Delay Analysis of a Circuit-Switched Interconnection Network with Non-Uniform Traffic

Paul Dietrich
Dept. of Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA 92093

Ramesh R. Rao *
Dept. of Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA 92093

Abstract

In this paper, we analyze a circuit-switched, blocking, multi-stage interconnection network (MIN) with arbitrary independent input distributions and arbitrary switch routing probabilities. The network uses a gated-hold strategy which retains partial path information. By formulating a dominant system, we derive a recursive expression for the mean time to process a batch of requests given a particular input distribution and a set of switch routing probabilities. This system is compared to a simulation of a non-blocking switch implementing a similar gated strategy. Results indicate that this method yields tight bounds for small networks with arbitrary input distributions and switch routing probabilities. For networks with uniform input and output distributions this bound is closer than one introduced in a previous work examining the same protocol.

1 Introduction

In parallel processing systems, connection networks are needed to allow communication among processors and between processors and memories. MINs are used in processor-memory connection when full crossbar switches are too costly or impractical to construct. A MIN offers full connectivity and reduced complexity at the expense of potential internal blocking and added delay. Analysis of MIN performance to quantify the effects of delay and blocking is essential in understanding associated issues and tradeoff involved in parallel system design. Analyses that rely on drastically simplifying system assumptions may not be useful in practical design and implementation issues. A practical MIN model should capture the effects of network topology, blocked requests, and non-uniform traffic on system performance.

*Supported by NSF under grant NCR-8904029 and the University of California San Diego, Academic Senate Committee on Research.

Research has shown [1], [2], [6], that blocking networks provide a better cost/performance ratio than crossbar networks. In his pioneering analysis, Patel [1] ignored the resubmission of blocked requests and derived an expression for the bandwidth of a Banyan network. Kruskal and Snir's asymptotic call blocking analysis [2] also assumed that blocked requests are not resubmitted. In blocking networks, a fraction of the messages are blocked due to contention within the switching fabric and must be resubmitted to guarantee reliable communications. Because requests may have to be submitted several times, a request experiences a random delay depending on the current state of the MIN. Therefore, analyses that ignores blocked requests cannot answer questions concerning circuit set-up delay.

Incorporating the resubmission of blocked requests in circuit-switched network model has a pronounced effect on predicted performance [7]. Bhattacharya, Rao, and Lin [9] derived an upper bound on message delay for a synchronous circuit-switched Delta network that accommodates the resubmission of blocked requests. To preserve uniformity and independence of the message input distribution they assumed that blocked requests are resubmitted after a random number of cycles. This model also assumes that the resubmitted requests re-randomize their destinations. If several devices require connection through the same switch in the MIN, the re-randomization of the blocked requests may yield performance results that are optimistic as the chance of repeated collisions is lowered.

On a different note, in these analyses, the arriving requests are assumed to be uniformly distributed among all inputs, and their destinations are assumed to be uniformly distributed among all outputs. Based on these assumptions, Kruskal and Snir [2] have shown that a request entering a switch is equally likely to require connection to the top and bottom output. Message routing in such a network can be modeled statistically as independent 'fair coin' tosses. Because these
analyses necessitate the assumption of uniform output distributions, skewed output distribution or 'hot spot' analysis cannot be accommodated.

Several authors have examined non-uniform output distributions in their analysis of circuit-switched MIN's. In Pombortsis and Halatsis' analysis [3], the effect of non-uniform traffic on a circuit-switched MIN was shown to be less significant that its effect on packet-switched networks (the effect of 'tree saturation' caused by 'hot spots' on buffered packet switched networks is severe for even small non-uniformities in output distributions). However, in [3], it is assumed that blocked requests are dropped so that request destinations are independent from cycle to cycle. This eliminates the effect of blocked request repeatedly attempting connection to the same output port. In circumstances where blocked requests must be resubmitted before the processor can generate a new request (i.e. a synchronization process), the effect of 'hot spots' may be significant. In fact, in [5], it was shown via simulation that the overflow traffic (blocked requests) from a MIN with 'hot' traffic is relatively much hotter than the traffic incident on the MIN.

