ANALYZING THE CONNECTIVITY AND BANDWIDTH OF MULTI-PROCESSORS WITH MULTI-STAGE INTERCONNECTION NETWORKS [1]

Israel Koren [2] and *Zahava Koren*

Dept. of Electrical and Computer Engineering
University of Massachusetts, Amherst, MA 01003

Abstract

When implementing multi-processing systems consisting of a large number of processors, memory modules and interconnection switches, we must expect some of the system elements to become faulty. These faults may be the result of manufacturing defects or failures that occur while the system is already in operation. If the system is allowed to continue its operation in the presence of a few faulty elements, we need to predict the performance of the degraded system.

In this paper we analyze the performance of multi-processor systems with a multi-stage interconnection network in the presence of faulty elements. We propose the use of two measures for performance, namely, bandwidth and connectivity. We then derive expressions for these measures for a non-redundant system and for a system with redundancy in its interconnection network. Finally, we compare the two systems through some numerical examples.

I. Introduction

Recent advances in VLSI technology and development of new computer-aided design tools like silicon compilers, enable the design and implementation of multi-processing systems consisting of a very large number of processors. One important class of these multi-processing systems includes the shared-memory multi-processors where all processors can access a set of memory modules through an interconnection network. This interconnection network can be a crossbar network, a multiple bus network or a multi-stage interconnection network.

[1] This work was supported in part by NSF under contract DCR-85-09423.

[2] On leave from the Technion, Haifa 32000, Israel

When implementing a complex multi-processor, some of its elements (like processors, memory modules or interconnection switches) are expected to become faulty. These faults may be the result of manufacturing defects or failures that occur while the system is already in operation. In many cases the faulty elements can not be immediately repaired or replaced and we would still like to use the system at a degraded rate of performance until a repair and/or replacement can take place.

An important question then is how well the gracefully degrading system performs in the presence of faults. This question may arise in different situations. In one case faulty elements can be neither repaired nor replaced. An example might be a multi-processor integrated on a small number of large area VLSI chips or even wafers where some defective elements may be present (e.g., [4]). A different situation is when the faulty elements can be replaced but the mean time to repair or replacement is considered to be too long. An example might be a real-time computing system where even a relatively short down-time period may be intolerable.

In this paper we attempt to answer the above question for one type of shared-memory multi-processors, namely, those interconnected through a multi-stage network.

Multi-stage interconnection networks were proposed as a cost-effective alternative to the expensive crossbar networks. However, these networks are inherently very sensitive to failures of any kind. They usually provide a unique path between any processor and any memory module and therefore, a single fault in any internal switch or link will render some memories unreachable from certain processors. Several schemes for introducing fault-tolerance into the architecture of these interconnection networks have been suggested in recent publications (e.g., [1], [3], [5], [6] and [8]). A survey of these and other schemes is presented in [2].

Most of the proposed schemes provide redundant paths between every source/-destination pair so that all single faults and many multiple faults can be tolerated. This is achieved by augmenting an existing topology either by adding an extra stage of switches (e.g., [1]), or by augmenting the switching elements and their interconnections (e.g., [5], [6] and [8]). The multiple disjoint paths provided by these schemes may in some cases also increase the performance of the network. For example, it has been shown in [8] that the bandwidth of their proposed fault-tolerant multi-stage network is comparable even to that of a crossbar network.

In this paper we examine the performance of multi-stage multi-processors (with and without redundancy) in the presence of faults. We also propose objective functions for measuring the performance of these systems. In the next section we present the proposed performance measures and introduce several basic assumptions and notations. In Section III the non-redundant network is analyzed. A similar analysis is then repeated in Section IV for a network with built-in redundancy, namely, the Extra Stage Cube network [1]. These networks are then compared through some numerical examples in Section V. Final conclusions are presented in Section VI.

II. Preliminaries and Notations

Consider N processors (where $N = 2^k$) connected to N memories through a multi-stage interconnection network designed out of 2×2 switches. Our analysis can be generalized to the case where the number of processors is not necessarily a power of 2, the number of memories is different from the number of processors and finally, the network is built from $a \times b$ switches. For the sake of clarity and brevity however, we restrict our discussion here to the above mentioned simpler case.

An $N \times N$ interconnection network with no redundancy is constructed of $k = logN$ stages, each containing $N/2$ switches as illustrated in Fig. 1. Redundant networks have a larger number of stages and/or more switches per stage and/or use more complex switches. Non-redundant networks and some networks with internal redundancy have been previously analyzed but in most previous studies it has been assumed that faults can occur only in the interconnection network while the processors and memories are assumed to be fault-free. Clearly, this assumption is invalid in our environment and we have to consider faulty processors and faulty memories in addition to faults in the interconnection network.

Let q_r denote the probability that a processor is faulty at some given time instant t and let $p_r = 1 - q_r$ denote the probability of a fault-free processor. If $t = 0$ then q_r is the probability that manufacturing defects have occurred in the processor. If $t > 0$ then q_r is the probability that the processor either had defects at $t = 0$ or became faulty later on. Similarly, we denote by q_m (p_m) the probability of a faulty (fault-free) memory and by q_l (p_l) the probability of a faulty (fault-free) link. Our fault model for the interconnection network is the link fault model (e.g., [2]), however, we allow multiple link faults so that switch faults are covered as well.

