# PERFORMANCE ANALYSIS OF BANYAN NETWORKS BASED ON BUFFERS OF VARIOUS SIZES

Arindam Saha and Meghanad D. Wagh

Department of Computer Science and Electrical Engineering
Lehigh University, Bethlehem, PA 18015

## ABSTRACT

Multistage interconnection networks (MINs) and in particular, banyan networks, constitute a viable alternative to the expensive crossbar switches for packet communication. This paper describes an analysis and simulation of banyan networks based on buffers of various sizes connected at the input link of each node. A generalized model to estimate performance indices is developed. It is demonstrated that while buffering produces a considerable improvement in performance, optimum buffer size depends on the average input load. Analytic estimates are substantiated by simulation results.

## I. INTRODUCTION

Performance of large-scale, multiprocessor systems rests primarily on an efficient design of the network interconnecting the processors to the memory modules. The function of this network is to provide a link between any processor and any memory module. Interconnection networks range from the least expensive but extremely slow time-shared bus to the high bandwidth but very expensive crossbar network. As a viable alternative to crossbar switches, multistage interconnection networks (MINs) have assumed widespread interest in recent times [1]-[5]. Among these networks, self-routing ones like the Delta [5] and Omega [3] are popular because of the ease in setting the switches by a destination tag generated by the processor. A specific type of Delta network, known as the banyan network, is being considered in this paper.

Any MIN, unlike a crossbar network, is a blocking network even if no two processors request the same memory module. A conflict arises when two or more processors need the same link between two successive stages in reaching their destinations. Due to this interference, a subset of processors may be blocked, thus degrading the performance. Further, we consider random request patterns such that any number of processors might request the same memory module. It may be noted that the performance of a crossbar is also degraded when the requests are random [6].

In this paper we analyze the performance of banyan networks based on first-in-first-out register queues or data buffers placed at the input links of each switch or node. Results demonstrate considerable improvement in the performance due to the buffers. The network is defined in Section II. Section III discusses the performance of the unbuffered banyan network. A discrete Markov chain model of the banyan network based on single-buffered nodes is developed in Section IV. In Section V we simplify the model, which is then generalized for buffers of arbitrary size in Section VI. We also discuss the behavior of the performance indices with respect to the size of the buffer, size of the network, and the applied load in this section. In Section VII we present some simulation results. Section VIII summarizes the paper and contain some concluding remarks.

## II. DEFINITIONS

A network is a directed graph with three types of nodes: processor nodes with indegree 0, memory nodes with outdegree 0, and switches which have positive indegree and outdegree. Each graph edge represents a link going from a node to a successor. A banyan network is defined [2] to be a network with a unique path from each processor to each memory module which implies that the links leading to a node and from a node form trees. An $n$ stage banyan network has processors attached to stage 0 and memories to stage $n+1$. All outputs from stage $i$ are connected to inputs to stage $i+1$. It is a uniform network in which all switches at the same stage have the same number of input and output ports. A banyan network of degree $k$ is a network built of k X k switches and supports $k^n$ processor/memory pairs. Fig.1 shows a $2^3$ X $2^3$ banyan network.

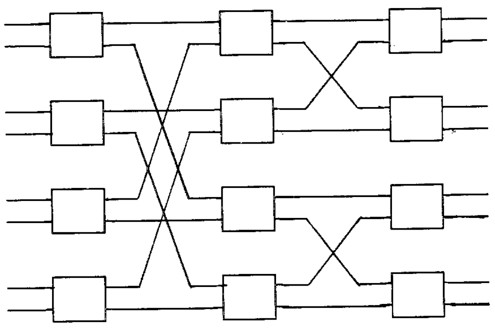

Fig. 1. A $2^3$ by $2^3$ banyan network.

The routing algorithm is very simple. The most significant bit of the binary destination tag selects the upper (0) or lower (1) output of the stage 1 switch towhich the request is applied. The next bit selects the stage 2 switch output, and so on. If two or more requests arrive at one switch and request the same output link then a conflict occurs. Then we assume that one request is randomly selected and forwarded while the others are blocked or discarded. In reality, a blocked request will be resubmitted. But for simplicity we generally assume that a blocked request is ignored. Previous studies [7] have shown that the performance under random request patterns is not substantially affected by the above assumption.

The requests are assumed to be generated by identically distributed random processes. Conflicts are resolved randomly. Thus [8] due to the uniqueness of paths in banyans we have the following:

i) Patterns of packet arrivals at the inputs of the same switch are independent.
ii) All input packets are independently and equiprobably directed to each output link.