Recently, Dietrich and Rao [10] analyzed a synchronous, circuit-switched square Banyan network of $2 \times 2$ crossbar switches implementing a gated-hold protocol. In this paper we extend this analysis, incorporating non-uniform, independent input distributions and arbitrary switch routing probabilities so that the 'fair coin' switching assumption can be removed. By selecting these switching probabilities, the effect of 'hot spots' can be incorporated into the analysis. It must be noted that by picking these probabilities, one cannot engineer arbitrary output distribution for each input device. However, certain choices of these probabilities result in individual output distribution that more closely represent non-uniform connection demand in a MIN. The classical 'hot spot' model in which, for each device, one output port is chosen with higher probability than the other can be accurately modeled via these switching probabilities.

2 Proposed Protocol

2.1 Protocol description

We consider a synchronous circuit-switched network with a holding strategy as in [10]. Time is divided into periods called network cycles. At the beginning of each network cycle, the input devices submit their request for connection to the output devices. As these request propagate through the MIN, some are blocked and some survive. The requests that are blocked hold their partial paths. When the requests that did not get blocked finish their jobs and release their connections, the blocked jobs continue, starting with those in the stage of the MIN closest to the outputs of the network. This process continues until all requests have been served in which case a new period begins and the processors submit new requests that may have arrived. Unlike the synchronous circuit-switched protocol with request resubmission or dropping, these service periods are not of equal length, and depend upon the number of collisions that occur during contention for the communications paths.

For example, consider the set of active users represented by dots in Fig 1a. After one switching cycle, no collisions occur and all requests advance to the next stage (Fig. 1b). During the second switching cycle, some of the requests require connection to the same output ports and only four of them progress (Fig. 1c). Of those that progress, one is blocked in the next switching cycle and the remaining requests transmit their messages and release their circuits requiring $d$ switching cycles (Fig. 1d). In the first switching cycle following this, the blocked request in the last stage advances and transmits its message (Fig. 1e). In the switching cycle following this, both requests in the second stage advance (Fig. 1f). Finally, in the last switching cycle, these two requests set up a circuit and transmit their messages (not shown). At this point a new network cycle begins with a new input distribution.

This synchronous holding protocol retains the infor-
mation gained in collisions about destination locations of packets. It would seem that a protocol that uses such information might yield better performance than one which discards information concerning collisions.

2.2 Assumptions

We chose an $N$-input, self-routing, square Banyan network with $2 \times 2$ crossbar switches for the interconnection network. This network consists of $k = \log_2 N$ stages of $N/2$ switches. We assume that at the beginning of each network cycle, requests are present at the two inputs to switch $i$ independently with probability $p_1$ and $p_2$, respectively, for all $i : 1 \leq i \leq N/2$. This allows us to infer different demands on the interconnection network. We do not specify the output distribution. Rather, we assume that at each switch $i$ a message is routed to the top output with a certain probability and to the bottom output with the remaining probability, independent of other switches and of the request's source. By varying these probabilities, a certain class of output distributions can be generated. The time required for a request to propagate through one switch is called a switching cycle and is taken to be $1$. In addition we make the assumption that the network cycle time required to service a network with no requests at its input is $0$. Thus no time is spent in a cycle if the cycle has no initial requests.

3 Analysis

3.1 Dominant system construction

Consider a self-routing square Banyan network constructed of $2 \times 2$ crossbar switches with $N$ inputs and $k$ stages of switches where $N = 2^k$. At the beginning of each network cycle, each input device submits a request independently with probabilities given by:

$$P = \{P_{11}, P_{12}, P_{21}, P_{22}, \ldots, P_{N1}, P_{N2}\}^T. \quad (1)$$

No information is assumed about the destinations of these requests. It is assumed however that when a request arrives at switch $s_{ij}$, it is routed to the top output with probability $\alpha_{ij}$ and to the bottom output with probability $1 - \alpha_{ij}$ where $(i,j)$ represents the position of a switch within the network. It follows from the topology of the network, that requests arriving to a given switch are independent. The set of all the requests arriving at a given stage, however, are not independent.

