cycle. This renewal assumption renders the analytical model optimistic since it "allows" blocked packets to be routed around a congested queue. In the real world, blocked packets repeatedly access the same destination, and most likely, these blocked packets will be blocked again (especially when the traffic is not uniform).

The EFOS (Even-First-Odd-Second) pattern was proposed in [12] using an Omega network where even addressed processing elements send all their traffic to the first half of memory modules uniformly while the odd addressed ones send their traffic to the second half of memory modules uniformly. The destination traffic

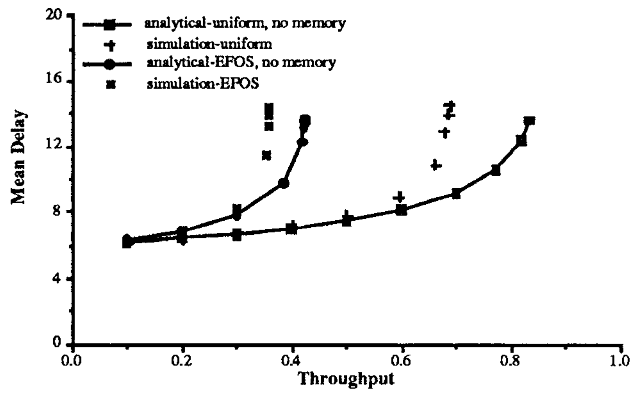

Figure 4: Comparison of results for a 6 stage, 4-buffered Omega network

distribution looks uniform, but there are severe contentions for the common paths inside the network. It is an example of our general traffic pattern where there are two traffic patterns for the processing elements. A 6 stage Omega network with 4 output buffers at each switching element was evaluated under both a uniform traffic pattern and an EFOS traffic pattern. The analytical results we obtained are plotted against simulation results in Figure 4. As predicted , the analytical model is very optimistic due to the independent routing choices it allows. When severe blocking is present due to contention, the blocked packets will choose the same output queues repeatedly in the real world while the renewal choice in the analytical model allows the blocked packets to choose other queues. This inherited "memory" structure in blocking switches severely degrades the performance since it is likely to have persistent contention for a queue once contention occurs. The discrepancy between analytical and simulation data is caused mainly by this memory characteristic of the blocking switch. We propose an improvement in the next section to model this "memory" behavior of a blocking switch.

4 Analytical Model for a Blocking Switch with Persistent Blocking

4.1 Model Approach

Since the basic model is a renewal process, we continue to model the memory behavior as a renewal process. However, the behavior of a blocked packet, after its first blocking, is such that the routing choice no longer uses the renewal probability $r_{i,j}$. Biasing the routing probabilities to account for this does not help since it

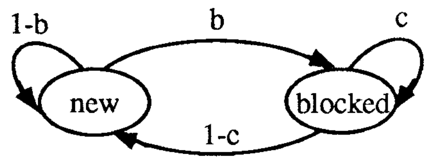

Figure 5: The states of a server during its busy period.

changes the memory referencing pattern. The routing probabilities were created to reflect the steady state memory referencing pattern; therefore, it is necessary to keep the values unchanged.

Although an exact model of this persistent blocking behavior would require that we keep track of how many times a packet has been blocked at a given node, we choose an approximation which captures the "first order" effect of this persistence using the following two state model. When the queue is not empty : we model the server as being either in the "new" state or the "blocked" state. When a packet first comes into the server, the server is in the new state. The server enters the blocked state when the packet is blocked, and it remains in the blocked state until the blocked packet finally goes through to the next stage. This cycle repeats until the server empties the queue and becomes idle. Observe that the server is inactive when it is in the blocked state. While in the new state, the server obeys the renewal behavior choosing an output port according to the routing probability $r_{i,j}$. Hence, we can approximate a blocking switch with "memory" characteristics by a finite buffer queue with a reduced service rate. The reduced portion is the probability that the server is in the blocked state.