We assume that each 2 X 2 crossbar switch or node is capable of passing two packets concurrently if they are directed to different output links. Each input link of a node contains data buffers. Buffers are uniform such that identical buffers are placed at every input link, or else none of the links contain any buffer. These buffers serve as temporary storage for the packets. A packet blocked in some intermediate stage can take refuge in the corresponding buffer until it is unblocked and able to move forward. Also, different nodes on a path can operate on different packets at the same time. This pipelining effect should increase the throughput. However, a full buffer will obstruct packets. Cumulatively, this may cause a 'backlash' such that the packets are unable to enter the network at the very first stage. Buffers will also tend to increase the minimum possible delay experienced by any packet because at heavy loads packets will generally pass through an entire buffer at every stage. So intuitively, buffers have both advantages as well as drawbacks.

## III. UNBUFFERED BANYAN NETWORKS

This analysis is similar to Patel's[5]. The assumptions are as follows:

i) At every cycle every processor issues an independent request with a probability p;
ii) Requests are uniformly distributed among the memories;
iii) Blocked requests are ignored.

In any given cycle, for a single k X k node,

probability(an input port receives a request to a particular output port) = p/k

probability(an input port does not receive a request to a particular output port) = (1 - p/k).

probability(a particular output port is not requested) $= (1 - p/k)^k$

probability(an output port is selected by at least one processor) $= 1 - (1 - p/k)^k$

Now consider a $k^n$ X $k^n$ banyan. Since the outputs from a node at stage $i$ become the inputs to a node at stage $i+1$, we obtain the following recurrence relation:

$$p(m) = 1 - (1 - (1/k)^{*}p(m-1))^k, \qquad (1)$$

where p(m) = probability(there is a packet on any particular input at the m-th stage of the network). The boundary condition of this recurrence relation is p(0) = L, where L is the offered load. For k = 2, we get $p(m) = 1 - (1 - .5p(m-1))^2$. The asymptotic solution to this recurrence can be obtained as in Kruskal and Snir [8]:

$$p(m) = 4/m(1 - \ln(m)/m + c/m). \qquad (2)$$

Unfortunately, for each initial value p(0) we need a different constant $c$. Moreover, for small p(0) the approximation (2) is not good. We present an asymptotic formula which is an excellent approximation for low values of p(0) and $m$.

$$\begin{aligned} p(m) &= p(m-1)[1-.25p(m-1)] \\ &= p(m-2)[1-.25p(m-2)][1-.25p(m-2)+.0625p^2(m-2)] \\ &= p(m-3)[1-.25p(m-3)][1-.25p(m-3)+.0625p^2(m-3)] \\ &\quad [1-.25p(m-3)+.125p^2(m-3)-.03p^3(m-3)+.004p^4(m-3)] \end{aligned}$$

Since p(m) decreases monotonically to zero, one may neglect the third and higher order terms and approximate it as

$$\begin{aligned} p(m) &\approx p(m-i)[1-.25p(m-i)+.0625p^2(m-i)][1-.25p(m-i)]^i \\ &= p(0)[1-.25p(0)+.0625p^2(0)][1-.25p(0)]^m. \qquad (3) \end{aligned}$$

Equation (3) is a fairly good approximation even when p(0) is 0.5 as demonstrated by the plot in Fig. 2.

## IV. A DISCRETE MARKOV CHAIN MODEL

We consider n-stage banyan networks built with 2 X 2 nodes, each node having a buffer of size one at each of its input links. To exactly model such a network with a discrete Markov chain the number of states will be enormous and the problem soon becomes intractable. In order to reduce the number of states the following assumptions [9] are made:

i) Packets are generated by independent, uniform, random processes at the network input ports. Furthermore, the destination addresses are uniformly distributed over the set of the output addresses. Thus a node at any stage is statistically indistinguishable from any other node at the same stage; hence the state of a stage can be modeled by a single node.

ii) The two buffers in the same node are statistically independent because the input addresses are from disjoint sets of input links at the first stage. Thus the state of a

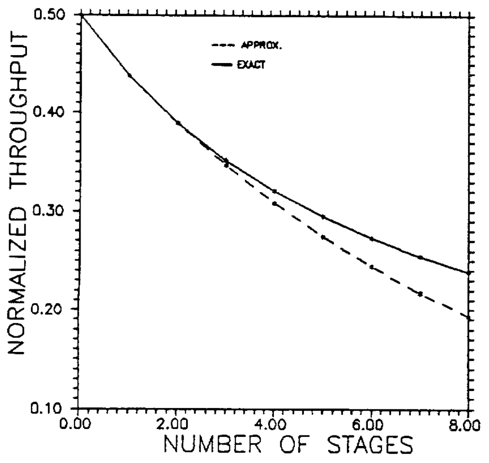

Fig.2. Performance of unbuffered Banyan network when p(0) = 0.5.

node can be modeled by a single buffer; hence the state of a stage is reduced to the state of a single buffer.

