Seedling Project

Final Report

COMPARATIVE STUDY OF MULTICOMPUTER SYSTEMS

By

Dharma P. Agrawal
and
Virendra K. Janakiram

CENTER FOR COMMUNICATIONS
AND SIGNAL PROCESSING

North Carolina State University

Raleigh, NC 27650

CCSP-TR-84/1
I. Introduction

1.1. Purpose of Project

The main purpose of this project is to examine and to determine the relative merits of some of the important interconnection structures that could be used in a multicomputer environment. To this end, various factors that characterize the networks need to be identified so that the system performance could be quantified. This would, in turn, help in selecting an appropriate network configuration that could effectively satisfy given specifications.

1.2. Relevance to CCSP Goals

Many signal and image processing applications require handling of a large volume of data and a very sophisticated decision making process may have to be employed. The complexity of computation has reached such an extent that the solution of such problems on a simple uniprocessor system has become very inefficient and time consuming. In addition, advances in LSI and VLSI technology has led to relatively cheaper microprocessors and microcomputers, and the co-ordinated use of multiple microcomputers has become a lively research topic. An efficient utilization of such a system calls for partitioning of the algorithms such that they could be run in parallel and independent of one another. The signal and image processing algorithms form an important class of applications that are naturally amenable to implementations in multicomputer systems. But, to insure an efficient implementation it is imperative to investigate which particular configuration of multicomputers will be able to provide a reasonable performance for a given set of applications. With this objective in mind, several multicomputer systems have
been examined and their performance has been compared with respect to some important parameters. SIMD type multiprocessor systems have been excluded in this study as they are suitable for a very limited class of applications.

II. Background: Brief Tutorial and Previous Work

2.1 Introduction

A major consideration in the design of a multicomputer system is the selection of an interconnection network. Such a system will consist of many nodes, or individual computers, connected together, functioning in unison. Put simply the problem is--given the numerous computers, how to connect them to one another in the most cost-effective manner. Considerable research effort has been expended in devising new topologies; and the literature [3-6, 9,10] is replete with the results of these endeavours.

As ever new networks are being proposed, it becomes important to be able to evaluate these topologies quickly and accurately. Some of the relevant questions which must be answered are: Through how many intermediate computers must a communication path between two computers be routed, on an average? Is there a simple algorithm which may be used to perform this routing? If an intermediate computer fails, can an alternative path be found? Can the network be expanded without drastic alteration of the entire scheme?

We will attempt to answer these questions for some of the interconnection networks described in the literature. Interconnection networks can be divided into two broad categories: Link structures and Bus structures. The former comprises those networks which require that each computer be connected directly to certain other, designated ones. In bus structures, a group of computers share a common bus; each computer may belong to more than one group. Communication proceeds from bus to bus through the appropriate computers.
2.2 Parameters

We will discuss briefly some of the considerations in the evaluation of interconnection networks from the standpoint of their suitability for use in supercomputers. It should be emphasised that improving one parameter might adversely affect some other parameters: what is sought is an optimization of the network.

2.2.a. One of the more important evaluative measures of an interconnection network is the average distance. This is the distance messages must travel, on an average, in the network. It is advantageous to make this as short as possible. The average distance (in links) is defined as [4]:

\[ \text{AvgDist} = \frac{\sum_{d=1}^{r} d N_d}{(N-1)} \]  

(1)

where N is the number of computers at a distance d links away, r is the diameter (maximum of the minimum distance between any two pairs of nodes) and N is the total number of computers.

However, a network that has a low average distance may require an unreasonable number of communication ports for each computer. In order to distinguish these cases, a normalized average distance is defined [4] for link based structures:

\[ \text{NormAvgDist(link)} = \text{AvgDist} \times \text{Ports/comp} \]  

(2)

where Ports/comp is the number of communication ports required of each computer.