The diagram in Figure 5 shows how the server alternates between the new state and the blocked state during its busy period. Let b be the probability that a new packet is blocked when it tries to go to the next stage. Let c be the probability that a blocked packet is blocked again when it tries to go to the same destination. Then for our approximation, the steady state probability that the server is in the blocked state, $P_{blocked}$, can be solved in terms of b and c :

$$P_{blocked} = \frac{b}{1 - c + b}$$

where b is the blocking probability (for which we used the notation $B_{i,j}$ in section 2.3). Once blocked, it is more likely that a blocked packet gets blocked again; therefore, the value of c is selected to be larger than the value of b. In fact, when a packet is in the blocked state, the length of the destination queue in the next cycle will be either K (full) or K-1 (only one space available). If we disregard how many times it has been blocked previously, there will be only two cases : either the blocked packet faces a full queue or a queue with one space left. In the first case, with probability $\frac{P[K]}{P[K]+P[K-1]}$, the packet will be blocked again. In the second case, with probability $\frac{P[K-1]}{P[K]+P[K-1]}$, the packet will face possible contention from the other queue in the same stage which feeds this destination queue. Incorporating these two probabilities in equation (2), c can be found in a similar way :

$$c = r \cdot C_{i+1,j1} + (1-r) \cdot C_{i+1,j2}$$

The probability that the j1-th queue in stage i+1 is blocking a packet in stage i, $C_{i+1,j1}$, is :

$$C_{i+1,j1} = \frac{r}{2} \cdot (1 - P_{i,l}(0)) \cdot \frac{P_{i+1,j1}(K-1)}{P_{i+1,j1}(K) + P_{i+1,j1}(K-1)}$$
$$+ \frac{P_{i+1,j1}(K)}{P_{i+1,j1}(K) + P_{i+1,j1}(K-1)}$$

$P_{blocked}$ is the probability that the server is in the blocked state. During this period, the server is inactive. Therefore, we may use this probability to approximate the blocking switch with "memory" characteristic. At the beginning of each cycle, the server tosses a coin which comes up heads with probability $P_{blocked}$, in which case the server will be blocked (inactive). If there is a packet at the server, it stays idle until the next cycle when the coin will be tossed again. With probability $1 - P_{blocked}$, the server will be active. The queue length then determines whether the server will send a packet or not. If there are packets in the queue, the server takes the first packet and routes it according to the routing probability.

Incorporating the probability $P_{blocked}$ into our previous model, the approach is then similar except that the equivalent input rates and blocking probabilities are different. In the original model, when a queue is not empty (with probability $1 - P_{i,j}(0)$), it tries to transmit a packet to the destination in stage i+1. However, for the persistent blocking model, a queue tries to transmit a packet to the next stage with probability $(1-P_{i,j}(0)) \cdot (1-P_{blocked})$, the former is the probability that the server is not empty and the latter is the probability that the server is in the "active" state. When the server is not empty and it is active, it transmits a packet to the next stage.

Let us define $P^{eff}_{i,j}(0)$ to be the effective probability that $Q_{i,j}$ will not send a packet (either the server is empty or the server is not empty and is blocked). Let $P_{i,j,blocked}$ be the probability that $Q_{i,j}$ is not empty and is in the blocked state. Then the effective input rates of a queue, $Q_{i,j}$ in this persistent blocking model are similar to the ones in section 2.3 :

$$X[1] = 2r(1-r)(1 - P^{eff}_{i-1,j1}(0))(1 - P^{eff}_{i-1,j2}(0))$$
$$+ r[P^{eff}_{i-1,j2}(0)(1 - P^{eff}_{i-1,j1}(0)) + P^{eff}_{i-1,j1}(0)(1 - P^{eff}_{i-1,j2}(0))]$$
$$X[2] = [r(1 - P^{eff}_{i-1,j1}(0))] \cdot [r(1 - P^{eff}_{i-1,j2}(0))]$$
$$X[0] = 1 - X[1] - X[2]$$

