Topological Comparison of Perfect Shuffle and Hypercube

Ran Ginosar[1] and David Egozi[1]

Novel topological measures for static multiprocessor interconnection networks, *disconnectivity*, *looseness*, and *cost-effectiveness*, are developed. These and other measures are employed for a comparative analysis of such networks. The goal of this analysis is to predict network effectiveness, without resorting to execution benchmark techniques. In particular, we compare the **hypercube** and **perfect-shuffle-nearest-neighbor** networks, and show that they are the best candidates for multiprocessor interconnections. We specifically find that the hypercube is capable of somewhat better performance than perfect-shuffle-nearest-neighbor, but the latter is significantly more cost-effective.

1. INTRODUCTION

The great progress in parallel processing is originated by the needs and demands for increasing computing power on-one-hand, and the technological capabilities to achieve this goal on the other. Many approaches to parallel processing have been proposed and experimented with, such as vector processors and multiprocessors.[1,2]

In this paper, we concentrate on MIMD multiprocessors interconnected through a statically configured communication network, where nodes are identical to each other. Such a computer was named an *Ensemble Machine* by Seitz.[3] One such machine is the Ultracomputer, as proposed by Schwartz,[4] which employs the Perfect Shuffle Exchange network.[5]

---

[1] Department of Electrical Engineering, Technion-Israel Institute of Technology, Haifa 32000, Israel.

37

Another is the Cosmic cube, based on the Hypercube interconnection network, proposed and implemented by Seitz.[6] Ensemble machines in the context of VLSI are surveyed in Ref. 7. We analyze the influence of the interconnection network on the performance of the ensemble machine. Among many different interconnection networks for ensemble machines, we concentrate on the *Perfect Shuffle Nearest Neighbor* (PSNN) and the *Hypercube* (HC) networks.

We utilize two complementing approaches for comparative evaluation of static multiprocessor interconnection networks. A collection of "typical" algorithms are analyzed on candidate machines, in an attempt to find out on which machine those algorithms perform better. This study is documented in another paper (Ref. 8). The approach presented here examines the topological characteristics of each interconnection network and tries to predict performance out of these features. While the former approach is operational, the latter analysis focuses on inherent qualities of the networks. Hence, the algorithmic analysis is in essence limited to those algorithms which are employed for the study, while the topological analysis is intended as a more powerful and more general performance prediction tool. It is important to note that the results presented in this paper and the results of the algorithmic analysis[8] agree with each other.

From the algorithmic analysis[8] we have realized that there are two important communication primitive operations, out of which all other communications may be constructed. One is the forwarding of a single message, either to a single receiver or broadcast to all nodes. The second is permutation, that is, the simultaneous forwarding of N different messages from N different senders to N different receivers. In this paper, we address these two capabilities as well as the network cost, within a unified framework.

In Section 2, we define the scope of machines analyzed, and in Section 3 the communication model is presented. The analyzed networks are described in Section 4, and the topological measures are discussed in Section 5. The networks are compared in Section 6, and special attention to hypercubes and perfect shuffles is paid in Section 7. Detailed application of the topological measures to the networks comprises the Appendix.

2. ENSEMBLE MACHINES

The focus of this work is on MIMD, loosely coupled multiprocessors. Each processor of a loosely coupled system may access directly only its own local memory, and together they form a *node* in the network. The node may contain additional elements, such as various coprocessors. Communications with other nodes are achieved by exchanging messages

between the nodes. Each node may also contain a special processor, called a *communication controller*, designed to assume the task of handling message exchange with other nodes.[6,9,10]

Examples of loosely coupled multiprocessors are Cm*,[11] Ultracomputer,[4] X-Tree,[12,13] the Cosmic Cube[6] and Intel iPSC.[14] Other machines, such as the BBN Butterfly,[15] the IBM RP3,[16] TRAC,[17] Cedar,[18] MP/C,[19] SM3[20] possess characteristics of both tightly and loosely coupled organizations. However, they all employ some dynamic switching mechanism in order to achieve this duality.

All the processors of a *homogeneous multiprocessor* are functionally and computationally identical to each other. A loosely coupled system which is also homogeneous is named an *Ensemble Machine* (EM).[3] Most of the architectures mentioned here are ensembles, and therefore we choose to concentrate on EMs in this paper.