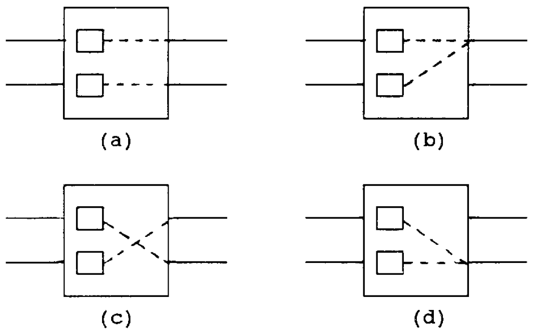

Fig.3. Four possible switch configurations.

At the beginning of each clock cycle a node can always be in one of the four configurations, as shown in Fig. 3. But each of these configurations may not be equiprobable or independent of the previous cycle. A buffer with a packet that was blocked while trying to reach the upper output link will again try to accomplish the same in the next cycle; hence that particular node will definitely be in either of the two configurations shown in Figs. 3(a) and 3(b). So instead of keeping just two state variables for an empty and a full buffer, we introduce two more states to incorporate the effect of blocking. Later we will simplify our model by removing these two variables and find out whether that causes any substantial effect or not. We now describe our model.

In the following notation, *m* denotes the stage number ($1 \le m \le n$, where n is the total number of stages), and *t* stands for time.

p(0,m,t) = probability of a buffer of a stage *m* node being empty at time *t*.

p(1,m,t) = probability of a previously unblocked buffer (of a stage *m* node) being full at time *t*.

$p^u(1,m,t)$ = probability of a buffer (of a stage *m* node), previously blocked while going up, being full at time *t*.

$p^d(1,m,t)$ = probability of a buffer (of a stage *m* node), previously blocked while going down, being full at time *t*.

q(m,t) = probability of a packet being offered to a buffer of a stage *m* node, irrespective of whether it can accept it or not, at time *t*.

f(m,t) = probability of a packet in a buffer of a stage *m* node moving forward at time *t*, given that the buffer has an unblocked packet.

$f^u(m,t)$ = probability that a packet in a buffer of a stage *m* node moves forward at time *t*, given that the buffer has a packet blocked previously going up.

$f^d(m,t)$ = probability that a packet in a buffer of a stage *m* node moves forward at time *t*, given that the buffer has a packet blocked previously going down.

Calculation of q(m,t): While calculating this transition probability we consider, without loss of any generality, the upper buffer of a node at stage *m*. We express q(m,t) as the sum of four factors corresponding to the four cases when the upper buffer of the node at stage *m-1* is full with an unblocked packet, full with a packet blocked going up, full with a packet blocked going down, and empty respectively. For example, if the upper buffer at stage *m-1* is full with an unblocked packet then the probability that a packet is offered to the buffer at stage *m* is derived as

$[.5p(0,m\text{-}1,t)+.75p(1,m\text{-}1,t)+p^u(1,m\text{-}1,t)+.75p^d(1,m\text{-}1,t)]$.

Therefore, q(1,t) = L, the average offered load, and

$$\begin{aligned} q(m,t) = {} & p^d(1,m\text{-}1,t)[p^u(1,m\text{-}1,t)+.5p(1,m\text{-}1,t)] \\ & + p(0,m\text{-}1,t)[.5p(1,m\text{-}1,t)+p^u(1,m\text{-}1,t)] \\ & + p(1,m\text{-}1,t)[.5p(0,m\text{-}1,t)+.75p(1,m\text{-}1,t)+p^u(1,m\text{-}1,t) \\ & +.75p^d(1,m\text{-}1,t)] + p^u(1,m\text{-}1,t) \qquad \text{for } m = 2,3,\ldots,n. \end{aligned}$$

Calculation of f(m,t), $f^u(m,t)$, and $f^d(m,t)$: Each of these probabilities is computed as the product of two factors. The first factor represents the probability that the buffer at stage *m+1* is able to accept the packet. It is equal to $[p(0,m+1,t) + p(1,m+1,t)f(m+1,t) + p^u(1,m+1,t)f^u(m+1,t) + p^d(1,m+1,t)f^d(m+1,t)]$. The second factor corresponds to the probability that the packet can get past the node at stage *m*. For example, if the buffer at stage *m* has an unblocked packet then the second factor is $[p(0,m,t) + .75p(1,m,t) + p^u(1,m,t) + p^d(1,m,t)]$.

