MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

A.I. Memo No. 1337 December, 1991

Calculation of Blocking Probabilities in Multistage Interconnection Networks with Redundant Paths

Patrick G. Sobalvarro

Abstract

The blocking probability of a network is a common measure of its performance. There exist means of quickly calculating the blocking probabilities of Banyan networks; however, because Banyan networks have no redundant paths, they are not inherently fault-tolerant, and so their use in large-scale multiprocessors is problematic. Unfortunately, the addition of multiple paths between message sources and sinks in a network complicates the calculation of blocking probabilities. A methodology for exact calculation of blocking probabilities for small networks with redundant paths is presented here, with some discussion of its potential use in approximating blocking probabilities for large networks with redundant paths.

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the laboratory's artificial intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contracts N00014-88-K-0825 and N00014-85-K-0124, and in part by the Defense Advanced Research Projects Agency under contract N00014-87-K-0825. The author's work was supported by the Charles Stark Draper Laboratory through a Draper Laboratory Fellowship.

1 Introduction

The M.I.T. Artificial Intelligence Laboratory's Transit Project is developing high-performance networks for large-scale parallel computers. One of the project's aims is to make these networks fault-tolerant. This has been difficult in traditional multistage interconnection networks for parallel computers, because these networks have often been Banyan networks [9, 10]. Banyan networks do not have redundant paths [2], and thus the failure of a switching element will necessarily cut off communication between at least one message source and one message sink in the network.

The addition of redundant paths can enhance fault-tolerance. Unfortunately, it also creates problems for the traffic theorist. In a Banyan network, if one assumes messages at the inputs are generated by independent processes, the presence or absence of messages at the inputs of any switch in the network is independent of the presence or absence of messages at the other inputs of that switch. Thus the analysis of blocking probabilities in Banyan networks is simplified [5]. When redundant paths are allowed, independence is violated.

A similar problem has been studied in the context of telephone switching systems [4]. However, in telephone switching systems the model is one of a circuit-switched network where the holding time for circuits varies. Furthermore, in the methods described in [4], it is assumed that the networks modeled are symmetric; because there are classes of asymmetric networks that are of interest, and because we are partly interested in calculating blocking probabilities in the presence of (asymmetric) faults, these methods are not satisfactory.

The number of equations that must be solved in order to find the exact blocking probability of a general network is probably exponential in the number of communications channels entering a stage in a network; however, there appears to be no proof of this in the literature. Still, even if large problems are intractable, a method of exact solution is useful because it allows some evaluation of approximation methods through comparison with exact solutions for small problems, and also because it can be used as the basis for a Monte Carlo approximation method. This is the approach taken by Harvey and Hills in [3]. Harvey and Hills were considering circuit-switched telephone networks with unique paths; but their approach, which was to find approximate solutions of exact equations, rather than exact solutions to approximate equations, can still be of use here.

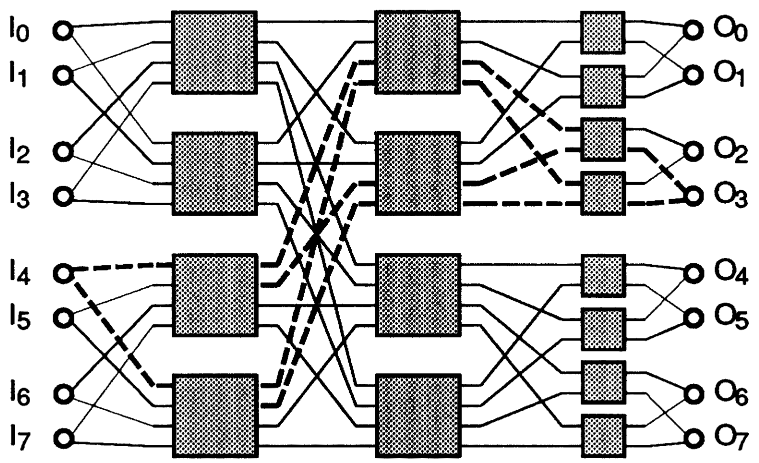

Figure 1: An 8×8 deterministically-interwired network with redundant paths. There are a number of different paths from any source to any sink, to increase fault-tolerance; redundant paths from message source 4 to sink 3 are highlighted. Routing is oblivious, with stochastic concentration. This wiring scheme is from [1].