$$P^{eff}_{i-1,j1}(0) = 1 - (1 - P_{i-1,j1,blocked}) \cdot (1 - P_{i-1,j1}(0))$$

$$P^{eff}_{i-1,j2}(0) = 1 - (1 - P_{i-1,j2,blocked}) \cdot (1 - P_{i-1,j2}(0))$$

The equivalent blocking probability $B_{i,j}$ can be found as follows :

$$B_{i,j} = r \cdot C_{i+1,j1} + (1-r) \cdot C_{i+1,j2}$$

$$C_{i+1,j1} = P_{i+1,j1}(K) + \frac{r}{2} \cdot (1 - P^{eff}_{i,l}(0)) \cdot P_{i+1,j1}(K-1)$$

Incorporating these equivalent input rates and blocking probabilities into the previous model, we can evaluate a multistage interconnection network with persistent blocking behavior. We first calculate the equivalent input rates and blocking probabilities for each queue in the first stage. Then we decompose the first queue in the first stage and solve its Markov chain with the equivalent input rates and blocking probabilities. The remaining queues in the first stage are decomposed and their Markov chains are solved one by one. The steady state probabilities for these queues are used to calculate the equivalent input rates to the queues in the second stage. The queues in the second stage are then decomposed and solved. This process is repeated for all stages. The analytical model iterates this decomposition process until the throughput converges. Then we calculate the time delay using Little's result. Other performance measures can be computed using the steady state parameters of the system.

4.2 Results

We ran our model incorporating this new technique to handle the memory behavior for the same 6-stage Omega network (as in Section 3.5) with buffer size 4 under both the uniform traffic and the EFOS traffic pattern. The result is shown in comparison with the former model in Figure 6. The improved model greatly reduces the discrepancy between the simulation and analytical model results.

For a detailed study, we compare the improved analytical results with simulation in Figures 7-14. The confidence range of these simulations is 95%. The first case shown in Figures 7-8, is for a 4-buffered, 6 stage

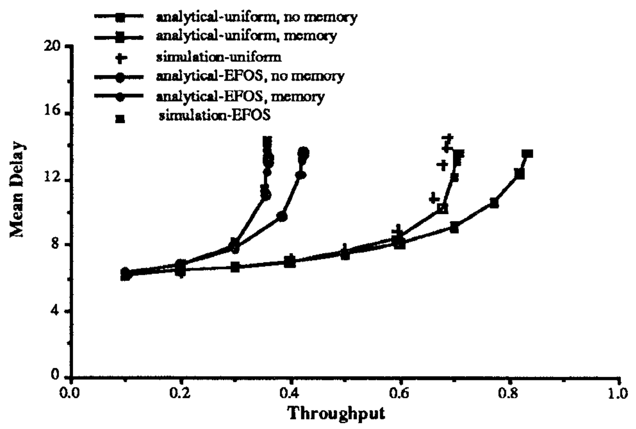

Figure 6: Comparison of results for a 6-stage, 4-buffered Banyan network with and without the "memory" behavior improvement

Omega network with a uniform traffic pattern. In Figure 7, the throughput is compared with various offered loads and in Figure 8, the average time delay is compared. The offered load is varied from 0.1 pkt/cycle to 1.0 pkt/cycle for each processing element. For offered loads within the range of small loss probability ($q \leq 0.7$ pkt/cycle), the simulations verify the accuracy of the analytical results. Beyond this load (when packets begin to be discarded), the analytical results are slightly optimistic.

Figures 9-10 show the case of a 4-buffered, 6 stage Omega network with an EFOS traffic pattern. Throughput and average time delay are plotted against offered load. The non-uniformity of this pattern severely degrades the performance. The throughput graph shows very good correspondence between analytical results and simulations. For offered load within the low-loss range ($q \leq 0.4$ pkt/cycle), delay performance of the analytical result is very accurate. However, delay performance of analytical results are still slightly optimistic in heavy load cases.