Therefore,

$$\begin{aligned} f(m,t) = {} & [p(0,m+1,t)+p(1,m+1,t)f(m+1,t) \\ & +p^u(1,m+1,t)f^u(m+1,t)+p^d(1,m+1,t)f^d(m+1,t)] \\ & [p(0,m,t)+.75p(1,m,t)+p^u(1,m,t)+p^d(1,m,t)]. \end{aligned}$$

$$\begin{aligned} f^u(m,t) = {} & [p(0,m+1,t)+p(1,m+1,t)f(m+1,t) \\ & +p^u(1,m+1,t)f^u(m+1,t)+p^d(1,m+1,t)f^d(m+1,t)] \\ & [p(0,m,t)+.75p(1,m,t)+.5p^u(1,m,t)+p^d(1,m,t)]. \end{aligned}$$

$f^d(m,t) = [p(0,m+1,t)+p(1,m+1,t)f(m+1,t) + p^u(1,m+1,t)f^u(m+1,t)+p^d(1,m+1,t)f^d(m+1,t)]$
$[p(0,m,t)+.75p(1,m,t)+p^u(1,m,t)+.5p^d(1,m,t)]$,
for $m = 1,2,...,n-1$.

$f(n,t) = [p(0,n,t)+.75p(1,n,t)+p^u(1,n,t)+p^d(1,n,t)]$.
$f^u(n,t) = [p(0,n,t)+.75p(1,n,t)+.5p^u(1,n,t)+p^d(1,n,t)]$.
$f^d(n,t) = [p(0,n,t)+.75p(1,n,t)+p^u(1,n,t)+.5p^d(1,n,t)]$.

Updating the state variables: The probability of any state variable at time *t+1* can be expressed as the sum of two factors (maybe zero), corresponding to the two cases of a packet being offered or not at time *t*. We derive the following:

$$p(0,m,t+1) = (1-q(m,t))[p(0,m,t) + p(1,m,t)f(m,t) + p^u(1,m,t)f^u(m,t) + p^d(1,m,t)f^d(m,t)].$$

$$p^u(1,m,t+1) = (1-f^u(m,t))p^u(1,m,t) + .5(1-f(m,t))p(1,m,t)$$

$$p^d(1,m,t+1) = (1-f^d(m,t))p^d(1,m,t) + .5(1-f(m,t))p(1,m,t)$$

$$p(1,m,t+1) = q(m,t)[p(0,m,t) + p(1,m,t)f(m,t) + p^u(1,m,t)f^u(m,t) + p^d(1,m,t)f^d(m,t)].$$

We can verify that the updated state variables add up to unity. At steady state all quantities will converge to time-independent ones and so we can drop the time subscripts while calculating the network performance indices like the normalized throughput and the normalized delay. The normalized throughput, TP, is defined as the number of packets per output link per unit time. Note that at steady state this should be the same at all stages.

$TP=p(1,m)f(m)+p^u(1,m)f^u(m)+p^d(1,m)f^d(m)$ for a valid $m$.

The delay per stage can be related to the probability that a packet moves forward. We know that the probability that a request is serviced in $x$ cycles is $g(m)(1 - g(m))^{x-1}$ where $g(m)$ is the probability that the packet moves forward at stage $m$. Thus the delay per stage, or the expected number of cycles required per stage, is

$$\sum_{i=1}^{\infty} g(m)(1-g(m))^{i-1} \quad \text{which equals } 1/g(m).$$

In our case,

$$g(m) = \frac{p(1,m)f(m) + p^u(1,m)f^u(m) + p^d(1,m)f^d(m)}{p(1,m) + p^u(1,m) + p^d(1,m)}.$$

So the normalized delay, D, for the whole network is

$$D = (1/n) \sum_{i=1}^{n} (1/g(i)) .$$

The above equations are solved iteratively. Fig. 4 shows TP plotted against the average input load for various network sizes. The results are similar to those reported by Jenq[9]. We find that for low loads, typically less than 0.4, the throughput increases linearly and is almost equal to the load. But as the load increases the rate of increase in the throughput is reduced. This reduction, as expected, is more for larger networks. At full load, the throughput drops from 0.633 to 0.453 as we increase the number of stages from 2 to 10. The throughput seems to converge to a value of 0.45 approximately. So we can conclude that with a single buffer the maximum throughput for a large sized network (10 or more stages) is about 0.45. Fig. 5 shows the normalized delay plotted versus the load for different