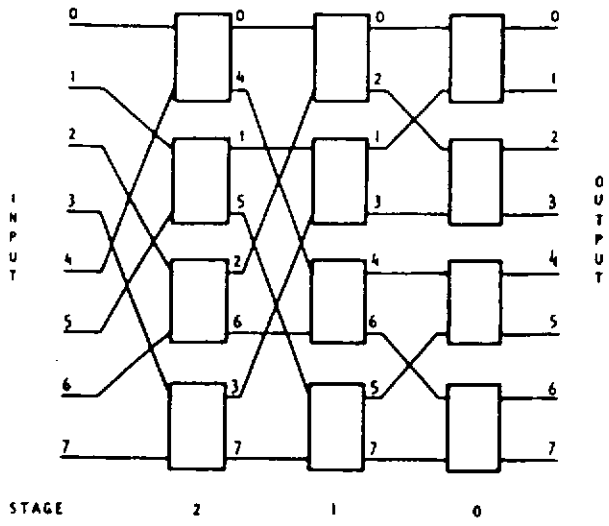

Figure 1: An 8×8 non-redundant interconnection network.

We are interested in comparing the performance of alternative architectures for multi-stage networks in this environment where processors, memories and links can fail. The performance measure to be used should capture the capabilities of the gracefully degrading multi-stage system, e.g., the number of fault-free processors and memories, the number of fault-free paths within the interconnection network, and the number of memory requests (from the processors) that can be transmitted through the interconnection network simultaneously.

A commonly used measure for the performance of an interconnection network is the *bandwidth* [7]. The bandwidth is defined as the expected number of requests for the shared memory which are accepted per cycle, given that each processor generates, with probability p_a, a request during a cycle, and that a request blocked at any stage is lost. The bandwidth measures the effect of blocking which results from the fact that in multi-stage interconnection networks paths are shared by two or more processor-memory pairs. A processor-memory connection can be blocked by a previously established connection even if the memories involved are distinct.

The bandwidth measure is concerned only with the interconnection network and is therefore, insufficient for our purposes. It is strongly affected by the amount of traffic in the network (i.e., the probability of request p_a) and hence, it provides only limited insight regarding the *connectivity* of the multi-processing system, i.e., the number of fault-free processors which are still connected to some fault-free memories. System connectivity can be the

basis for measuring the processing power of the system or its computational availability. In what follows we present two measures for connectivity to be used in addition to the bandwidth measure.

One proposed measure for connectivity is C - the average number of operational processor-to-memory paths where both processor and memory are fault-free. A shortcoming of this objective function is that it provides no indication on how many *distinct* processors and memories are still connected. We propose therefore, as an additional measure for connectivity, the tuple (N_r, N_m) where N_r denotes the average number of fault-free processors which are connected to at least one fault-free memory module and N_m is similarly defined for memories. Note that this tuple does not necessarily imply that a complete fault-free $N_r \times N_m$ interconnection network exists.

The bandwidth and the two connectivity measures, namely, C and the tuple (N_r, N_m), can together adequately characterize the capabilities of a multistage multi-processor in the presence of faulty elements.

III. A Non-redundant Interconnection Network

In this section we analyze a non-redundant interconnection network and derive expressions for its bandwidth and the different connectivity measures as previously defined. An analysis of the bandwidth for a fault-free network appears in [7]. In what follows we generalize it to an environment in which faults may be present.

We adopt here the simplifying assumption that the destinations of the memory requests are independent and uniformly distributed among the N memories. Therefore, the network bandwidth can be obtained by multiplying the number of memories N by the probability that a given memory module is non-faulty and has a request at its input. This last probability is calculated iteratively, following a path leading to this memory , i.e, the probability of a request on an output link of a switch is calculated from the probability that such a request has been accepted at the input links to the same switch.

To simplify our discussion we say that a link is in state 1 (0) if it has (has not) a request for the memory. A faulty link is considered to be in state 0. The probability of a request on a link is thus the probability that this link is in state 1. We assign numbers to the $k = logN$ stages in a descending order so that stage 0 is the last stage and its output links are connected to the memories, stage $(k-1)$ is the first one and its inputs are connected to the

processors (see Fig. 1). Consider a switch in stage i and denote its outputs $X^{(i)}, Y^{(i)}$. Its input links are the outputs of (two different) switches in stage $(i+1)$ and are denoted by $X^{(i+1)}$ and $Y^{(i+1)}$. Based on our assumption that memory requests are uniformly distributed among the memories, the probability that an incoming request will be routed to any output link is the same. Hence, it is sufficient to consider only a single output link and derive an expression for the probability that it is at state 1, i.e., $P\{X^{(i)} = 1\}$.