The exact analysis of such a system can be accomplished using a Markov chain description of the entire MIN. However, the number of states grows exponentially with the number of inputs making the analysis intractable for large networks. Instead, we compute a bound on performance of such a system by defining a worse case, dominating system. The dominating system is easier to analyze and we are able to derive a recursive equation that expresses the expected cycle time of a set of requests at the beginning of $k$ stages of an $N$-input network as some function of the expected cycle time of a set of requests at the beginning of $k - 1$ remaining stages of this $N$-input network.

Consider a dominating system similar to that in [10] in which 'dummy' requests are added at the output of the switches to make the set of advancing requests independent across the entire stage of switches. In addition, 'dummy' requests are added to the residual requests so that after passing through the first stage of switches, they form a set of independent requests. The switches in the dominating system of [10] added to the advancing and residual requests so that the request distribution across the output of a stage was independent and uniform. However, in the dominating system constructed here, we no longer require that the requests be uniform, but only independent.

Such an addition always exists. For example, if, as a set of request propagate through a stage in the network, enough requests can be added so that there is a request at every output of that stage, these requests will be trivially independent; the marginal probability of a request being present at each output of each switch would be $1$. However, a dominating system constructed with the above addition strategy shares little of the characteristics of the actual system, as adding too many 'dummy' requests will hide the actual system behavior. Therefore, we need to find the addition which increases the switching load by as little as possible and simultaneously forces independence at the switch outputs.

Consider the following modification of the joint probabilities of requests at the output of a single switch with switching probability $\alpha$. Define $\{X_j\}_{j=1,2}$ as indicator random variables each taking the value $1$ when a request is present at the $j^{th}$ switch input. We assume that these random variables take the value $1$ with probability $p_j$ independent of each other. $\{Y_j\}_{j=1,2}$ are random variables that each take the value $1$ if a re-
quest is present at the corresponding output after one switching cycle. Without the addition of any requests, the joint probabilities of $Y_1$ and $Y_2$ are given by

$$
Pr[Y_1 = y_1, Y_2 = y_2] = \begin{cases} 
(1 - \alpha)(1 - \beta)(p_1 + p_2 - p_1p_2(\alpha + 1)) & y_1 = 0, y_2 = 1 \\
\alpha(p_1p_2(\alpha - 2) + p_1 + p_2) & y_1 = 1, y_2 = 0 \\
(1 - p_1)(1 - p_2) & y_1 = 0, y_2 = 0 \\
2p_1p_2(1 - \alpha) & y_1 = 1, y_2 = 1 
\end{cases}
$$

Consider a switch in the dominating system, that upon seeing a $[1,0]$ or $[0,1]$ at its output, adds a 'dummy' request at its other output with probability $\gamma$ and $\beta$ respectively. The joint output probabilities at such a switch are:

$$
Pr[Y_1 = y_1, Y_2 = y_2] = \begin{cases} 
(1 - \alpha)(1 - \beta)(p_1 + p_2 - p_1p_2(\alpha + 1)) & y_1 = 0, y_2 = 1 \\
\alpha(1 - \gamma)(p_1p_2(\alpha - 2) + p_1 + p_2) & y_1 = 1, y_2 = 0 \\
(1 - p_1)(1 - p_2) & y_1 = 0, y_2 = 0 \\
2p_1p_2(1 - \alpha) & y_1 = 1, y_2 = 1 
\end{cases}
$$