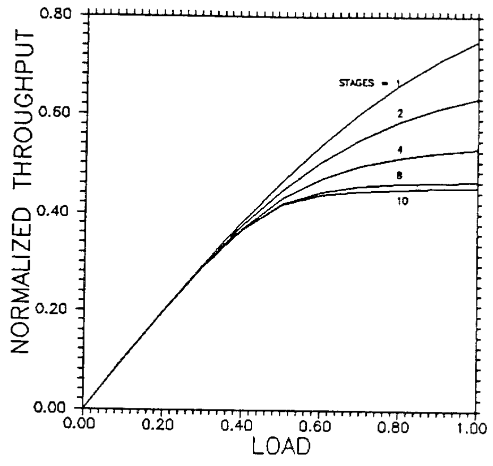

Fig. 4. Throughput of an 1-buffer Banyan network.

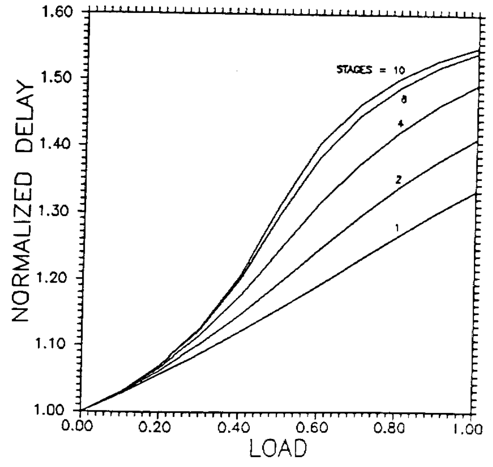

Fig. 5. Delay of an 1-buffer Banyan network.

network sizes. The lowest delay is obviously 1.0, which translates to an average delay of $nt_c$ where $t_c$ is the clock period. Like the throughput, the delay also converges to a constant, about 1.55, as the network becomes very large. Hence we conclude that large banyan networks (10 or more stages) with single buffers will have at full load, about 0.45 packets per output link per unit time, and each packet will be delayed by $1.55nt_c$ time units.

## V. A SIMPLER MODEL

We now simplify our analytic model by getting rid of the two state variables corresponding to the blocked cases. This is substantiated by the fact the results obtained by this simpler model are almost identical to those reported in the previous section (less than 0.05% differences). The variables of this model are similar to those used by Jenq[9]. We present an alternative derivation for all the variables. The model is described below:

$p(0,m,t)$ = probability of a buffer of a stage $m$ node being empty at time $t$.
$p(1,m,t)$ = probability of a buffer of a stage $m$ node being full at time $t$.
$q(m,t)$ = probability of a packet being offered to a buffer of a stage $m$ node at time $t$.
$f(m,t)$ = probability of a packet in a buffer of a stage $m$ node moving forward at time $t$, given that the buffer has a packet.

Using the concepts outlined in the previous section, we derive the following:

$q(1,t) = L$ and

$q(m,t) = p(1,m-1,t)[1-.25p(1,m-1,t)]$ for $m = 2,3,\ldots,n$.

$f(m,t) = [p(0,m+1,t)+p(1,m+1,t)f(m+1,t)][1-.25p(1,m,t)]$, for $m = 1,2,\ldots,n-1$.

$f(n,t) = 1 - .25p(1,n,t)$.

$p(1,m,t+1) = (1 - q(m,t))p(1,m,t)(1 - f(m,t)) + q(m,t)$.

$p(0,m,t+1) = (1 - q(m,t))[p(0,m,t) + p(1,m,t)f(m,t)]$.

$TP = p(1,m)f(m)$ for any valid $m$.

$$D = (1/n)\sum_{i=1}^{n} (1/f(i)).$$

## VI. MODEL WITH MULTIPLE BUFFERS

The model in the previous section was restricted to a buffer of size one only. We now generalize it for multistage banyan networks built with 2 X 2 nodes, with each input link of a node having multiple buffers. Let b and n denote the size of each buffer and the number of stages respectively, and the statistics be defined as

$p(i,m,t)$ = probability of a buffer of a stage $m$ node having $i$ packets at time $t$ $(0 \le i \le b)$.

$q(m,t)$ = probability of a packet being offered to a buffer of a stage $m$ node at time $t$.

$f(m,t)$ = probability of a packet in a buffer of a stage $m$ node moving forward at time $t$, given that the buffer has *at least one* packet.

The transition probabilities, the updated state variables, and the performance indices are derived byusing similar principles as described in Section IV. They are as follows:

$q(1,t) = L$, and

$q(m,t) = (1-p(0,m-1,t))[.75+.25p(0,m-1,t)]$ for $m = 2,3,\ldots,n$.