Since a request for a memory module can reach the output link of a switch through any of the two input links, the state probability $P\{X^{(i)} = 1\}$ of the given output link has to be calculated from the joint probabilities of these input links, i.e.,

$$P\{(X^{(i+1)}, Y^{(i+1)}) = (u, v)\}; \quad u, v = 0, 1.$$

This calculation is performed using transition probabilities which take into account the status (faulty or fault-free) of the (physical) input links and the destinations of the incoming requests. Since memory modules are assumed to be equivalent, the incoming requests are routed to any of the two output links with probability 0.5. Consequently, the transition probabilities between the two inputs and the output of a switch are,

$$\begin{aligned} &P\{X^{(i)} = 1 \;/\; (X^{(i+1)}, Y^{(i+1)}) = (0,0)\} = 0 \\ &P\{X^{(i)} = 1 \;/\; (X^{(i+1)}, Y^{(i+1)}) = (0,1)\} = \tfrac{1}{2}\, p_l \\ &P\{X^{(i)} = 1 \;/\; (X^{(i+1)}, Y^{(i+1)}) = (1,0)\} = \tfrac{1}{2}\, p_l \\ &P\{X^{(i)} = 1 \;/\; (X^{(i+1)}, Y^{(i+1)}) = (1,1)\} = p_l - \tfrac{1}{4}\, p_l^2 \end{aligned} \tag{3.1}$$

Note that only input link faults are taken into account. Faults at the output links are considered as input link faults at the next stage.

The state probability $P\{X^{(i)} = 1\}$ of the output link is given by,

$$P\{X^{(i)} = 1\} = \frac{1}{2}\, p_l \left[P\{(X^{(i+1)}, Y^{(i+1)}) = (0,1)\} + P\{(X^{(i+1)}, Y^{(i+1)}) = (1,0)\}\right]$$

$$+ p_l(1 - \frac{1}{4}\, p_l) \times P\{(X^{(i+1)}, Y^{(i+1)}) = (1,1)\} \tag{3.2}$$

For the non-redundant network the inputs into each switch are independent. Hence,

$$P\{(X^{(i+1)}, Y^{(i+1)}) = (u, v)\} = P\{X^{(i+1)} = u\} \times P\{Y^{(i+1)} = v\}; \quad u, v = 0, 1 \tag{3.3}$$

Using (3.3) and the following equation

$$P\{Y^{(i+1)} = 0\} = P\{X^{(i+1)} = 0\} = 1 - P\{X^{(i+1)} = 1\}$$

we obtain from (3.2) after some algebraic manipulations,

$$P\{X^{(i)}=1\} = p_l \times P\{X^{(i+1)}=1\} - \frac{1}{4}\ p_l^2 \times (P\{X^{(i+1)}=1\})^2 \qquad (3.4)$$

This expression is identical to the one derived in [7] if fault-free (i.e., $p_l = 1$) 2×2 switches are assumed.

This simple recursion formula enables us to calculate the successive state probabilities, starting from the processors outputs up to the memory inputs. For the processors output links we have

$$P\{X^{(k)}=1\} = p_a\ p_r \qquad (3.5)$$

Recursively, we calculate $P\{X^{(0)}=1\}$. To compute the bandwidth note that the memory and its input link can be faulty as well, hence,

$$BW = N \times P\{X^{(0)}=1\} \times p_m\ p_l \qquad (3.6)$$

The next part of this section is devoted to the analysis of the network connectivity, as measured by C - the average number of connected processor-memory pairs, and by N_r and N_m - the average number of processors connected to at least one memory, and of memories connected to at least one processor, respectively.

In a non-redundant interconnection network there is exactly one path between a processor and a memory and consequently, the calculation of C is straightforward. C is obtained by multiplying the number of processor-memory pairs by the probability of a fault-free path. The latter probability is,

$$p_r\ p_l^{k+1}\ p_m \qquad (3.7)$$

where $(k+1)$ is the number of links along the path. Therefore,

$$C = N^2 \times p_r\ p_l^{k+1}\ p_m \qquad (3.8)$$

Let Φ_r be the probability that a given processor (say processor 0) is fault-free and is connected to at least one fault-free memory. N_r is obtained by multiplying N by Φ_r. Due to the overlapping of the paths leading to the same processor, we must utilize the inclusion and exclusion principle to calculate the probability Φ_r. Define E_j as the event in which memory j is connected to processor 0. Φ_r can now be expressed in terms of the events E_j as the probability that at least one of the events E_j occurs,

$$\Phi_r = p(E_1 \cup E_2 \cup \ldots \cup E_N) \qquad (3.9)$$

The inclusion and exclusion formula states that

$$p(\cup E_j) = \sum_{i=1}^{N} (-1)^{(i-1)} W(i) \tag{3.10}$$

where $W(i)$ is the sum over all $\begin{pmatrix} N \\ i \end{pmatrix}$ subsets $\{j_1, j_2, ..., j_i\}$ of size i, of the probability that all paths in the subset are operational, namely,

$$W(i) = \sum_{\{j_1,...,j_i\}} P(E_{j_1} \cap E_{j_2} \cap ... \cap E_{j_i}) \tag{3.11}$$