In the case of bus structures, the distance d is the number of buses a message has to cross en route to its destination. Also, the number of computers tied to a bus is of importance as several computers on a single bus may create bottlenecks due to bus contentions. To account for this, we define
the normalized average distance for bus structures as the average distance weighted by the number of computers that may have access to a single bus.

\[
\text{NormAvgDist}(\text{bus}) = \text{AvgDist} \times \text{Ports/bus}
\] (3)

2.2.b. The total number of communication links in a network of given size is another useful measure. Clearly, among two networks, the one that has fewer connecting links is the more desirable, all else being equal.

2.2.c.1 When a message is to be routed from one computer to another, the route it must take is obtained from the routing algorithm. It is desirable that the routing algorithm be simple and not require a complete knowledge of the entire network [5]. In particular, it would be convenient if, by merely having the destination address, it is possible to obtain the exact—and preferably the shortest—sequence of computers the message must traverse.

2.2.c.2 If one of the computers along this route were to be faulty, then a breakdown in communication would result, and it is possible that the entire system crashes. To preclude such a possibility, networks must be fault tolerant. Fault tolerant networks have at least one redundant path between any two computers; and these redundant paths are used in the case of a computer fault.

2.2.d. Any large system must be capable of expansion in such a way that it causes a minimal disruption of the existing set-up. Clearly, a network which requires a complete rebuilding, with fresh demands on the number of communication ports of individual computers, every time extra computers are added, is less preferable to one which can be extended in a natural way, without major upheaval of the entire system.
III. Results

3.A. Work Completed

3.A.1. Link Structures

In this section we analyse some link-oriented structures. The issues mentioned in the previous section are dealt with in seriatim.

3.A.1.1. The Cube Connected Cycles (CCC) Network

This network, proposed by Preparata [10], connects $2^k$ computers ($k$ is integer) in such a way that groups of $2^r$ ($r$ is the smallest integer such that $r + 2^r \geq k$) are interconnected so as to form a $(k-r)$-dimension cube.

Each computer has a $k$-bit address which is expressed as a pair $(l,p)$ of integers, $l$ having $(k-4)$ bits and $p$, $r$ bits. There are three ports, called F, B and L (for Forward, Backward and Lateral) provided on each computer, and the interconnection rule is:

- $F(l,p)$ is connected to $B(1, (p+1) \mod 2^r)$
- $B(1,p)$ is connected to $F(1, (p-1) \mod 2^r)$
- $L(1,p)$ is connected to $L(1, (1+ 2^p, p)$

where $= 1-2 \text{bit}_p(1)$.

An example of such a structure for $k=5$ is shown in Fig. 1.

a) The average distance for the CCC is obtained as the product of the average distance of the sub-group of 2 processors (which form a ring) and the main $(k-r)$ cube network. The number of ports in each computer is three, so the normalized distance is simply the average distance times three.
b) The total number of communication links is at most \((3/2)N\) \([10]\).

c) Wittie \([12]\) gives a simple algorithm to route messages between computers. Even when a node is faulty, an alternative path may be found with ease.

d) Because of the cube structure employed, expansion is not easy. Not only must the expansion be in powers of two, but the system must be completely restructured.

3.A.1.2 The Alpha Network

This is a generalized Hypercube structure \([4]\). Unlike the hypercube, which needs the number of nodes to be of the form \(N = W ** D\), the alpha network is valid for all non-prime values of \(N\). However, we consider only the special case of \(N = W ** D\), where \(W, D\) are integers. The alpha network is constructed in the following manner: Let \(m_1, m_2, \ldots, m_D\) be chosen such that \(m_i\) is integer and \(\prod_{i=1}^{D} m_i = N\). Then each node can be expressed in a mixed radix form as an \(r\)-tuple \((x_D, x_{D-1}, \ldots, x_1)\), which forms the address of the node. Connectors are made from each node to every node whose address differs in any one coordinate. An example of an alpha structure is shown in Fig. 2.

\[\text{a) The average distance in a network is given by:}\]

\[\text{AvgDist(alpha)} = D*(W-1)*W ** (N-1) / (N-1)\]  \(\text{(4)}\)

The number of ports on each processor is given by:

\[\text{Ports(alpha)} = D*(W-1)\]  \(\text{(5)}\)

b) The total number of links is:

\[\text{Links(alpha)} = N * \text{Ports} / 2\]  \(\text{(6)}\)

c) A simple routing algorithm is given in \([3]\).
Because of the several redundant paths that exist, this network is highly fault tolerant.

d) Since this network is a generalized cube network, expansion is not easy as the number of ports is dependent on network size. Unlike cube networks, however, any non-prime value of $N$ can be accommodated.