$f(m,t) = [1-p(b,m+1,t)+p(b,m+1,t)f(m+1,t)]$ $[.5+.5p(0,m,t)+.25(1-p(0,m,t))]$ for $m = 1,2,\ldots,n-1$.

$f(n,t) = .5 + .5p(0,n,t) + .25(1 - p(0,n,t))$.

$p(0,m,t+1) = (1-q(m,t))[p(0,m,t)+p(1,m,t)f(m,t)]$.

$p(1,m,t+1) = (1-q(m,t))[p(1,m,t)(1-f(m,t)) + p(2,m,t)f(m,t)] + q(m,t)[p(1,m,t)f(m,t) + p(0,m,t)]$.

$p(i,m,t+1) = (1-q(m,t))[p(i,m,t)(1-f(m,t)) + p(i+1,m,t)f(m,t)] + q(m,t)[p(i-1,m,t)(1-f(m,t)) + p(i,m,t)f(m,t)]$, for $2 \le i \le b-1$.

$p(b,m,t+1) = (1-q(m,t))p(b,m,t)(1-f(m,t)) + q(m,t)[p(b,m,t) + p(b-1,m,t)(1-f(m,t))]$.

$TP = (1 - p(0,m))f(m)$ for any valid $m$.

$$D = (1/n)\sum_{i=1}^{n} (1/g(i)),$$

$$\text{where } g(i) = (f(i)/(1 - p(0,i))\sum_{j=1}^{b} (1/j)p(j,i).$$

The above equations have been solved iteratively. Normalized throughput and normalized delay are computed with respect to three parameters -- size of the buffer, size of the network, and the average input load. Figs. 6 and 7 show the throughput and the delay versus the number of stages for various sizes of the buffer, with the average load fixed at 1.0. We find that for lower sized buffers, the throughput decreases considerably as the number of stages is increased. For example, for networks with buffers of size 2, the throughput reduces from 0.75 to 0.6 as the size of the network increases from 1 to 10 stages. But with larger sized buffers the rate of decrease of throughput is much less. For networks with buffers of size 7, the throughputs are 0.75 and 0.71 when the number of stages are 1 and 10 respectively. Fig. 7 shows that the normalized delay decreases with the increase in the number of stages when the buffer size is more than one. We must note that the total average delay is the

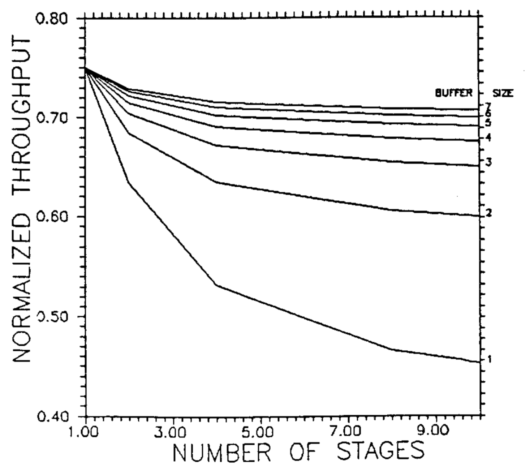

Fig. 6. Throughput dependence on number of stages with varying buffer sizes at full load.

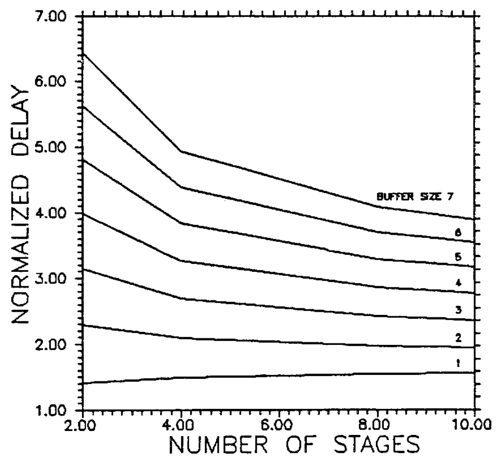

Fig. 7. Delay dependence on number of stages with varying buffer sizes at full load.

product of the normalized delay and the number of stages; obviously this quantity increases steadily as the network becomes larger.