For a subset of i paths to be operational, all links in the subset must be fault-free. Hence, the required probability $P(E_{j_1} \cap E_{j_2} \cap ... \cap E_{j_i})$ depends not only on i but also on the number of links that the paths in the given subset have in common, since each link must be taken into account exactly once. Let d denote the number of distinct links in the subset, then

$$P(E_{j_1} \cap E_{j_2} \cap ... \cap E_{j_i}) = p_m^i \; p_r \; p_l^d \tag{3.12}$$

and

$$W(i) = p_m^i \; p_r \sum_d S_{i,d} \; p_l^d$$

where $S_{i,d}$ is the number of subsets of size i which consist of exactly d distinct links. Note that d can be expressed as the sum $d_0 + d_1 + ... + d_k$, where d_n is the number of distinct links in the subset at level n. Also note that for any subset of size i, $d_k = 1$ and $d_0 = i$. Using combinatorial arguments which are omitted here for the sake of brevity, we obtain the following equation for $S_{i,d}$,

$$S_{i,d} = \sum_{d_0+d_1+...+d_k=d} \begin{pmatrix} 2 \\ d_{k-1} \end{pmatrix} 2^{d_1+...+d_{k-2}+2d_{k-1}-i} \prod_{n=1}^{k-1} \begin{pmatrix} d_n \\ d_{n-1} - d_n \end{pmatrix} \tag{3.13}$$

As an example, for the 8×8 interconnection network in Fig. 1 we obtain,

$$\begin{aligned}
W(1) &= p_m \; p_l \; p_r \; 8p_l^2 \\
W(2) &= (p_m \; p_l)^2 \; p_r \; p_l \; (4p_l^2 + 8p_l^3 + 16p_l^4) \\
W(3) &= (p_m \; p_l)^3 \; p_r \; p_l \; (8p_l^3 + 16p_l^4 + 32p_l^5) \\
W(4) &= (p_m \; p_l)^4 \; p_r \; p_l \; (2p_l^3 + 4p_l^4 + 48p_l^5 + 16p_l^6) \\
W(5) &= (p_m \; p_l)^5 \; p_r \; p_l \; (24p_l^5 + 32p_l^6) \\
W(6) &= (p_m \; p_l)^6 \; p_r \; p_l \; (4p_l^5 + 24p_l^6) \\
W(7) &= (p_m \; p_l)^7 \; p_r \; p_l \; 8p_l^6 \\
W(8) &= (p_m \; p_l)^8 \; p_r \; p_l \; p_l^6
\end{aligned} \tag{3.14}$$

Substituting $W(1),...,W(N)$ into (3.10) and then multiplying by N yields N_r. N_m is obtained similarly by interchanging p_r and p_m.

IV. The Extra Stage Interconnection Network

The analysis of the non-redundant network was simplified by the independence between the two inputs to any switch, enabling the straightforward calculation of the joint probabilities in (3.2). The incorporation of redundancy into the multi-stage interconnection network, resulting in two (or more) paths connecting any given processor-memory pair, introduces dependency among the links. Equation (3.3) is no longer valid in the general case and a different analysis is required, depending on the network's topology.

As an example for an interconnection network with redundancy we analyze in this section the Extra Stage Cube Network (ESC) [1] which includes $k+1 = logN+1$ stages and is depicted in Fig. 2. The analysis of this network is further complicated by the existence of multiplexers and demultiplexers at the input and output stages, respectively. The purpose of these circuits is to avoid the disconnection of a fault-free processor (or a fault-free memory) upon the failure of a single link. When calculating the bandwidth of the network we have therefore, to distinguish between the first and last stages on one hand and the internal stages on the other hand. All internal stages will be analyzed in one way while the first stage (stage k) and the last stage (stage 0) require a different treatment.

To calculate the state probability $P\{X^{(i)} = 1\}$ for an internal stage we must utilize the joint probabilities $P\{(X^{(i+1)}, Y^{(i+1)}) = (u,v)\}; u,v = 0,1$. Since the input links $X^{(i+1)}$ and $Y^{(i+1)}$ are dependent (there is at least one processor which may send its memory requests through either one of them), their joint probability has to be calculated from the joint probabilities of the four links at level $(i+2)$ through which all incoming requests pass. For example, to calculate the joint probability of output links 0 and 1 of stage 1 in Fig. 2 we need the joint probabilities of the output links 0, 1, 2 and 3 of stage 2. These probabilities might in turn, require the knowledge of the joint probabilities of eight links (and so on for larger ESC networks) making the analysis mathematically intractable.

However, each processor connected to the ESC has only two alternative paths to any given memory and therefore, no more than two links out of every four leading to two switches are dependent at any stage. For example, output links 0 and 1 of stage 2 in Fig. 2 are dependent since processor 0 (and 1)

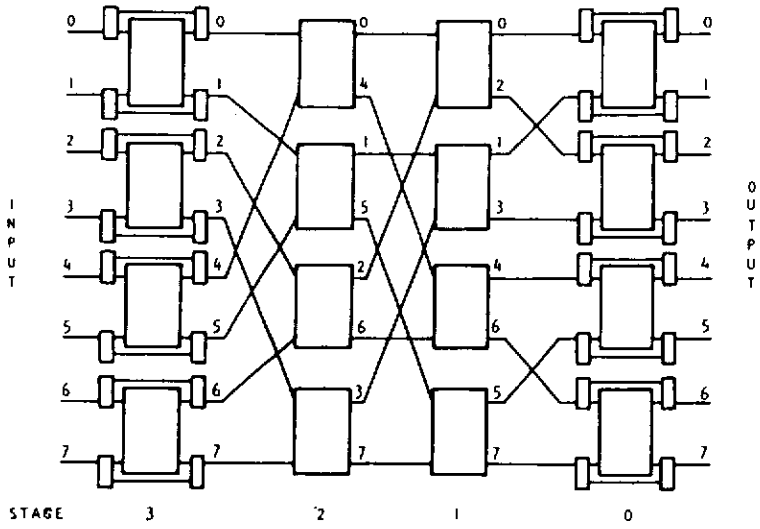

Figure 2: An 8 × 8 Extra Stage Cube network (ESC).

can send requests to memory 0 through either one of them. Similarly, links 2 and 3 are dependent. The pair 0,1 is however, independent of the pair 2,3.