2 Problem Statement

Consider a multistage, unbuffered, synchronous, packet-switched network with redundant paths. Such a network might look like the one depicted in Figure 1. Messages enter the network on the channels leading from the message sources, which in Figure 1 are depicted on the left side and labeled I_0 through I_7. The network has multiple stages: if we consider the stage consisting of all the sources to be stage 0, then stage 1 consists of the column of switching elements connected directly to the sources; stage 2 the column of switches to the right of stage 1, etc. The message sinks are the nodes on the right side, labeled O_0 through O_7.

The processes generating messages at the sources are independent and memoryless. With some specified probability p_i, each source i generates or fails to generate a single message at the beginning of each cycle. Each generated message is directed to a stage 1 switch via one of the channels connecting the source to the network. The channel by which to send the message is chosen randomly, and each channel has the same probability of being chosen; i.e., routing is *oblivious.* The network is synchronous: at each cycle messages move from stage i to stage $i+1$. It is also unbuffered: if a message is blocked at some stage, it is considered to be lost, and does not in any way affect the future states of the system.

In the networks we model, the set of output channels of a switching element is divided into nonempty disjoint subsets called *logical directions.* At each cycle, the switching element directs each incoming message in one logical direction. We assume *uniform addressing* here; messages are equally likely to be addressed in any one of the logical directions. We use *stochastic concentration*:

- If there are fewer messages or exactly the same number of messages directed in the logical direction as there are channels in that logical direction, then the channels that will carry the messages are chosen randomly, with uniform probability.
- If there are more messages directed in a logical direction than there are channels in that direction, the messages that can be carried are chosen with uniform probability, and the other messages are blocked and lost.

We refer to a switching component with M input channels and N logical directions, each consisting of K channels, as an "$M \times N$, dilation K switch."[1]

Having defined our model, let us return to the network of Figure 1. The switches here are 4×2, dilation 2 switches, except at the last stage, where they are simply 2×2 (dilation 1) switches. In the 4×2, dilation 2 switches, the top two output channels constitute one logical direction, and the bottom two constitute another.

We may state our question as follows: What is the probability that an arbitrary message entering the network at a channel leading from a source reaches a channel leading to a sink? We answer this question by finding the probability mass functions of the loads on channels leading to sinks.

[1]One such switch is the RN1 switching component, designed at the M.I.T. Artificial Intelligence Laboratory's Transit Group. The RN1 switching component, intended for use in interconnection networks for parallel multiprocessors, is an 8×4, dilation 2 switch. It is described further in [7].

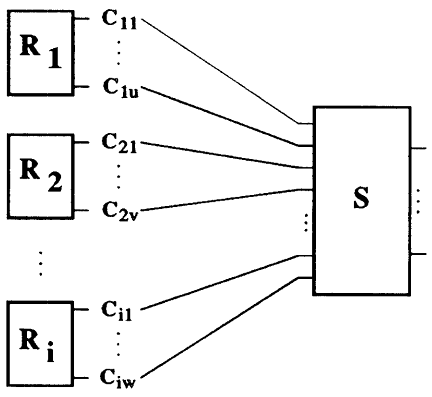

Figure 2: Interstage wiring. Note that no subset of the channels depicted need be mutually independent in a network with redundant paths.

3 The Joint Probability Mass Function of an Aggregate of Channels

Suppose that the input channels of a switch S, depicted in Figure 2, are connected to several switches $R_1, R_2, \ldots, R_i$. Let us use the random variable L to denote the entire output loading configuration of S at some specified discrete time t, so that $\mathrm{P}\{L = l\}$ is the probability that the output channels of the switch have some particular loads designated in their aggregate by l during cycle t.