where $\beta$ and $\gamma$ reflect the addition of requests made in this dominating system. We wish to find the $\beta$ and $\gamma$ that minimizes the average number of requests at the switch output while still forcing $Y_1$ and $Y_2$ to be independent. Analysis of these equations has shown that the $\beta$, $\gamma$ that minimize the expected number at the output and guarantee independence are for $\alpha < .5$, $\beta = 0$ and $\gamma = (1 - \alpha)(p_1 - p_1p_2 + p_2 - p_1p_2(\alpha + 1))$, and are for $\alpha \geq .5$, $\gamma = 0$ and

$$
\beta = \frac{\alpha(1 - p_1)(1 - p_2)(p_1 + p_2 - p_1p_2(\alpha + 1))}{(1 - p_1)(1 - p_2)(p_1 + p_2 - p_1p_2(\alpha + 1))}
$$

For $\alpha = .5$, either solution minimizes the expected number of requests at the outputs. By using these request addition constraints the distribution at the output of a switch can be made independent if the input distribution is independent. The new output distribution is given by:

$$
Pr[Y_1 = 1] = \alpha(p_1 + p_2 - \alpha p_1p_2) \\
Pr[Y_2 = 1] = 1 - \frac{(1 - p_1)(1 - p_2)}{(1 - p_1)(1 - p_2)}
$$

for $\alpha \leq .5$ and

$$
Pr[Y_1 = 1] = 1 - \frac{(1 - p_1)(1 - p_2)}{(1 - p_1)(1 - p_2)(1 - \alpha p_1p_2)} \\
Pr[Y_2 = 1] = (1 - \alpha)(p_1 + p_2 - (1 - \alpha)p_1p_2)
$$

for $\alpha \geq .5$.

The residual requests are also dependent after one switching cycle. We add to these requests as they pass through the switch after the forwarded requests have completed their service. The joint probabilities of the residual requests at the output of the switch is given by:

$$
Pr[R_1 = r_1, R_2 = r_2] = \begin{cases} 
(1 - \alpha)(1 - 2a + 2a^2)(1 - \zeta) & r_1 = 0, r_2 = 1 \\
\alpha(1 - 2a + 2a^2)(1 - \eta)p_1p_2 & r_1 = 1, r_2 = 0 \\
1 - p_1p_2(1 - 2a + 2a^2) & r_1 = 0, r_2 = 0 \\
(1 - 2a + 2a^2) & r_1 = 1, r_2 = 1
\end{cases}
$$

where $\zeta$ and $\eta$ represent the addition of 'dummy' requests. In the same manner as for the forwarded requests, the $\zeta$ and $\eta$ that minimize the expected number at the output of the switch and simultaneously guarantee independence are

$$
\zeta = \frac{p_1p_2(1 - \alpha)(a^2 + (1 - \alpha)^2)}{1 - p_1p_2(a^2 + (1 - \alpha)^2)}
$$

for $\alpha < .5$ and

$$
\eta = 0, \quad \zeta = \frac{p_1p_2(a^2 + (1 - \alpha)^2)}{1 - p_1p_2(a^2 + (1 - \alpha)^2)}
$$

for $\alpha \geq .5$. For $\alpha = .5$ either solution produces a minimum average output number. The output distribution of the switch due to these residual requests is given by:

$$
Pr[Y_1 = 1] = p_1p_2(1 - \alpha)(a^2 + (1 - \alpha)^2) \\
Pr[Y_2 = 1] = \frac{p_1p_2(a^2 + (1 - \alpha)^2)}{1 - p_1p_2(a^2 + (1 - \alpha)^2)}
$$

for $\alpha \leq .5$ and

$$
Pr[Y_1 = 1] = \frac{p_1p_2(a^2 + (1 - \alpha)^2)}{1 - p_1p_2(a^2 + (1 - \alpha)^2)} \\
Pr[Y_2 = 1] = p_1p_2(1 - \alpha)(a^2 + (1 - \alpha)^2)
$$

for $\alpha \geq .5$. If each switch in the first stage makes such modifications, then forwarded and residual requests that reach the second stage will be independent across the entire stage. Thus the second stage of switches see an independent distribution and similar additions to its outputs will ensure independence of the following stage and so on. In this way, it is possible for all switches to make additions so that the output requests will be independent at every stage.