In general, for the internal stages in the ESC network, we have

$$P\{(X^{(i)}, Y^{(i)}) = (u, v)\}$$

$$= \sum_{(s_0,\ldots,s_3)=(0000)}^{(s_0,\ldots,s_3)=(1111)} P\{(X^{(i+1)}, Y^{(i+1)}, Z^{(i+1)}, W^{(i+1)}) = (s_0, s_1, s_2, s_3)\}$$

$$\times P\{(X^{(i)}, Y^{(i)}) = (u, v) \;/\; (X^{(i+1)}, Y^{(i+1)}, Z^{(i+1)}, W^{(i+1)}) = (s_0, s_1, s_2, s_3)\}$$

$$= \sum_{(s_0,\ldots,s_3)=(0000)}^{(s_0,\ldots,s_3)=(1111)} P\{(X^{(i+1)}, Y^{(i+1)}) = (s_0, s_1)\} \times P\{(Z^{(i+1)}, W^{(i+1)}) = (s_2, s_3)\}$$

$$\times P\{X^{(i)} = u/(X^{(i+1)}, Z^{(i+1)}) = (s_0, s_2)\} \times P\{Y^{(i)} = v/(Y^{(i+1)}, W^{(i+1)}) = (s_1, s_3)\} \tag{4.1}$$

$$u, v = 0, 1$$

Consequently, only joint probabilities of two links are required and these can be calculated recursively beginning at the last stage (stage 0) down to the first stage (stage k). Once we reach the first stage, the independence between the two processors enables us to calculate the joint probabilities of their states, similarly to (3.3). However, the existence of multiplexers makes the calculation of the joint probabilities of stage k slightly more complicated

than in the non-redundant network. By observing the ESC network we see that the joint probabilities for the two outputs of any switch in the first stage are:

$$\begin{aligned}
P\{(X^{(k)},Y^{(k)}) = (0,0)\} &= (P\{X^{(k+1)}=0\})^2 \\
&+q_l^3\left[2\,P\{X^{(k+1)}=0\}\times P\{X^{(k+1)}=1\}+q_l(P\{X^{(k+1)}=1\})^2\right] \\
P\{(X^{(k)},Y^{(k)}) = (0,1)\} &= (1-q_l^3)\;P\{X^{(k+1)}=1\}\times P\{X^{(k+1)}=0\} \\
&+(2p_lq_l^3+p_l^2q_l^2)\;(P\{X^{(k+1)}=1\})^2 \\
P\{(X^{(k)},Y^{(k)}) = (1,0)\} &= P\{(X^{(k)},Y^{(k)}) = (0,1)\} \\
P\{(X^{(k)},Y^{(k)}) = (1,1)\} &= p_l^2(2-p_l)^2\;(P\{X^{(k+1)}=1\})^2
\end{aligned} \tag{4.2}$$

where $P\{X^{(k+1)}=1\}$ is given by (3.5) and $P\{X^{(k+1)}=0\}=1-P\{X^{(k+1)}=1\}$.

For the last stage (stage 0) which includes demultiplexers we use the following transition probabilities:

$$\begin{aligned}
&P\{X^{(0)}=1\;/\;(X^{(1)},Y^{(1)})=(0,0)\}=0 \\
&P\{X^{(0)}=1\;/\;(X^{(1)},Y^{(1)})=(0,1)\}=\tfrac{1}{2}\,p_l \\
&P\{X^{(0)}=1\;/\;(X^{(1)},Y^{(1)})=(1,0)\}=\tfrac{1}{2}\,(1-q_l^2) \\
&P\{X^{(0)}=1\;/\;(X^{(1)},Y^{(1)})=(1,1)\}=\tfrac{5}{4}p_l-\tfrac{1}{2}\,p_l^2
\end{aligned} \tag{4.3}$$

The bandwidth is then calculated from

$$BW = N\times P\{X^{(0)}=1\}\times p_m\;p_l \tag{4.4}$$

In what follows we calculate the two measures for network connectivity. The measure C can be expressed as the product of the number of processor-memory pairs (i.e., N^2) times the probability that at least one fault-free path between a given processor-memory pair exists. Each processor-memory pair in the ESC network is connected by two disjoint paths (except for both ends), hence

$$\begin{aligned}
&P\{\textit{At least one path is fault}-\textit{free}\} = P\{\textit{First path is fault}-\textit{free}\} \\
&+P\{\textit{Second path is fault}-\textit{free}\}-P\{\textit{Both paths are fault}-\textit{free}\} \quad (4.5)\\
&=p_r(1-q_l^2)p_l^4(1-q_l^2)p_m+p_rp_l^6p_m-p_rp_l^{10}p_m=p_rp_mp_l^6(5-4p_l+p_l^2-p_l^4)
\end{aligned}$$

Multiplying by N^2 yields C.