Now consider the loads on the input channels $C_{11}, \ldots, C_{iw}$ at cycle $t-1$. (Because we assume a synchronous, unbuffered network with memoryless processes generating the messages at the inputs, only the cycle before cycle t is of interest.) Let us denote the loads on the input channels at cycle $t-1$ with the random variables $L_{C_{11}}, \ldots, L_{C_{iw}}$.

In order to find the joint probability mass function of the loads on the output channels of S, we condition on the loads on the input channels:

$$\begin{aligned}\mathrm{P}\{L = l\} = \sum_{l_{C_{11}}, \ldots, l_{C_{iw}}} & \mathrm{P}\{L = l \mid L_{C_{11}} = l_{C_{11}}, \ldots, L_{C_{iw}} = l_{C_{iw}}\} \cdot \\ & \mathrm{P}\{L_{C_{11}} = l_{C_{11}}, \ldots, L_{C_{iw}} = l_{C_{iw}}\} \qquad (1)\end{aligned}$$

where the sum is over all tuples $l_{C_{11}}, \ldots, l_{C_{iw}}$ with elements in $\{0, 1\}$.

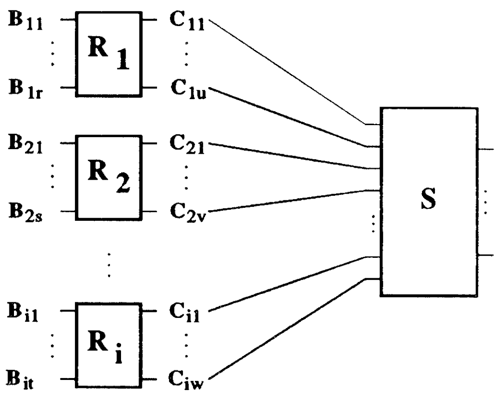

Figure 3: Although the probabilities of the message loads on the channels $C_{11}, \ldots, C_{iw}$ are not in general independent, the loads on the subset of channels from each switching element are independent given the message loads on the input channels $B_{11}, \ldots, B_{it}$.

Suppose that we can compute $\mathrm{P}\{L = l \mid L_{C_{11}} = l_{C_{11}}, \ldots, L_{C_{iw}} = l_{C_{iw}}\}$.[2] In order to compute the probability of an output loading configuration of S we will still need to find the joint probability mass function of the channel loads $L_{C_{11}}, \ldots, L_{C_{iw}}$. In a Banyan network, it would be easy to compute this function; it would simply be the product of the probability mass functions of the loads on the individual channels, as channel loads in a Banyan network are independent.[3] In a network with redundant paths, however, the loads on these channels are not in general independent, as they may derive from the same sources, and a message from a single source that has traveled one path in the network cannot be traveling along another path. Thus another method must be used.

In Figure 3, we see that the input channels $C_{11}, \ldots, C_{iw}$ of switch S are the output channels of switches $R_1, \ldots, R_i$. Let us call the loads on the input channels to these switches $L_{B_{11}}, \ldots, L_{B_{it}}$. We may now calculate $\mathrm{P}\{L_{C_{11}} = l_{C_{11}}, \ldots, L_{C_{iw}} = l_{C_{iw}}\}$ by conditioning on the values of the variables $L_{B_{11}}, \ldots, L_{B_{it}}$. We have

$\mathrm{P}\{L_{C_{11}} = l_{C_{11}}, \ldots, L_{C_{iw}} = l_{C_{iw}}\} =$

[2] An expression for this conditional probability is derived in Appendix A.

[3] See Appendix B.

$$\sum_{l_{B_{11}},\ldots,l_{B_{it}}} \mathrm{P}\{L_{C_{11}} = l_{C_{11}},\ldots,L_{C_{iw}} = l_{C_{iw}} \mid L_{B_{11}} = l_{B_{11}},\ldots,L_{B_{it}} = l_{B_{it}}\} \cdot$$
$$\mathrm{P}\{L_{B_{11}} = l_{B_{11}},\ldots,L_{B_{it}} = l_{B_{it}}\} \qquad (2)$$

where the sum is over all tuples $l_{B_{11}},\ldots,l_{B_{it}}$ with elements in $\{0,1\}$.