### 3.2 Analysis of the dominant system

Consider an $N$-input network of $2 \times 2$ crossbar switches with network switching probabilities $\alpha$. The stages are numbered in descending order with the initial stage of the network denoted as the $k^{th}$ stage or $(\log_2 N)^{th}$ stage and the final stage of the network de-
noted as the 1st stage. The request at the beginning of the kth stage are independent and have marginal probabilities given by p_{11} and p_{12} at switch i where \( 1 \leq i \leq N/2 \). The matrix of switching probabilities \( \alpha \) is given by

\[
\alpha = \begin{pmatrix}
\alpha_{1,k} & \alpha_{1,k-1} & \ldots & \alpha_{1,1} \\
\alpha_{2,k} & \alpha_{2,k-1} & \ldots & \alpha_{2,1} \\
\vdots & \vdots & \ddots & \vdots \\
\alpha_{q,k} & \alpha_{q,k-1} & \ldots & \alpha_{q,1}
\end{pmatrix}
\]

and \( p \) is the vector shown in (1). Define a random variable \( f^N(p, \alpha, j) \) to be the number of switching cycles required to set up circuits for a set of requests that have progressed to the output of the \( j \)th stage of an \( N \)-input network. The circuit set-up time excludes the time required to transmit the messages. These transmission times will be included in a later section of the analysis.

The expected circuit set-up time or \( E[f^N(p, \alpha, k)] \) can be computed by conditioning on \( A_k \), defined to be the event that there is at least one request at the input to the \( k \) stage network. Further conditioning on \( B_k \), the event that there was contention as a result of requests advancing through the \( k \)th stage yields:

\[
E[f^N(p, \alpha, k)] = E[f^N(p, \alpha, k) | A_k] \cdot Pr[A_k] = (E[f^N(p, \alpha, k) | A_k, B_k] \cdot Pr[B_k | A_k] + E[f^N(p, \alpha, k) | A_k, B_k] \cdot Pr[B_k | A_k]) \cdot Pr[A_k]
\]  \hspace{1cm} (2)

Several properties of the dominating system are crucial to the derivation of a recursive solution. We outline the key arguments here. The event \( A_k \) implies that there will be at least one request that advances through this stage as it is impossible for all request to be blocked. In addition, because, in our construction of the dominating system we did not add to the output requests if no requests were present at the input, \( A_{k-1} \), the event that there is at least one request at the input of the next stage implies that there was a request at the input to the previous stage. Thus \( A_k = A_{k-1} \). From this, we have:

\[
E[f^N(p, \alpha, k) | A_k, B_k] = E[f^N(p, \alpha, k) | A_k, A_{k-1}, B_k]
\]

Since, in this term of (2), we have conditioned on \( B_k \), these requests will pass this stage in one switching cycle and, because of the addition of requests, will advance to the next stage with a new independent distribution given by \( S_{k-1} \cdot F(p, \alpha) \) where the matrix \( S_{k-1} \) describes the connection between the \( k \)th and \( k-1 \)th set of switches and \( F(p, \alpha) \) describes the output distribution of the previous stage with input distribution given by \( p \). Thus

\[
E[f^N(p, \alpha, k) | A_k, B_k] = 1 + E[f^N(S_{k-1} \cdot F(p, \alpha), \alpha, k-1) | A_{k-1}, B_k]
\]

Because we add requests in the dominating system in a manner described in subsection 3.1, given \( A_k \), future evolution of the protocol is not dependent on the events \( B_k \) or \( B_k^c \). Thus we can remove the \( B_k^c \) in the above expression. In a similar fashion the first term of (2) can be rewritten. Plugging in expressions for the remaining probabilities into (2), one can rigorously derive the recursive equation

\[
E[f^N(p, \alpha, k)] = E[f^N(S_{k-1} \cdot F(p, \alpha), \alpha, k-1)] + E[f^N(S_{k-1} \cdot R(p, \alpha), \alpha, k-1)] + 2 - \prod_{i=1}^{N/2} (1 - p_{1i})(1 - p_{2i}) - \prod_{i=1}^{N/2} (1 - p_{1i} p_{2i} (\alpha_{1,i}^2 + (1 - \alpha_{1,i})^2))
\]