N_r and N_m are calculated following the same steps as in Section 3. Define Φ_r as the probability that a given processor (say processor 0) is connected to

at least one memory, and E_j as the event in which the j-th path emanating from processor 0 is fault-free. Recalling that each processor is connected to all memories through $2N$ paths, equations (3.9) (3.10) become,

$$\Phi_r = P\{E_1 \cup E_2 \cup ... \cup E_{2N}\} = \sum_{i=1}^{2N} (-1)^{i-1} W(i) \tag{4.6}$$

where $W(i)$ is defined in (3.11). To obtain the probability that a given subset of paths $\{j_1, ..., j_i\}$ is fault-free, note that in the ESC network each path has $(k+2)$ links, hence the number of distinct links in a subset of paths can be expressed as $d_0 + d_1 + ... + d_{k+1}$ where $d_{k+1} = 1$ and d_0 is the number of memories that the paths in the subset lead to. Given d and d_o,

$$P(E_{j_1} \cap ... \cap E_{j_i}) = p_r \; p_m^{d_0} \; p_l^{d-d_0-1} \; (1 - q_l^2)^{d_0+1} \tag{4.7}$$

This equation differs from (3.12) because of the existence of multiplexers and demultiplexers in the ESC network. Denote by S_{i,d,d_0} the number of subsets of size i which consist of exactly d distinct links, out of which d_0 are at level 0, then

$$W(i) = p_r \sum_{d,d_0} S_{i,d,d_0} \; p_m^{d_0} \; p_l^{d-d_0-1} \; (1 - q_l^2)^{d_0+1} \tag{4.8}$$

Using combinatorial arguments which are omitted here, we obtain

$$S_{i,d,d_0} = \binom{2}{d_k} 2^{d_1+...+d_{k-1}+2d_k-i} \prod_{n=1}^{k} \binom{d_n}{d_{n-1}-d_n} \times \frac{weight(d_0/d_1, ..., d_k)}{\sum_\delta weight(\delta/d_1, ..., d_k)} \tag{4.9}$$

where $\quad weight(\delta/d_1, ..., d_k) = \sum_{x_1,x_2} \binom{d_1 - d_2}{x_2} \binom{2d_2 - d_1}{x_1}$

$$\times \binom{d_1 - d_2 - x_2}{2d_1 - i - x_1 - 2x_2} 2^{2d_1-i-x_1-2x_2} \binom{x_2}{2d_2 - \delta - x_1} \left(\frac{1}{2}\right)^{x_2} \tag{4.10}$$

Substituting $W(1), ..., W(2N)$ into (4.6) and then multiplying by N yields N_r. N_m is obtained similarly by interchanging p_r and p_m.

V. Numerical Results

In this section we present some numerical comparisons between the two previously analyzed networks. The bandwidth of the two systems of size

16×16 has been calculated as a function of p_a (the probability of request), for three different sets of values of the probabilities p_l, p_r and p_m. The results are depicted in Fig. 3. A very important conclusion that can be drawn from this comparison is that both networks have exactly the same bandwidth if there are no faulty elements (case (*i*) in Fig. 3). The additional path between every pair of processor-memory that the ESC network provides, does not increase its bandwidth over that of a non-redundant network, due to the sharing of the redundant paths by other processor-memory pairs.

The situation is different when faulty elements are present in the system (cases (*ii*) and (*iii*) in Fig. 3). The ESC network shows a smaller reduction in bandwidth in the presence of faulty elements. Here, the redundant paths in the ESC network reduce the effect of faulty links on the bandwidth. And, as is evident from Fig. 3, the advantage of the ESC network over the non-redundant one increases as the probability p_l of a fault-free link decreases.

Fig. 4 depicts the average number of fault-free connected processors N_r as a function of p_l for the two systems (for three cases similar to those in Fig. 3). This figure shows that the ESC network is less sensitive to link faults than the non-redundant one when N_r is employed as a measure for system connectivity. We have also tried to separate the effect of the extra stage from the effect of the additional multiplexers and demultiplexers. When the latter were removed, the ESC network showed no advantage over the non-redundant one and both systems produced the same N_r curves.

The other measure for system connectivity, i.e., C - the average number of fault-free connected processor-memory pairs is depicted in Fig. 5. As in Fig. 4, this connectivity measure is shown as a function of p_l for the same three cases. Here again we can see the advantage of the network with redundancy over the non-redundant one. Combining Figures 4 and 5 we can conclude that not only is the number of connected processors N_r larger in the ESC network than in the non-redundant system, but in addition, each fault-free processor in the ESC network is connected (on the average) to a larger number of fault-free memories.

VI. Conclusions

The performance of two multi-processor systems with a multi-stage interconnection network in the presence of faulty elements has been analyzed in this paper. The first is a non-redundant network and the second is the Extra Stage Cube network which was selected as an example for a network

with redundancy. The bandwidth and connectivity of the multi-processing system have been suggested as measures and used to analyze and compare the performance of these two systems.

The approach to performance analysis which was introduced in this paper, can be applied to other schemes for incorporating redundancy into multistage networks. Such an analysis will allow a more accurate comparison of the performance of these architectures in the presence of faulty elements. It can also suggest ways to develop new fault-tolerant architectures.