The loads on the output channels of these switches are not in general mutually independent. However, let us partition them into subsets according to the switch at which they originate, so that for the channels shown in Figure 3 we would have the subsets $\{C_{11},\ldots C_{1u}\},\{C_{21},\ldots C_{2v}\},\ldots,\{C_{i1},\ldots,C_{iw}\}$. Note that, under the assumption of uniform addressing, and given the loads on the channels $B_{11},\ldots,B_{it}$, the loads on the switch output channel subsets *are* mutually independent. That is, if we know the input loads for the switches $R_1,\ldots,R_i$, then the loading probabilities for the output channels of each of the switches do not depend on the output loads of any other switch. We may use this fact to derive the joint probability mass function of the loads on the output channels $C_{11},\ldots,C_{iw}$ by conditioning on the input channel loads. We have then

$$\mathrm{P}\{L_{C_{11}} = l_{C_{11}},\ldots,L_{C_{iw}} = l_{C_{iw}}\} =$$
$$\sum_{l_{B_{11}},\ldots,l_{B_{it}}} \mathrm{P}\left\{L_{C_{11}} = l_{C_{11}},\ldots,L_{C_{1u}=l_{C_{1u}}} \mid L_{B_{11}} = l_{B_{11}},\ldots,L_{B_{1r}} = l_{B_{1r}}\right\} \cdot$$
$$\mathrm{P}\left\{L_{C_{21}} = l_{C_{21}},\ldots,L_{C_{2v}=l_{C_{2v}}} \mid L_{B_{21}} = l_{B_{21}},\ldots,L_{B_{2s}} = l_{B_{2s}}\right\} \cdot$$
$$\ldots \quad \cdot$$
$$\mathrm{P}\{L_{C_{i1}} = l_{C_{i1}},\ldots,L_{C_{iw}} = l_{C_{iw}} \mid L_{B_{i1}} = l_{B_{i1}},\ldots,L_{B_{it}} = l_{B_{it}}\} \cdot$$
$$\mathrm{P}\{L_{B_{11}} = l_{B_{11}},\ldots,L_{B_{it}} = l_{B_{it}}\} \qquad (3)$$

where the sum is once again over all tuples $l_{B_{11}},\ldots,l_{B_{it}}$ with elements in $\{0,1\}$.

The conditional probabilities can be evaluated as described in Appendix A, and $\mathrm{P}\{L_{B_{11}} = l_{B_{11}},\ldots,L_{B_{it}} = l_{B_{it}}\}$ can be evaluated recursively by means of Equation (3), until the channels $B_{11},\ldots,B_{it}$ correspond to sources. If these channels originate at message sources, then we substitute instead the probability mass functions corresponding to sources. We may simply take the product of these functions for the sources in question, as in our model the processes generating messages at the sources are mutually independent.

If source i, depicted in Figure 4, generates a message with probability p_i and has k channels into the network, then we have for the loads on the

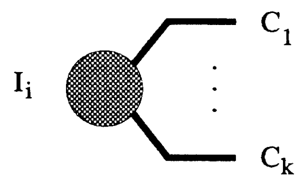

Figure 4: The source I_i generates a single message at each cycle with probability p_i. The message is transmitted with uniform probability over a randomly picked channel in the set $\{C_1, \ldots, C_k\}$.

channels $C_1, \ldots, C_k$ the joint probability mass function

$$\mathrm{P}\{L_{C_1} = l_{C_1}, \ldots, L_{C_k} = l_{C_k}\} = \begin{cases} 1 - p_i & \text{if all the } l_{C_j} \text{ are } 0 \\ \frac{p_i}{k} & \text{if exactly one } l_{C_j} \text{ is } 1, \\ & \text{and the rest are } 0 \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

4 Automatic Calculation of Blocking Probabilities

It will be clear that the automatic calculation of blocking probabilities by this means will require a great deal of time. Suppose we have a computer program that calculates the blocking probabilities for a network in the most obvious way, by finding the joint probability mass function of the channels leaving the final stage, using Equation (3) recursively. In the worst case, we can imagine a network where there are N stages and M dependent channels between each of the N stages, and the joint probability mass function of all of the channels between each of the stages must be formed. The domain of the joint probability mass function for each stage then is of size 2^M, each value being calculated as a sum over 2^M terms. Assuming the time to calculate each of the terms summed over in Equation (3) is $\mathrm{O}(M)$, we have then $\mathrm{O}\left(NM2^{2M}\right)$ for the worst-case performance.