In Fig. 8 we plot the normalized throughput against the size of the buffer for different network sizes, at full load. Similar results, obtained by simulation only, have been reported elsewhere [10]. In our case, the throughput is remarkably improved as we increase the size of the buffer, especially when the network size is large. An 8-stage network shows an improvement in throughput of

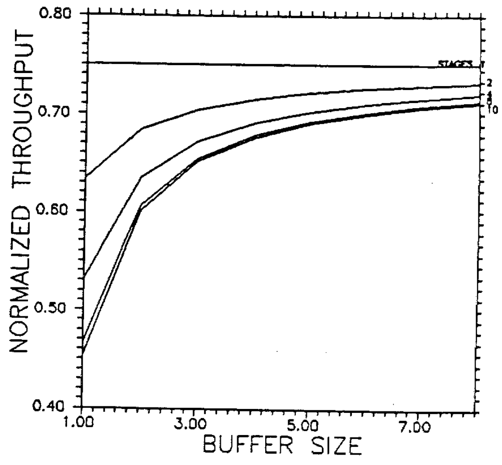

Fig. 8. Throughput dependence on buffer size at full load.

about 0.2463 as the buffer size is increased from 1 to 8. But interestingly, throughput of large networks (8 or more stages) seems to converge to a value of about 0.71 when the size of the buffer is 5 or more. So we find that for large networks, buffers of size greater than 5 do not result in any substantial gain in the throughput. Whereas the throughput converges to a constant value, Fig. 9 shows that the normalized delay steadily increases as the size of the buffer is increased. Unlike the throughput, this delay does not converge to any maximum constant value. Hence at full load buffers of size greater than 5 will introduce considerable delay without really increasing the throughput much.

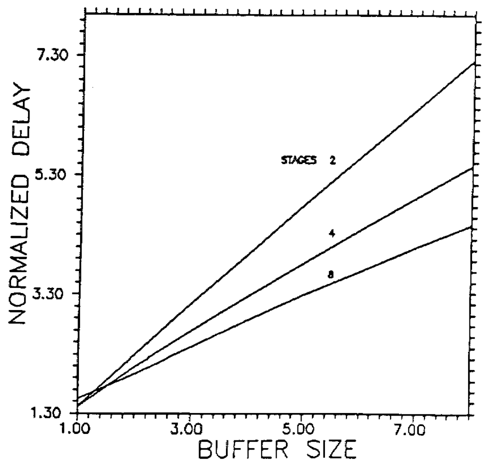

Fig. 9. Delay dependence on buffer size at full load.

The optimal size of the buffer is also dictated by the input traffic density or the average input load. In Figs. 10 and 11 we have plotted the throughput and the delay respectively versus the size of the buffer for an 4-stage network at different values of the input load. We find that at lower loads, the throughput saturates quickly at buffer sizes lower than those at full load. But the delay increases steadily at lower loads also. Hence if the input load is low (0.6 or less) then it is advisable to restrict the buffer to sizes 2 or 3 for optimal performance.

## VII. SIMULATION RESULTS

In order to substantiate our analytic estimates we have developed a simulator for a buffered banyan network in a packet communication environment. The two performance indices used are the normalized throughput and the normalized delay. After allowing some settling time, point estimates are obtained over a huge number of packets passing through the network. Most of our analytic estimates were closely matched by the simulation results. Some results are shown in Figs. 12 and 13. Fig. 12 shows

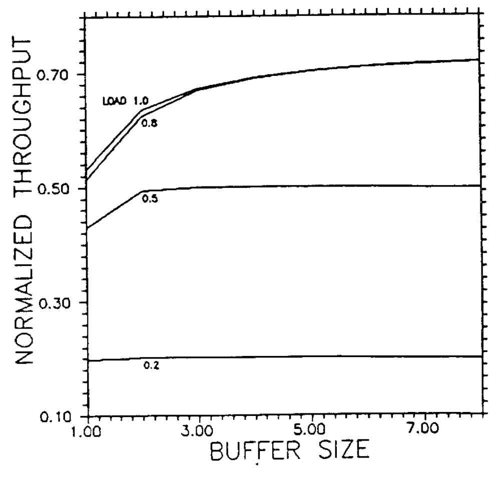

Fig. 10. Throughput dependence on buffer size for a 4-stage Banyan network.

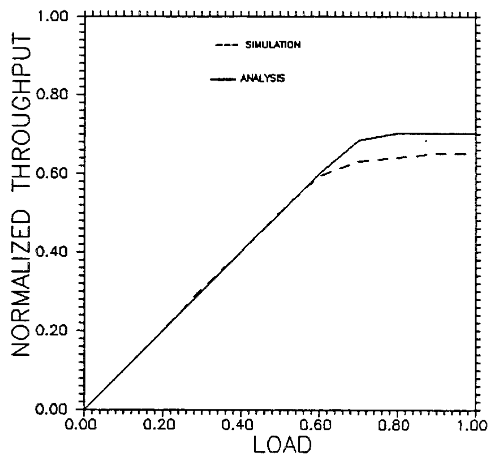

Fig. 12. Throughput of a 4-stage, 5-buffer Banyan network.