The third case is included to determine whether the analytical model performs well with a larger buffer. The results for an 8-buffered, 6 stage Omega network with a uniform traffic pattern are shown in Figures 11-12. Except with heavy load (q=0.9 and 1.0), analytical results measure well when compared to simulations. The throughput and delay performance are optimistic when the total input enters the range of heavy load. This simulation indicates that the analytical model performs well for networks with other buffer sizes.

We show the analytical results of a large sized network in Figures 13-14, namely a 4-buffered, 10 stage (1024x1024) Omega network with a uniform traffic

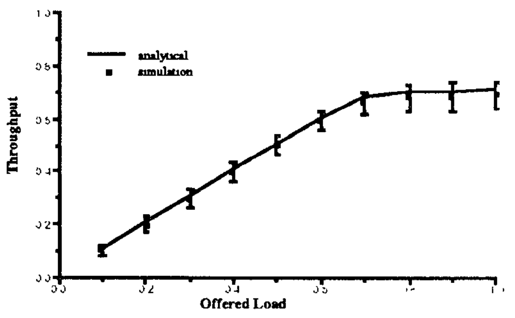

Figure 7: Throughput comparison for a 4-buffered, 6 stage Omega MIN with a uniform traffic pattern

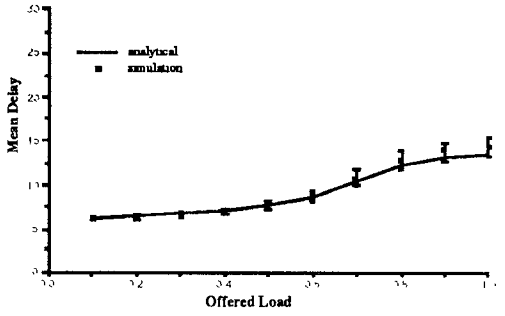

Figure 8: Mean delay comparison for a 4-buffered, 6 stage Omega MIN with a uniform traffic pattern

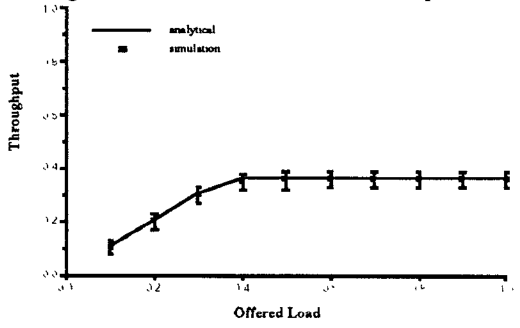

Figure 9: Throughput comparison for a 4-buffered, 6 stage Omega MIN with EFOS traffic pattern

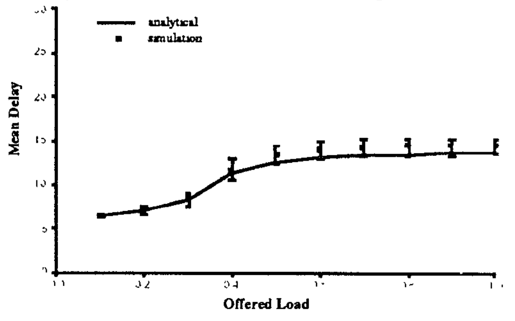

Figure 10: Mean delay comparison for a 4-buffered, 6 stage Omega MIN with EFOS traffic pattern

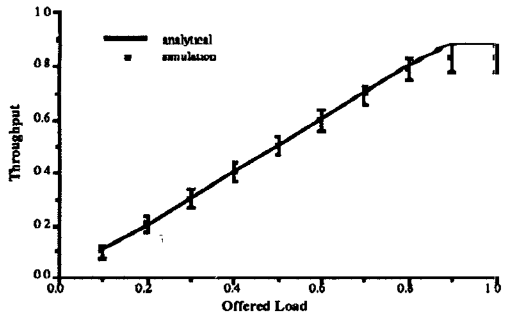

Figure 11: Throughput comparison for an 8-buffered, 6 stage Omega MIN with a uniform traffic pattern