The performance on some networks can be better than this, however. Suppose that we need to calculate $\mathrm{P}\{L_{C_1} = l_{C_1}, \ldots, L_{C_n} = l_{C_n}\}$. Let $\mathcal{S}(c)$ denote the set of source nodes from which messages can reach channel c. If we can partition the set of channels $\{C_1, \ldots, C_n\}$ into disjoint subsets

$S_1, \ldots, S_m$ such that for any $C_1 \in S_i$ and $C_2 \in S_j$, $i \neq j$, $\mathcal{S}(C_1) \cap \mathcal{S}(C_2)$ is empty, then the loads on the channels in each subset S_i are independent of the loads on the channels in any and all of the other subsets in the partition.[4] Then the expression $\mathrm{P}\{L_{C_1} = l_{C_1}, \ldots, L_{C_n} = l_{C_n}\}$ can be factored into the product of m joint probability mass functions, one for each subset S_i. In the limiting case of a Banyan network, a complete factoring will be possible for every set of channels, and the summation itself can be factored, so that the worst case performance for a Banyan network of N stages with M channels between the stages becomes $\mathrm{O}(NM)$.

A program has been written to evaluate the joint probability mass function of the loads on specified channels in a multistage interconnection network. The program is given a symbolic description of the interconnection network. The program uses the network representation to build an internal structure in which (for example) information about independence of channel loads has been precomputed, and channels have been assigned names generated from the names of the their nodes of origin and destination. One can then assign message generation probabilities to the sources and query the program for the probability mass function of interest. The result is numerical, as in the example below:

```
> (setq d8x8 (parse-multistage-network
              deterministically-interwired-8x8-rep))
#<MULTISTAGE-NETWORK 8x8>
> (jlpmf '(tt6-o7-0 tt7-o7-0) d8x8)
(#S(JLPMF-PART CHANNELS (#<CHANNEL TT6-O7-0>
                         #<CHANNEL TT7-O7-0>)
               NUMBER-OF-CHANNELS 2
               VECTOR #(10321939817/17179869184
                        2931771091/17179869184
                        2931771091/17179869184
                        994387185/17179869184)))
```

Here we have calculated the joint probability mass function of the loads on two channels leading from two 2×2 switches to sink $O7$ in the network of Figure 1, given a probability of transmission in each message source of 1/2; we assume here, as in [1], that a message sink can receive two messages during a single cycle.

To find the blocking probability of the network, we can form the probability of successful message transmission as the ratio of the expected number

[4]As will be seen from the argument in Appendix B.

of messages entering the network to the expected number of messages arriving at sinks. Because of the symmetry of the network, all the channels leading to sinks have identical loading probabilities, and so we can simply sum the expectations of their loads. We have then that the expected number of messages arriving at a single sink is

$$\begin{aligned} 1 \cdot \mathrm{P}\{L_{\mathrm{TT6\text{-}O7}} = 1, L_{\mathrm{TT7\text{-}O7}} = 0\} &+ 1 \cdot \mathrm{P}\{L_{\mathrm{TT6\text{-}O7}} = 0, L_{\mathrm{TT7\text{-}O7}} = 1\} \\ &+ 2 \cdot \mathrm{P}\{L_{\mathrm{TT6\text{-}O7}} = 1, L_{\mathrm{TT7\text{-}O7}} = 1\} \\ \approx 0.457 & \end{aligned}$$

and the expected number of messages arriving at all sinks during any cycle is then $8 \cdot 0.457 \approx 3.66$.