\( S_{k-1} \) is the matrix that defines the connections between the \( k \)th and \( (k-1) \)th stages of switches. \( F(p, \alpha) \) and \( R(p, \alpha) \) are the functions that determine the new independent distributions of the forwarded and residual requests.

To finish the recursive equation, we must compute \( E[f^N(p, \alpha, 1)] \). This is computed in a similar fashion as in [10] and found to be equal to

\[
E[f^N(p, \alpha, 1)] = 2 - \prod_{i=1}^{N/2} (1 - p_{1i})(1 - p_{2i}) - \prod_{i=1}^{N/2} (1 - p_{1i} p_{2i} (\alpha_{1,i}^2 + (1 - \alpha_{1,i})^2))
\]

With careful thought one could have written relationship (3) down directly. The switches at the input effectively sort the incoming requests into two groups so that no two request in a given group are destined to the same output port of the same switch. The number of switching cycles required to pass these two groups to the next stage is given by the last three terms of (3). These two groups must then pass through the remaining \( k-1 \) stages which requires a total of \( E[f^N(S_{k-1} \cdot F(p, \alpha), \alpha, k-1)] + E[f^N(S_{k-1} \cdot R(p, \alpha), \alpha, k-1)] \) switching cycles.

Adding the message transmission or circuit hold time time \( d \) is not difficult, but leads to a slightly more complicated recursive equation. Consider incorporating the time \( d \) into the last stage of the request's
advance. In other words, when the request travels through the last stage an additional delay of \(d\) is added due to the message delivery time. Define \(g^N(p, \alpha, j)\) to be the number of slots required to completely fulfill a set of request (i.e. set up links and transfer messages) for a set of requests with input distribution vector \(p\) at the input to the \(j^{th}\) stage of an \(N\)-input network. One can derive a similar recursive equation for \(E[g^N(p, \alpha, k)]\), and obtain

\[
E[g^N(p, \alpha, k)] =
\]

\[
= E[g^N(S_{k-1} \cdot F(p, \alpha), \alpha, k - 1)]
\]

\[
+ E[g^N(S_{k-1} \cdot R(p, \alpha), \alpha, k - 1)]
\]

\[
+ 2 \prod_{i=1}^{N/2} (1 - p_{i1})(1 - p_{i2})
\]

\[
- \prod_{i=1}^{N/2} \left(1 - p_{i1}p_{i2} \left(\alpha_{i,k}^2 + (1 - \alpha_{i,k})^2\right)\right).
\]

The initial step of the recursion can be shown to be:

\[
E[g^N(p, \alpha, 1)] =
\]

\[
= (d + 1) \left\{2 \prod_{i=1}^{N/2} (1 - p_{i1})(1 - p_{i2})
\right.

\[
- \prod_{i=1}^{N/2} \left(1 - p_{i1}p_{i2} \left(\alpha_{i,1}^2 + (1 - \alpha_{i,1})^2\right)\right)\right\}.
\]

From equations (4) and (5), we can compute \(E[g^N(p, \alpha, k)]\) for any choice of the system parameters in a straightforward manner.

4 Results

To compare the performance of this dominating system to the actual system, a simulation of the actual system was used. The MIN used for this comparison is a square Omega network, chosen for simplicity in its connection pattern (\(S_k = S_1\)). We compare performance based on a hot spot model, i.e. for a given device with a request, one output is chosen as the destination for this request with probability \(h\) and the remaining outputs chosen uniformly with the remaining probability. The simulation does not add 'dummy' requests and represents the performance of the actual gated-hold protocol. Simulation results are based on 1000 trials.