3.A.1.3 The Hypertree Network

The Hypertree network [5] is basically a binary tree network, which, by the judicious addition of extra edges connecting sibling nodes, has been made to have both, a smaller average distance, and a measure of fault tolerance. These new edges are chosen to be $n$-cube connections, i.e., they link nodes which have (binary) addresses that differ in only one bit. Each processor has four ports—one from the parent, two to the children and one to the sibling.

As mentioned above, the linking of nodes is done with a view to decrease the distance between them. For example, in Fig. 3a, which shows an instantiation of a 15-node hypertree, the distance between nodes 8 and 15 is six (ignoring sibling links). So the sibling links are chosen to reduce this distance. Fig. 3b is a table given by the authors of [5] showing these distances. An entry in the $i^{th}$ row and $j^{th}$ column of the table gives the distance between those siblings in the $(i+1)^{th}$ row whose addresses differ in the $j^{th}$ bit position. A circled entry represents the maximum for that row. Entries just below a circled entry are not considered because they are effectively reduced to three by the addition of sibling links in the previous level. Entries two rows below a circled entry are reduced to five, and so on. With this table the sibling links may be chosen in a fairly straightforward manner. For every level $i$, links are made between siblings that differ those bit positions that are given by the $j$ value of the circled entry of the table.
a) The average distance of the hypertree is tedious to calculate using relation 1. The authors of [5] arrive at a formula using a different approach and this yields a more conservative value. Their formula was used here. It should be noted that this network is not symmetric and hence, equation 1 will give different results, depending on the origin node.

b) The total number of communication links can be shown to be \( N^*2*(2^{([\log N]-1)}-1) \).

c) The routing algorithm is given in [5]. Under fault conditions, the algorithmic procedure becomes rather more complex.

d) In common with tree structures, the network is easily extensible, and requires a minimum disruption of the rest of the system.

3.1.4 The Multitree Structure (MTS)

The multitree structure [3] is another tree type structure which uses connections to sibling nodes to reduce the distance between them. An MTS(m:t) consists of m identical Component Trees (CT's) of t levels which have their roots and their leaves circularly connected. A CT of an MTS graph of degree d is a rooted undirected tree, where each non-leaf node has (d-1) children except the root, which has (d-2) children, and every leaf node is of the same depth.

An MTS (m:t), (m>=3, t>1) graph of degree d has the following properties:

i) there are m identical CT's of depth (t-1).

ii) The roots of m CT's are interconnected to form a ring.

iii) For each level, (t-1), each node is connected to other level (t-1) nodes, and there is at least one cycle containing all the level (t-1) nodes.

An example of an MTS(4:3) is shown in Fig. 4.
a) There is no closed-form formula for obtaining the average distance for an MTS. Arden and Lee [3] have given a table for a number of values of \( n \) and these have been used in the plots of Figs. 7 and 8. The maximum number of ports on each computer remains constant (four).

b) No routing algorithm has been given by the authors. The network has redundant paths, and is hence tolerant of single faults.

c) Although this is a tree structure, it is not easily extensible in small increments.

3. A. 2 Bus Structures

We now analyse two representative bus structures with regard to the criteria described earlier.