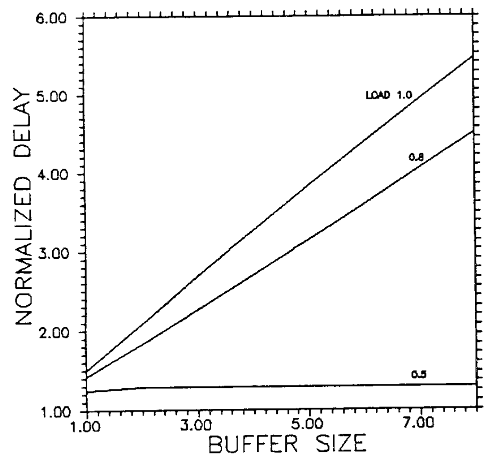

Fig. 11. Delay dependence on buffer size for a 4-stage Banyan network.

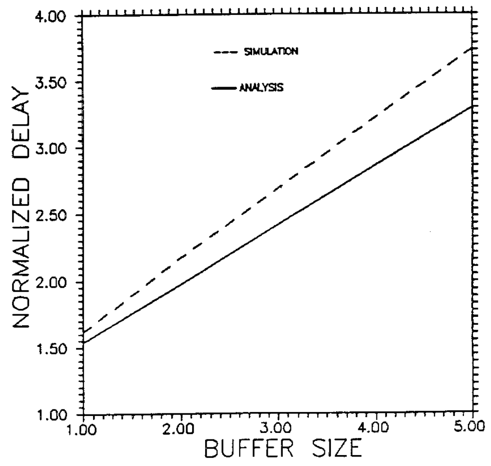

Fig. 13. Delay of an 8-stage Banyan network at full load.

the variation of the normalized throughput with the average load for a 4-stage banyan network based on 5 buffers at each input link. At low loads the simulation exactly matches the analysis. There is a small discrepancy at very high loads. Fig. 13 shows the variation of normalized delay with the change in buffer size for an 8-stage network at full load. Here we find that our analysis is consistently smaller than the simulation results, with an average discrepancy of about 6 percent. Besides statistical errors, we think that the discrepancies are due to the assumption of statistical independence of the events occurring at the input links in each stage.

## VIII. CONCLUSIONS

We have analytically estimated the performance of a type of multistage interconnection network, namely the banyan network based on FIFO register queues or buffers of various sizes at the input links of each node. Wederived an analytic model for computing the normalized throughput and the normalized delay of such networks. Buffered networks outperform unbuffered ones. With buffers of size one only, both the throughput and the delay converge to some maximum values as the networks become larger and larger. As the size of the buffers is increased the throughput converges to a constant value but the delay increases steadily. At higher input loads, the rate of increase if throughput is very small when the size of the buffers is 5 or more. At lower loads, we find that the size of the buffers should be restricted to 2 or 3. We hope that the findings of this research will help the system architects and network designers. Further work in this area could possibly include non-uniform input traffic analysis and reliability of these networks.

## REFERENCES

[1] C. Wu and T. Feng, "On a class of Multistage Interconnection Networks," *IEEE Trans. Computers*, Vol. C-29, No. 8, pp 694-702, Aug. 1980.

[2] L. R. Goke and G.J. Lipovski, "Banyan networks for partitioning Multiprocessing systems," Proc. First Annual Computer Architecture Conference, pp 21-28, Dec. 1973.

[3] D. H. Lawrie, "Access and Alignment of Data in an Array Processor," *IEEE Trans. Computers*, Vol. C-24, No.12, pp 1145-1155, Dec. 1975.

[4] K. E. Batcher, "The Flip Network in STARAN," Proc. 1976 International Conference on Parallel Processing, pp 65-71, Aug. 1976.

[5] J. H. Patel, "Processor-Memory Interconnections for Multiprocessors," Proc. Sixth Annual Symp. Computer Architecture, pp 168-177, Apr. 1979.

[6] D. P. Bhandarkar, "Analysis of Memory Interference in Multiprocessors," *IEEE Trans. Computers*, Vol. C-24, No. 9, pp 897-908, Sept. 1975.

[7] C. V. Ravi, "On the bandwidth and interference in interleaved memory systems," *IEEE Trans. Computers*, Vol. C-21, No. 8, pp 899-901, Aug. 1972.

[8] C. P. Kruskal and M. Snir, "The performance of Multistage Interconnection Networks for Multiprocessors," *IEEE Trans. Computers*, Vol. C-32, pp 1091-1098, Dec. 1983.

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

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