VII. References

[1] G. B. Adams and H. J. Siegel, "The Extra Stage Cube: A Fault-Tolerant Interconnection Network for Supersystems," *IEEE Trans. on Computers,* Vol. C-31, May 1982, pp. 443-454.

[2] G. B. Adams, D. P. Agrawal and H. J. Siegel, "A Survey and Comparison of Fault-Tolerant Multistage Interconnection Networks," *Computer,* Vol. 20, June 1987, pp. 14-27.

[3] M. Jeng and H. J. Siegel, "A Fault-Tolerant Multistage Interconnection Network for Multiprocessor Systems using Dynamic Redundancy," *Proc. of the 1986 Symp. on Distributed Computing Systems,* pp. 70-77.

[4] I. Koren and D.K. Pradhan, "Modeling the Effect of Redundancy on Yield and Performance of VLSI Systems," *IEEE Trans. on Computers,* Vol. C-36, March 1987, pp.344-355.

[5] V. P. Kumar and S. M. Reddy, "Augmented Shuffle-Exchange Multistage Interconnection Networks," *Computer,* Vol. 20, June 1987, pp. 30-40.

[6] K. Padmanabhan and D. H. Lawrie, "A Class of Redundant Path Multistage Interconnection Networks," *IEEE Trans. on Computers,* Vol. C-32, Dec. 1983, pp. 1099-1108.

[7] J. H. Patel, "Performance of Processor-Memory Interconnection for Multiprocessors," *IEEE Trans. on Computers,* Vol. C-30, Oct. 1981, pp. 771-780.

[8] N. Tzeng, P. Yew and C. Zhu, "A Fault-Tolerant Scheme for Multistage Interconnection Networks," *Proc. of the 12-th Annual Symp. on Comp. Arch.,* June 1985, pp. 368-375.

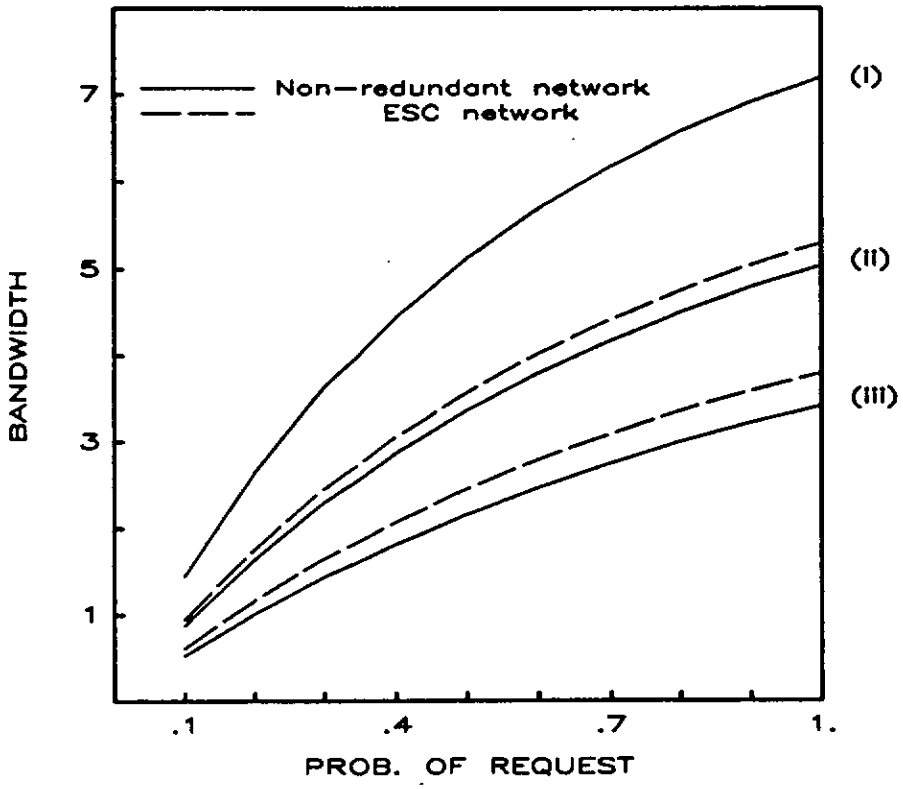

Figure 3: The bandwidth of two 16×16 networks as a function of p_a for (i) $p_l = p_r = p_m = 1$, (ii) $p_l = 0.95$, $p_r = 0.85$, $p_m = 0.9$ and (iii) $p_l = 0.9$, $p_r = 0.75$, $p_m = 0.8$.