Because the expected number of messages entering the network is $8 \cdot \frac{1}{2} = 4$, we have that the aggregate probability of successful message transmission in this network at a loading factor of $1/2$ is

$$\frac{\mathrm{E}[\text{messages arriving at sinks}]}{\mathrm{E}[\text{messages injected by sources}]} \approx 0.914$$

and thus the blocking probability is approximately 0.086.

We plot for the network of Figure 1 the probability of successful message transmission versus the probability that a source transmits in Figure 5.

The implementation internally records joint probability mass functions so that they need not be recomputed. Although the asymptotic performance is in general pessimal, the implementation has been coded with some attention to performance, because much of the same code is likely to be used in an approximation scheme.

5 Conclusions; Future Work

We have presented a means of exact calculation of the blocking probability of a multistage network with redundant paths, and demonstrated its use in a program that automatically calculates blocking probabilities and exploits independence of channel loading probabilities where this is possible.

The implementation described cannot be used to calculate the blocking probabilities of networks with much more path redundancy than the one of Figure 1. We might consider an implementation that could exploit the symmetry exhibited by some multistage networks, but such an implementation could still not be used on a network like that in Figure 6, in which

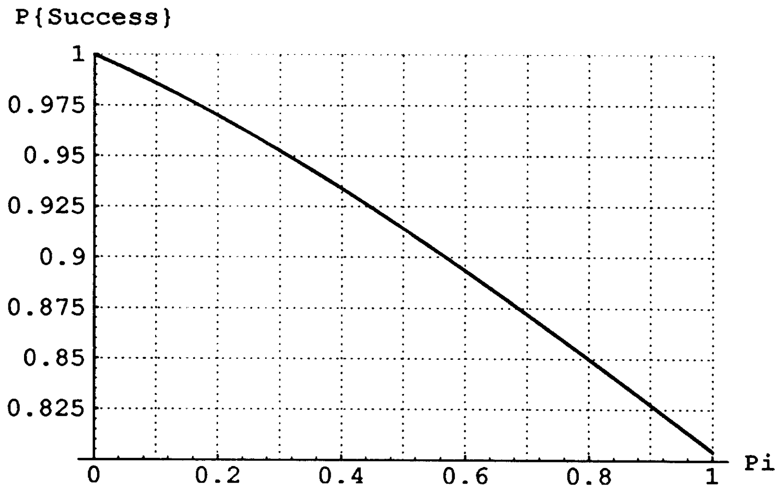

Figure 5: The probability of successful message transmission (`P{Success}`) plotted against the the source transmission probability (`Pi`) for the network of Figure 1.

the wiring in the first and second stages is not symmetric and is in fact randomly generated. That such networks are of interest is demonstrated in [6].

We are investigating the possibility of approximately solving these systems of equations. Our hope is that the approximate solution of the equations will yield approximations of higher confidence in fewer steps than would direct simulation of the network.

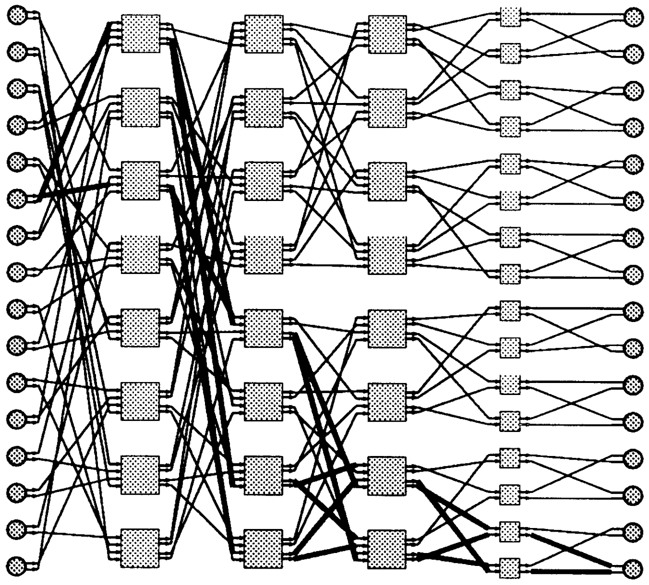

Figure 6: A 16×16 network with random interwiring in the first and second stages. The figure is from [1].