3. A. 2. 1 The Spanning Bus Hypercube

The spanning bus hypercube connection is similar to the mesh connection [10]. There are \( N \) computers connected on several buses. Each computer is connected to \( D \) buses which span each of the \( D \) dimensions of the hypercube space. Nodes which have all their coordinates—except the \( i \)th—-the same are connected to the \( i \)th bus. Fig. 5 shows a 3 spanning bus hypercube.

a) The average distance for the spanning bus hypercube is given by relation 4. The reason for this lies in the similarity of the addressing schemes between these networks.

b) The number of buses used is [11]: \( D \ W \ ^* \ (D-1) \).

c) Wittie [12] gives the routing algorithm for this network. A single fault in a bus can be tolerated by such networks.

d) Spanning bus hypercube structures may be expanded by increasing \( D \) or \( W \). Increasing \( W \) has the advantage of not requiring fresh ports in each computer.
3.A.2.2 Beta Networks

The beta structure [4] is of the same topology as the alpha structure. However, a link in the alpha structure is substituted for a node (computer) in the beta. The node of an alpha network is a bus in a beta structure. Fig. 6 is an example of a beta network.

a) There exists no closed-form relation for directly computing the average distance of such a network. The plots of Figs. 7 and 8 were obtained by manipulation of the adjacency matrix.

b) The number of buses is given by $W^D$. The number of nodes is given by:

$$N = W^D (W-1)/2$$

(7)

c) The routing procedure is simple and similar to that of the spanning bus hypercube. Beta networks are also fault tolerant.

d) An expansion of the beta network does require re-routing of the network, but the number of ports demanded of each computer remains fixed at two.

3.A.3 Concluding Remarks

Figs. 7 and 8 show plots of the average distance and normalized average distance vs. the number of computers for these networks. Table I summarizes the points discussed above. It should be noted that the formulae given therein for the average distances are very approximate and for comparison purposes only. Among link structures alpha networks have the smallest average distance. The next best structure is the MTS, followed by the CCC structure. The MTS has the best normalized average distance. But the routing presents a significant hurdle to its performance. The CCC is the next best; and together with its simple routing algorithm and its fault tolerance, looks attractive indeed.
In the case of bus structures, the beta network possesses smaller average distance, but its normalized average distance is larger and also increases at a faster rate than the spanning bus hypercube.

3.B Problems Encountered

It was observed that it is impossible to find a closed form solution for the average distance and programs have to be written for different structures. Moreover, the performance cannot be predicted by a single parameter. The quality factor is a function of various parameters and it is not possible to define the weights for each one of the characteristics.

IV. Conclusions

4.A. Significance of Results to CCSP Interests

As indicated earlier, the ALPHA and BETA structures do provide a better value of the average distance when the number of processors are the same in all the structures. This is particularly important in those image and signal processing applications in which a large volume of data has been moved around and a smaller value of average distance guarantees that the communication delay will be minimal. This, in turn, would provide a significant reduction in communication overhead and hence lead to a significant improvement in the speed and turnaround time. In addition, the ALPHA and BETA structures are highly fault tolerant [4] as they provide several alternate and disjoint paths between the source and destination nodes.

4.B. Potential Future Work

4.B.1 Continuation of Present Activity

The ALPHA and BETA structures possess some very important properties. But there are some shortcomings too: For example, in ALPHA structure, a large
number of input/output ports are required while the several processors connected on a single bus of the BETA structure introduce a heavy traffic load on the bus and hence, leads to the bus contention problem. Hence, it would be interesting to look into techniques that would enable us to retain advantages of both the schemes. One possible solution seems to be to look into a hierarchically structured interconnection network in which each node of the network is itself a smaller network of computers. The characteristics of such networks ought to be examined as they might provide a better performance of a lower interconnection cost. Intuitively, this appears to be a valid argument, as in a multicomputer environment, most algorithms are expected to exhibit the program locality and the "neighbouring" computers are expected to have the most interactions and, hence, more frequent transfer of data will take place among the computers logically close to one another.