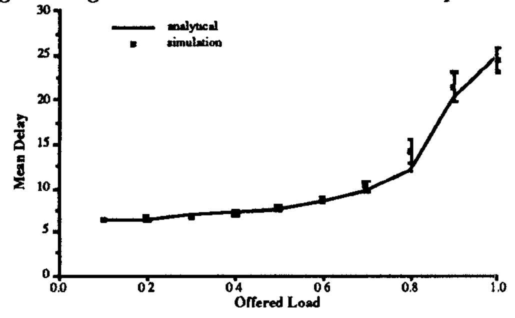

Figure 12: Mean delay comparison for an 8-buffered, 6 stage Omega MIN with a uniform traffic pattern

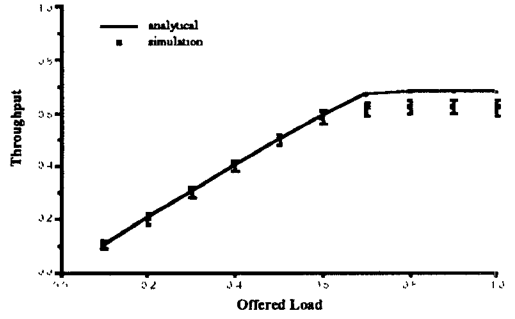

Figure 13: Throughput comparison for a 4-buffered, 10 stage Omega MIN with uniform traffic pattern

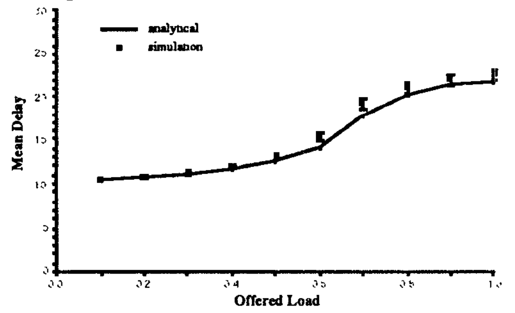

Figure 14: Mean delay comparison for a 4-buffered, 10 stage Omega MIN with uniform traffic pattern

pattern. Again, the analytical model is slightly optimistic when the offered load exceeds the maximal attainable output. This indicates that the model is suitable for large sized networks as well.

4.3 Conclusion

An analytical model was developed to evaluate the performance of finite-buffered Multistage Interconnection Networks with any general traffic pattern. The Interconnection Network was modelled as a network of finite-buffered queues with blocking. We proposed a decomposition and iteration method to analyze this network of queues with blocking. A transformation and superposition method was then proposed to analyze these networks for any general traffic pattern. However, the renewal routing choice assumption caused the modeling results to be optimistic. Therefore, we proposed approximate techniques to model the persistent blocking situation. The analytical results were compared to the simulation in different buffer sizes and different network sizes under both a uniform traffic and a general traffic pattern. The simulations show that the analytical results are very accurate when the offered load does not exceed the maximal attainable total output. For cases where the offered load is beyond the maximal attainable output, the analytical results are slightly optimistic. This verifies the accuracy and the flexibility of the analytical model.

References

[1] L.N. Bhuyan, "An Analysis of Processor-Memory Interconnection Networks ", *IEEE Transaction on Computers*, Vol. c-34, No.3, March 1985, pp.279-283

[2] A. Brandwajn, Y.L. Jow, "An Approximation Method for Tandem Queues with Blocking", *Operations Research*, Vol.36, No.1, January-February 1988, pp.73-83

[3] P. Caseau, G. Pujolle, "Throughput Capacity of a Sequence of Queues with Blocking Due to Finite Waiting Room", *IEEE Transaction on Software Engineering* Vol. SE-5, No. 6, November 1979, pp.631-642

[4] D.M.Dias, J.R. Jump, "Analysis and Simulation of Buffered Delta Networks", *IEEE Transaction on Computers*, Vol.C.30, No.4, April 1981, pp.273-282

[5] L.R. Goke, G.J. Lipovski, "Banyan Networks for Partitioning Multiprocessor Systems", *The Proceedings of the First Annual Symposium on Computer Architecture*, 1973, pp21-28

[6] A. Gottlieb, et al, "The NYU Ultracomputer – Designing an MIMD Shared Memory Parallel Computer", *IEEE Transaction on Computers*, Vol. C.32, No.2, February 1983, pp. 175-189.