A Probability Mass Functions of Switching Component Output Channels

Suppose we have an $M \times N$, dilation K switch. We may form the conditional probability mass function of the loads on the output channels, given the input load, by conditioning. Say that the random variable $L_{f,g}$ represents the load on the g^{th} channel in the f^{th} logical direction. Now for ease of representation, let us say that the random variable L represents the entire output configuration of the switch, and takes on values l that are $N \times K$ arrays whose elements $l_{f,g}$ are 0 if the channel in question carries no message, or 1 if it carries a message. Thus in our notation the event we denote by

$$\{L = l\}$$

is equivalent to the event

$$\begin{array}{ll} \{ & L_{1,1} = l_{1,1}, L_{1,2} = l_{1,2}, \ldots, L_{1,K} = l_{1,K}, \\ & L_{2,1} = l_{2,1}, L_{2,2} = l_{2,2}, \ldots, L_{2,K} = l_{2,K}, \\ & \vdots \\ & L_{N,1} = l_{N,1}, L_{N,2} = l_{N,2}, \ldots, L_{N,k} = l_{N,K} \quad \} \end{array}$$

We wish to evaluate the expression $\mathrm{P}\{L = l \mid L_{C_1} = l_{C_1}, \ldots, L_{C_M} = l_{C_M}\}$. The switches that we are modeling do not distinguish in any way between messages arriving at distinct input channels. Thus the loads on the individual input channels $C_1, \ldots, C_M$ are not individually of significance, but their sum is. Say that $\sum_{i=1}^{M} l_{C_i} = j$. For simplicity of expression, while working in an event space where $L_{C_1} + \ldots + L_{C_M} = j$, we use the superscript j. Thus we say $\mathrm{P}\{L = l \mid L_{C_1} = l_{C_1}, \ldots, L_{C_M} = l_{C_M}\} = \mathrm{P}\{L^j = l\}$. Now, to form the conditional probability $\mathrm{P}\{L = l \mid J = j\}$, we condition on the number of messages directed in each logical direction. If the random variable C_i represents the number of messages routed in logical direction i, we can condition:

$$\mathrm{P}\left\{L^j = l\right\} = \sum_{d_1, \ldots, d_N} \mathrm{P}\left\{L^j = l \mid D_1 = d_1, \ldots, D_N = d_n\right\} \cdot \\ \mathrm{P}\{D_1 = d_1, \ldots, D_N = d_n\} \qquad (5)$$

where the sum is over all N-tuples $d_1, \ldots, d_N$ such that each $d_i \geq 0$ and $\sum_{i=1}^{N} d_i = j$.

Under uniform addressing, the probability that of the j arriving messages, d_1 are directed in direction 1, d_2 in direction 2, and so on, is simply multinomial, so that

$$\mathrm{P}\{D_1 = d_1, \ldots, D_N = d_N\} = \binom{j}{d_1, \ldots, d_N} \left(\frac{1}{N}\right)^j \qquad (6)$$

Now let us evaluate $\mathrm{P}\{L^j = l \mid D_1 = d_1, \ldots, D_N = d_n\}$. Say that c_i is the number of messages output in direction i when $L^j = l$; that is, $c_i = \sum_{g=1}^{K} l_{i,g}$. This number is not the same as d_i, because if there are more than K messages to be output in a K-wide direction, some messages are dropped and lost. If c_i messages are output, then under stochastic concentration the channels are picked with uniform probability, and so the probability of any single configuration will be $1/\binom{K}{c_i}$. Thus

$$\mathrm{P}\Big\{L^j = l \mid D_1 = d_1, \ldots, D_N = d_N\Big\} = \begin{cases} 0 & \text{if for any } i, c_i \neq \min(d_i, K) \\ \displaystyle\prod_{i=1}^{N} \frac{1}{\binom{K}{c_i}} & \text{otherwise} \end{cases} \qquad (7)$$

where $c_i = \sum_{g=1}^{K} l_{i,g}$.