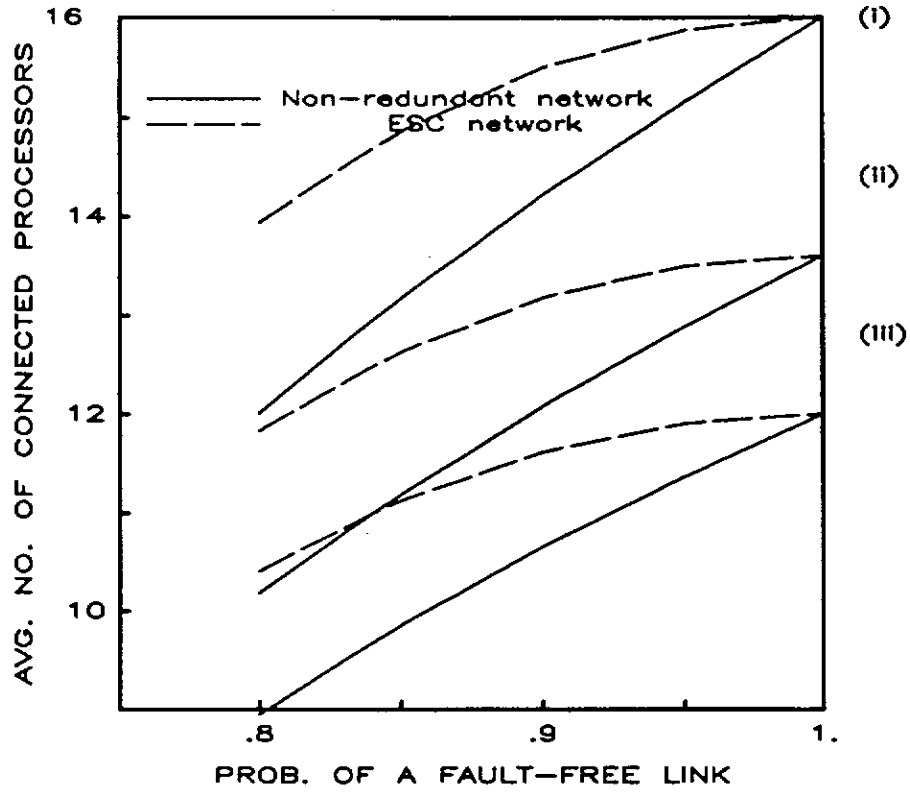

Figure 4: The average number of connected fault-free processors N_r for two 16×16 networks as a function of p_l for (i) $p_r = p_m = 1$, (ii) $p_r = 0.85$, $p_m = 0.9$ and (iii) $p_r = 0.75$, $p_m = 0.8$.

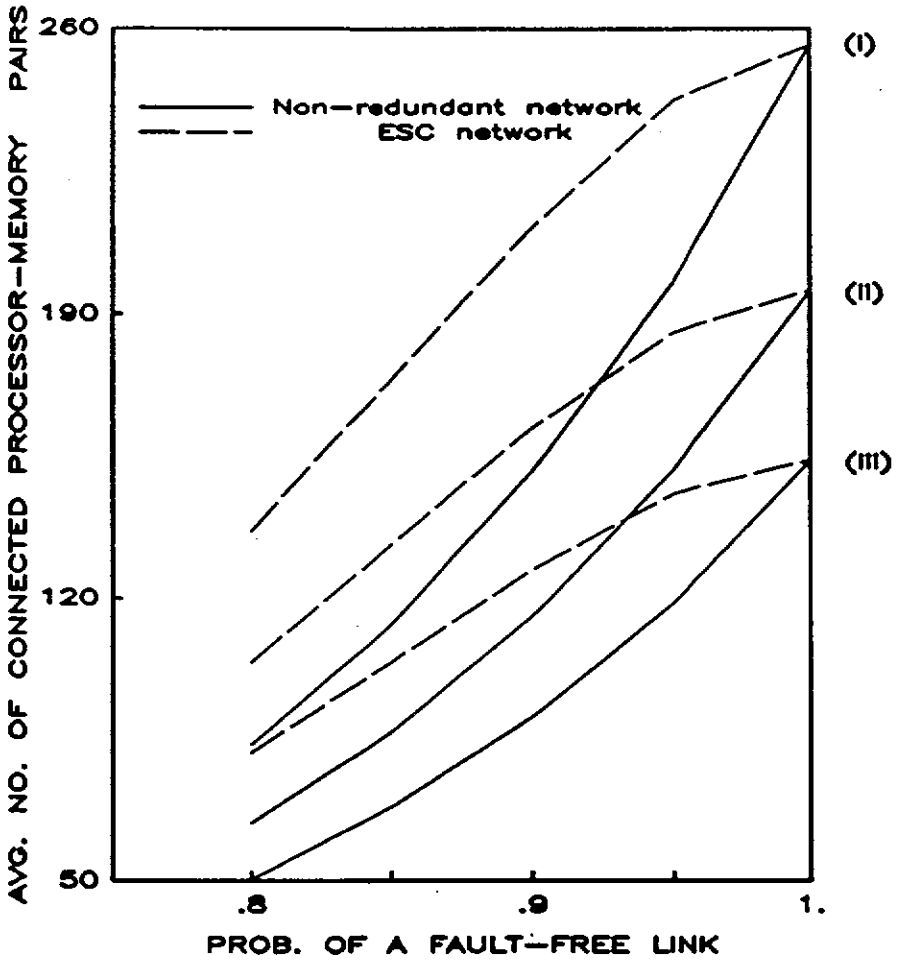

Figure 5: The average number of connected fault-free processor-memory pairs for two 16×16 networks as a function of p_l for (i) $p_r = p_m = 1$, (ii) $p_r = 0.85$, $p_m = 0.9$ and (iii) $p_r = 0.75$, $p_m = 0.8$.