3. THE COMMUNICATION MODEL OF THE ENSEMBLE

In order to compare different static interconnection networks for ensemble machines, a general model is required which emphasizes the communication aspects and is insensitive to irrelevant differences. We adopt an abstract model of communications, which is instrumental in both the present topological analysis, and the algorithmic analysis presented in Ref. 8. The model satisfies the following assumptions:

A) The multiprocessor interconnection network consists of point-to-point *links* between the computational nodes. The basic operation in a link is a *communication step*, during which some unit of information is transferred between two nodes. The network is homogeneous, in the sense that a communication step takes exactly c time units everywhere on the nerwork. During the communication step all nonconflicting information transfers, which occur over mutually exclusive links, may take place simultaneously.

B) Each node may either perform one processing (computation) step or one communication step at any point in time. A computation step is, for instance, one ALU operation on two or less operands. The length of a processing step, constant over all operations and over all nodes, is p time units.

C) Each point-to-point link in the network is bidirectional, but at any point in time it passes information in only one direction. That is, an *exchange* operation takes two steps.

D) Two nodes are connected by at most one link.

E) Suppose i, j, k are three distinct nodes, where j and k are each connected by a direct link to i. Node i may send data to node j and at the same time receive data from node k.

F) The cost of the ensemble consists of two components, one reflecting the cost of the processors and the other accounting for the cost of the network:

$$Cost(\text{EM}) = (number\ of\ processors) \times (cost\ of\ a\ processor)$$

$$+ (number\ of\ links) \times (cost\ of\ a\ link)$$

$$= N \cdot cost(P) + M \cdot cost(link)$$

where N is the number of processors and M is the number of links. In networks where all nodes have the same degree of connectivity C, the second term can be expressed by N and C:

$$Cost(EM) = N \cdot cost(P) + \frac{NC}{2} \cdot cost(link)$$

$$= N \cdot cost(P) + N \cdot cost(network)/node$$

where we define

$$cost(network)/node = \tfrac{1}{2} cost(link) \cdot C = K \cdot C$$

Thus C, the connectivity degree of the node, is proportional to the cost of the network per node. Therefore we define the network cost function as the node connectivity degree C.

G) The link bandwidth (capacity, in number of bits per second) is constant over all links, and is limited by the capability of the processor to send and receive data, as explained below. Figure 1 depicts the communication model of the single node. The communication controller (CC) is unable to store data. The bandwidth of each link is
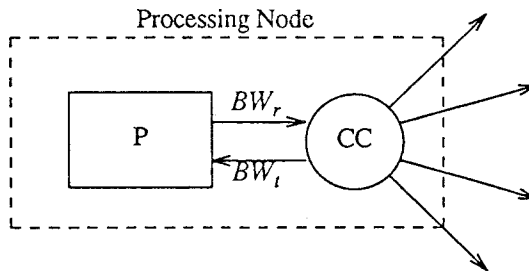

Fig. 1. Node communication model.

constant and equal to the bandwidth of the send and receive ports of the processor, and the CC merely switches the total transmitting bandwidth offered by the processor to a single link, and the total receiving bandwidth offered by the processor to another single link. We assume that $BW_r = BW_t = BW$, and that receiving and transmitting can be performed simultaneously on two different links, as in point E. The total bandwidth of the processing node is thus twice that of each link.

Commenting on the last point, in this *node-limited* model, communication bandwidth is limited by the bandwidth of the nodes in the network. Actually, three other models were also considered and rejected:

1. Another type of node-limited model imposes the same limit on the bandwidth of each node; i.e., twice that of a link, but allows directional flexibility. That is, the two ports of each node may be tied together into a single, twice as fast, link. This is an interesting model with practical implications, but it turns the analysis slightly more complex.

2. Another node-limited model divides the node bandwidth among all its incident links. That is, the more links there are, the lower each link's bandwidth is going to be. While the model adopted here is suitable for CCs which are capable of switching full capacity of ports and links, this alternative model takes into consideration the fact that more links may mean less wires per link. When the bandwidth of each wire (e.g., chip I/O pin) is constant and limited, less wires per link result in less bandwidth per link.

3. The last model is link limited: Each link is assigned a fixed bandwidth, but the processor's bandwidth is always larger than the sum of all bandwidths of all its incident links. This model is appropriate for ensembles employing slow links, which we prefer to ignore in this paper.

We prefer the model described above because it reflects most closely real implementations (or what we believe they should be...). It is also insensitive to irrelevant issues such as the specific charater and computational power of the processor, the width of the data links (bit serial, byte serial, etc.), and various characteristics of the communication (circuit versus packet switching, store-and-forward versus cut-through[21] etc.).