This selection of an appropriate interconnection network is only one facet in the design of a multicomputer system. If a large volume of data is to be transferred between various computers, then there will be a need to design a special purpose I/O controller with adequate buffers such that the processing unit of the node need not be overburdened with I/O handling. Various other factors include the queueing delays at the nodes, the design of control units, the operating system and the fault-tolerance. Models are an excellent tool in determining the expected behaviour of the system, but, because of the complexity of the multicomputer system, any tractable analytic mode will be oversimplified, and therefore, may be significantly inaccurate. It seems that the simulation studies, though quite expensive, might be more helpful.
4.B.2 Possible Related Topics

Several multicomputer structures have been compared with respect to the average distance, the normalized distance, the routing simplicity and the fault-tolerant capabilities. Of course, the true performance of such networks can be evaluated by mapping certain typical algorithms onto multicomputer structures and studying their performance. An on-going project [i] enabled us to look into the suitability of various multicomputer structures and the "ALPHA" structure seems to provide a better performance for dynamic scene analysis algorithm [2]. There is also a scope for investigation into various other issues involved, such as: (i) developing a technique to perform automatic modification in the algorithmic steps such that the performance could be improved, (ii) looking into ways to perform appropriate allocation in systems with non-homogeneous resources and (iii) determining the optimal size and types of computing resources for a given problem. Each one of these areas is of crucial importance although they require a considerable amount of effort. Algorithms will perform well on certain types of networks and poorly on others. Such simulation studies will be tedious and time-consuming, but will be of use in the final selection of a particular supercomputer network topology among a few others. A current research project that deals with such studies [1] will be described, and its results presented, in a forthcoming conference. The method we have outlined, on the other hand, can be used for a preliminary screening from among the vast number of networks that has been proposed.
V. REFERENCES


Table 1

<table>
<thead>
<tr>
<th></th>
<th>Average Distance</th>
<th>Ports/Node (link structures) or ports on a bus (bus structures) (P)</th>
<th>Normalized Average Distance (L)</th>
<th>Message Density</th>
<th>Number of Links (Buses)</th>
<th>Fault Tolerance in terms of No. of link failure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully Connected Network</td>
<td>1</td>
<td>N-1</td>
<td>N</td>
<td>N</td>
<td>(N^2-N)</td>
<td>N-2</td>
</tr>
<tr>
<td>Cube Connected Cycles</td>
<td>0.8 log&lt;sub&gt;2&lt;/sub&gt;N -2.4</td>
<td>3</td>
<td>2.4 log&lt;sub&gt;2&lt;/sub&gt;N -7.2</td>
<td>5D/4</td>
<td>3N/2</td>
<td>2</td>
</tr>
<tr>
<td>Alpha</td>
<td>k log&lt;sub&gt;2&lt;/sub&gt;N + 0.2 D(W-1)</td>
<td>k = 0.3 to 0.5</td>
<td>D(W-1)L</td>
<td>2L/D(W-1)</td>
<td>ND/2</td>
<td>D(W-1)-1</td>
</tr>
<tr>
<td>Hypertree</td>
<td>1.1 log&lt;sub&gt;2&lt;/sub&gt;N -0.7</td>
<td>4</td>
<td>4.4 log&lt;sub&gt;2&lt;/sub&gt;N -2.8</td>
<td>L/2</td>
<td>2(N-1)</td>
<td>3</td>
</tr>
<tr>
<td>MTS</td>
<td>log&lt;sub&gt;2&lt;/sub&gt;N-2</td>
<td>3</td>
<td>3 log&lt;sub&gt;2&lt;/sub&gt;N-6</td>
<td>2 log&lt;sub&gt;2&lt;/sub&gt;N-4</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Single Global Bus</td>
<td>1</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>1</td>
<td>N-1</td>
</tr>
<tr>
<td>Spanning Bus Hypercube</td>
<td>k log&lt;sub&gt;2&lt;/sub&gt;N + 0.2 W k = 0.3 to 0.5</td>
<td>W</td>
<td>W k log&lt;sub&gt;2&lt;/sub&gt;N + 0.2W k = 0.3 to 0.5</td>
<td>W</td>
<td>DWD-1</td>
<td>W-1</td>
</tr>
<tr>
<td>Beta</td>
<td>k log&lt;sub&gt;2&lt;/sub&gt;N + 0.25 D(W-1)</td>
<td>k = 0.3 to 0.5</td>
<td>D(W-1)L</td>
<td>L</td>
<td>W^D</td>
<td>D(W-1)-1</td>
</tr>
</tbody>
</table>