We choose, for simplicity, to display results with uniform input distributions with marginal probability \(p\). Thus performance can be parameterized by the quantities \(p\) and \(h\). Results indicate that for small \(p\), the dominating system forms a tight bound on the performance of the actual system for various choices of the hot factor \(h\). In addition for small values of \(N\), the bound is tight for all values of \(p\) and \(h\). Figure 3 shows the bound versus actual performance for an 8-input network with uniform input distribution with marginal probability \(p\) and various hot factors. Figure 4 shows the same results for a 32-input network. The bound is computed and shown for a 128-input network for several values of \(h\) (Fig. 5). Because of the computational burden, a simulation is not included.

We investigate the dependence of the network cycle time on \(h\). For an 8-input network, Fig. 6 illustrates the effect of \(h\) on network performance for various \(p\)'s. For the values of \(p\) shown the network suffers roughly the same relative degradation for an increase in the value of \(h\). Note also that the network is not sensitive to small deviations from a uniform output distribution. For example, in this 8-input network, when \(h = .375\), three times the amount of traffic in a uniform network is incident on the hot output, but a degradation of not more than ten percent is experienced. The same results are shown for a 32-input network (Fig. 7). Five times \((h = .16)\) the traffic in a network with uniform output distributions can be directed to the hot output with only a ten percent degradation in performance. Thus, in non-stationary traffic, the network will not become 'saturated' when traffic with mild non-uniform output demands arise.

To examine the limitations of the blocking network on this protocol, we consider a non-blocking network (Batcher-Banyan) with a gated input strategy as in [10]. Comparison suggests that for small values of \(d\), the blocking network can outperform a non-blocking network of similar complexity (Figs. 8 and 9).

This bound can also be used to compute performance of a network with uniform input traffic and uniform output traffic. This bound is compared to one in [10] and is shown to be somewhat tighter for large values of \(p\) (Fig. 10). For this comparison, the input distribution was chosen to be uniform and the switching probabilities were selected as \(.5\). This bound is tighter because it allows the output probabilities from a switch to take on non-uniform probabilities which yields a requests addition strategy that adds less traffic.

5 Conclusions

In this paper, a bound on the performance of a MIN with non-uniform input distributions and non-uniform switching probabilities is derived. The protocol used was a gated-hold strategy introduced in [10]. The bounds computed are tight for both uniform and non-uniform switching probabilities introduced in [10]. The bounds computed are tight for both uniform and non-uniform switching probabilities for small values of \(p\). For small and moderate values of \(N\), the bounds
are tight for all \( p \). As \( N \) grows, the bounds become increasingly looser. Results indicate that this protocol is relatively insensitive to small 'hot spots', i.e. output distributions where a relatively small percentage of traffic is incident on the hot output. This is a favorable attribute for networks with traffic that exhibits non-stationary connection requirements. If a system is extremely sensitive to periods of traffic with small deviations in output distribution, backlogs can occur and cause unnecessary congestion in the system.

This work is being extended to include a more realistic input traffic model. Stochastic bounds are being evaluated to replace the sample path bound derived here. In addition, this bounding method is being applied to switching networks with different topologies.

Overall, the methods used in this analysis seem to provide a promising alternative to approximating the stochastic evolution of these networks.

References


Figure 5: Performance bound of 128-input network with hot output with factor $h$: $d = 0$

Figure 6: Bound for Expected Cycle Time of 8-input network versus hot factor $h$ parameterized by $p$: $d = 10$.

Figure 7: Bound for Expected Cycle Time of 32-input network versus hot factor $h$ parameterized by $p$: $d = 10$.

Figure 8: Blocking versus Non-Blocking Performance for $N = 8$, $d = 10$ for hot factor $h$.

Figure 9: Blocking versus Non-blocking Performance for $N = 32$, $d = 10$ for hot factor $h$.

Figure 10: Actual performance of 32-input network versus two upper bounds