4. INTERCONNECTION NETWORKS

Multiprocessor interconnection networks and their effect on performance are the main issue of this paper. A thorough survey of interconnec-

tion networks is presented in Ref. 22. While some multiprocessors employ reconfigurable interconnections, we analyze only static networks in this work. Static networks require neither expensive switches nor complex switch control, and are based on simpler point-to-point links. They are employed in ensemble machines such as Schwartz's Ultracomputer[4] and the various hypercubes.[3,6] We apply a topological taxonomy to classify the following static networks which are most commonly proposed for multiprocessors: Completely connected network (CCN), one- and two-directional rings, nearest neighbor (NN), star, tree, chordal ring, hypercube (HC) and the perfect shuffle nearest neighbor (PSNN). Admittedly, additional networks have also been proposed and they too deserve analysis within the present framework. They include the cube-connected-cycles network,[23] multitree structured graphs,[24] the Hypertree,[25] the Lens network,[26] generalized chordal rings,[27] and meshes of trees and trees of meshes.[28] Preliminary study shows that none of them surpasses either hypercube or PSNN.

4.1. Examples of Static Networks

In the following a brief survey is provided of the completely connected network, one- and two-directional rings, nearest neighbor (NN), star, tree, chordal ring, hypercube (HC) and the perfect shuffle nearest neighbor (PSNN). The topological characteristics of these networks are discussed in Section 6. Examples of all these networks are presented in Fig. 2.

4.1.1. Completely Connected Networks

In a completely connected network (CCN) every node is directly connected to every other node. $N(N-1)/2$ links are required for an N node CCN. The number of links grows quadratically with N, and linearly per each node, while the complexity of routing and control remains minimal.

4.1.2. Rings

Rings may employ either uni- or bi-directional links. In the latter case, the ring is also called *one-dimensional nearest neighbor* (NN_1) network, and the connectivity degree of each node is 2.

4.1.3. k-*Coordinate* n-*Dimensional Nearest Neighbor Networks*

Each node in NN_n is connected to two neighbors in each one of the n dimensions. In other words, each node participates in n different NN_1 sub-networks, one per dimension, and each NN_1 sub-network consists of k nodes. Consequently, the degree of each node grows as $2n$.
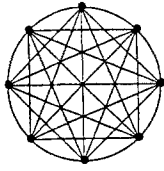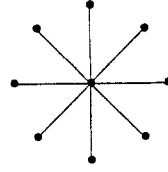
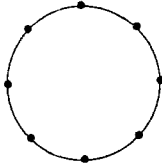Fig. 2a. Completely connected network (CCN).

Fig. 2e. The star network.

Fig. 2b. One-directional cyclic shift
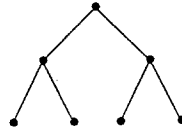network ··· ring.
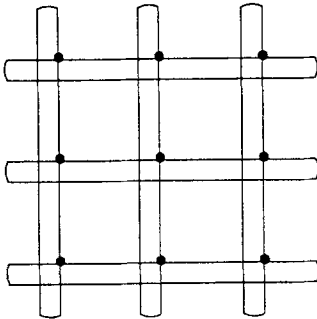
Fig. 2f. The tree network.
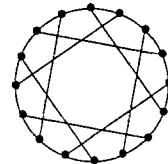
Fig. 2c. Two-dimensional nearest neighbor.

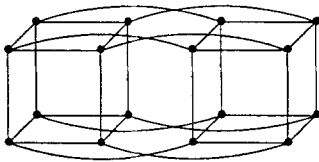Fig. 2g. Chordal ring, single chord.

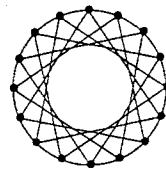Fig. 2d. The hypercube network.
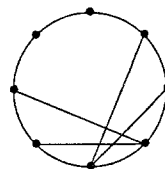
Fig. 2h. Chordal ring, dual chords.

Fig. 2i. The PSNN network.

NN_n networks are also called *Hypertorus* networks in Ref. 7. When the k nodes are connected by only $k-1$ links, such that instead of NN_1 rings the single dimensional subnetworks are only linear arrays, we call it a *Hypermesh*.

4.1.4. Hypercube

The *Hyprecube* (HC) is a special case of NN_n (or, rather, the hypermesh), where the number of nodes in each dimension is $k=2$. Hence it is also called *binary N-cube*, or *Boolean N-cube*. For n dimensions, the number of nodes is $N = 2^n$. Note that node i is directly connected to each node j where the numbers i, j differ in exactly one digit of their binary representation. Each such connection is a single point-to-point link, unlike a general NN_n, where each 1-dimensional subnetwork is a NN_1 ring. In other words, the number of links per node grows as $n = \log_2 N$.