[7] Y. Jenq, "Performance Analysis of a Packet Switch Based on Single-Buffered Banyan Network", *IEEE Journal on Selected Areas in Communications*, Vol. SAC-1, No. 6, December 1983, pp.1014-1021

[8] H.S. Kim, A. Leon-Garcia, "Performance of Buffered Banyan Networks under Nonuniform Traffic Patterns", *IEEE Transactions on Computers*, Vol. 38, No. 5, May 1990, pp.648-658

[9] C.P. Kruskal, M. Snir, "The Performance of Multistage Interconnection Networks for Multiprocessors", *IEEE Transaction on Computers*, Vol. C.32, No. 12, December 1983, pp.1091-1098

[10] C.P. Kruskal, M. Snir, A. Weiss, "The Distribution of Waiting Times in Clocked Multistage Interconnection Networks", *Conference of Parallel Processing*, 1986, pp.12-19

[11] C.P. Kruskal, M. Snir and A. Weiss, "The distribution of Waiting Times in Clocked Multistage Interconnection Network", *IEEE Transaction on Computers*, vol. 37, No. 11, November 1988, pp1337-1352

[12] T. Lang, L. Kurisaki, "Nonuniform Traffic Spots (NUTS) in Multistage Interconnection Networks", *UCLA Computer Science Department Technical Report* CSD-880001, January 1988

[13] D.H. Lawrie, "Access and Alignment of Data in an Array Processor", *IEEE Transaction on Computers*, Vol.C-24, Dec. 1975, pp. 1145-1155

[14] T. Lin and A. Tantawi,"Performance Evaluation of Packet-Switched Multistage Interconnection Networks under a Model of Hot-Spot Traffic", *ORSA/TIMS Joint National Meeting*, Philadelphia, PA, October 29-31, 1990.

[15] J.H. Patel, "Processor-Memory Interconnections", *IEEE Transaction on Computers*, Vol. c-30, No. 10, October 1981, pp.306-310

[16] N.M. Patel, P.G. Harrison, "On Hot-Spot Contention in Multistage Interconnection Networks", *ACM SIGMETRICS*, May 1988, pp. 114-123.

[17] H.G. Perros, T. Altiok, "Approximate Analysis of Open Networks of Queues with Blocking : Tandem Configurations", *IEEE Transaction on Software Engineering*, Vol. SE-12, No. 3, March 1986, pp.450-461

[18] G.F. Pfister, V. A. Norton, "Hot-Spot Contention and Combining in Multistage Interconnection Networks", *IEEE Transaction on Computers*, Vol. c.34, No. 10, October 1985, pp.943-948

[19] T. Szymanski, S. Shaikh, "Markov Chain Analysis of Packet-Switched Banyans with Arbitrary Switch Sizes, Queue Sizes, Link Multiplicities and Speedups", *1989 IEEE INFOCOM*, pp.960-971

[20] T.H. Theimer, E.P. Rathgeb, M.N. Huber, "Performance Analysis of Buffered Banyan Networks", *Performance of Distributed and Parallel Systems*, Kyoto University 1988, pp. 57-72

[21] J.S. Turner, "Design of an Integrated Services Packet Network", *IEEE J. Select. Areas of Communication*, vol. SAC-4, no. 8, Nov. 1986, pp.1373-1380

[22] D.L. Willick, D.L. Eager, "An Analytical Model of Multistage Interconnection Networks", *ACM SIGMETRICS* 1990, pp. 192-199

[23] H. Yoon, K.Y. Lee, M.T. Liu, "Performance Analysis of Multibuffered Packet-Switching Networks in Multiprocessor Systems", *IEEE Transaction on Computers*, Vol. 39, No. 3, March 1990, pp.319-327