Combining Equations (5), (6), and (7), we have

$$\mathrm{P}\{L = l \mid L_{C_1} = l_{C_1}, \ldots, L_{C_M} = l_{C_M}\} = \left(\prod_{i}^{N} \frac{1}{\binom{K}{c_i}}\right) \sum_{d_1, \ldots, d_N} \binom{j}{d_1, \ldots, d_N} \left(\frac{1}{N}\right)^j \qquad (8)$$

where $c_i = \sum_{g=1}^{K} l_{i,g}$, $j = \sum_{i=1}^{M} l_{C_i}$, and the sum is over the N-tuples $d_1, \ldots, d_N$ such that $\sum_{i=1}^{N} d_i = j$ and for each i, $c_i = \min(d_i, K)$.

B Loads on Banyan Network Channels at a Single Stage are Independent

A proof for the special case of delta networks is presented in [8]; here we present a different proof for the general case.

The proof is entirely straightforward. Note first that, if messages are generated at source nodes by mutually independent random processes, and the sets of messages on distinct channels entering a switching node originate at disjoint sets of source nodes, then the loads on those channels are necessarily independent.

We now claim that the sets of messages on distinct channels entering any switching node in a Banyan network satisfy this criterion: i.e., they originate at disjoint sets of sources.

For, consider: if channel A and channel B are two channels entering a switching node, and a message on channel A and a message on channel B originate at a single source, then it must be the case that at least two paths exist from that source to any sinks accessible from the switching node: one path that uses channel A and one that uses channel B. But this is impossible in a Banyan network, as Banyan networks are in fact those in which there is exactly one path from each source to each sink.

Thus the sets of messages on distinct channels entering any switching node in a Banyan network must originate at disjoint sets of sources, and so the loads on the channels entering any switching node in a Banyan network must be mutually independent, as was to be proved.

References

[1] Chong, F., Egozy, E., and DeHon, A. "Fault Tolerance and Performance of Multipath Multistage Interconnection Networks," in *Advanced Research in VLSI: MIT/Brown Conference 1992,* edited by Thomas F. Knight Jr. and John Savage, MIT Press, to be published March 1992.

[2] Goke, L. R., and Lipovski, G. J. "Banyan Networks for Partitioning Multiprocessor Systems," in *Proceedings of the First Annual Symposium on Computer Architecture*, 1973.

[3] Harvey, C., and Hills, C. R. "Determining Grades of Service in a Network," in *Ninth International Teletraffic Conference*, Torremolinos, Spain, October, 1979.

[4] Hui, J. Y. "Switching and Traffic Theory for Integrated Broadband Networks," pp. 246-270. Kluwer Academic Publishers, Boston, 1990.

[5] Knight, T. F., and Sobalvarro, P. G. "Routing Statistics for Unqueued Banyan Networks." M.I.T. Artificial Intelligence Laboratory Memo No. 1101, September, 1990.

[6] Leighton, T., and Maggs, B. "Expanders Might be Practical: Fast Algorithms for Routing Around Faults in Multibutterflies," in *30th Annual Symposium on Foundations of Computer Science,* IEEE Computer Society Press, November, 1989.

[7] Minsky, H., DeHon, A., and Knight, T. F. "RN1: Low-latency, Dilated, Crossbar Router," in *Hot Chips Symposium III*, 1991.

[8] Patel, J. H. "Performance of Processor-Memory Interconnections for Multiprocessors," in *IEEE Transactions on Computers,* Vol. C-30, No. 10, October 1981.

[9] Pfister, G. F. *et al.,* "The IBM Research Parallel Processor Prototype (RP3): Introduction and Architecture," in *Proceedings of the 1985 International Conference on Parallel Processing*, August, 1985.

[10] Rettberg, R., and Thomas, R. "Contention is no Obstacle to Shared-Memory Multiprocessing," in *Communications of the ACM,* Vol. 29, No. 12, December, 1986.