4.1.5. Star

The central node of the star network is connected to each of the other nodes with a single link. A shared bus system is the dynamic parallel of the star network. The connectivity is $N-1$ at the central node, and 1 for all other ones, or $C \cong 2$ on the average.

4.1.6. Tree

Trees as multiprocessor interconnection networks were investigated by Browning.[29] The tree network is planar (similar to the star and the ring), routing is simple, the degree of each node is bounded, and there are no loops in the network, which helps eliminate routing deadlocks.

4.1.7. Chordal Rings

Chordal rings[30] have either one chord per node (see Feng[22]), or two (see Lang[31]). On a single-chord ring, node i is linked to node $i+a$ and node $i+1$ is linked to node $(i+1)-a$, in addition to the ring connections. The chord distance a is fixed over the network. In a dual chord ring network, every node i is linked to nodes $i+a$ and $i-a$, as well as to its two ring neighbors $i \pm 1$. Optimally, the distance bridged by the chord is $\sqrt{N}$.

4.1.8. Perfect Shuffle Nearest Neighbor

The *perfect shuffle nearest neighbor* (PSNN) network combines NN_1 and perfect shuffle interconnections.[5] The perfect shuffle permutation σ is defined as follows:

$$\sigma(i) = \begin{cases} 2i & 0 \leqslant i < \dfrac{N}{2} \\[2ex] 2i + 1 - N & \dfrac{N}{2} \leqslant i < N \end{cases}$$

$$\sigma^{-1}(i) = \begin{cases} \dfrac{i}{2} & i \quad even \\[2ex] \dfrac{i + N - 1}{2} & i \quad odd \end{cases}$$

Node P_i is linked to $P_{\sigma(i)}$ and to $P_{\sigma^{-1}(i)}$. Considering the binary representation of i, $i = (b_{n-1}, b_{n-2}, ..., b_0)$, then $\sigma(i) = (b_{n-2}, b_{n-3}, ..., b_0, b_{n-1})$ and $\sigma^{-1}(i) = (b_0, b_{n-1}, ..., b_1)$. Hence, $\sigma^n = (\sigma^{-1})^n = I$ (*identity permutation*).

The σ interconnection graph induces a partition of the network into unconnected components, e.g., $\{0\}$, $\{1, 2, 4\}$, $\{3, 5, 6\}$, $\{7\}$ for $N = 8$. Each such component is a ring of nodes whose numbers are cyclic shifts of each other, and are called *necklaces*.[28] For that reason, NN_1 interconnections are added in the PSNN network. A known variation is the *Perfect Shuffle Exchange* (PSE) network, where the ε interconnection replaces NN_1,

$$\varepsilon(b_{n-1}, b_{n-2}, ..., b_0) = (b_{n-1}, b_{n-2}, ..., \overline{b_0})$$

Other variations of PSNN are *shuffle-shift-reverse* and *shuffle-shift-transpose*, as defined in Ref. 28. All these networks are very similar in their characteristics, and we prefer to represent them all by PSNN.

5. TOPOLOGICAL MEASURES

For the purpose of comparative network evaluation, we define topological measures of performance, cost, and cost effectiveness. The measures are independent of applications and algorithms. Rather, they reflect only the network topology. Hockney and Jesshope[1] define maximal distance d, fan out time F, and the degree of connectivity C. The remaining parameters, disconnectivity D, looseness L, and cost-effectiveness LC, constitute the original contribution of this work.

The measures are computed for each of the networks, as summarized in Table I. The detailed computation is included in the Appendix. The analysis shows that the HC and PSNN networks perform better than most others. The exception is the completely connected network, which is not a practical solution for large systems due to its quadratically growing cost, and is brought here merely as an upper bound on performance. Consequently, PSNN and HC make the focal point of this work.

Table I. Asymptotic Topological Measures of Static Networks of N Nodes
(Exact Measures are Given in Table A)

Network	Distance d	Fanout F	Disconnectivity D	Looseness L	Connectivity C	Cost-Effectiveness LC
CCN	1	$\log N$	$1/N$	$\log N$	N	$N \log N$
NN_1	N	N	N	N	1	N
NN_2	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	1	$\sqrt{N}$
NN_n	$N^{1/n} \log N$	$N^{1/n} \log N$	$N^{1/n}$	$N^{1/n} \log N$	$\log N$	$N^{1/n} \log^2 N$
Star	1	N	N	N	1	N
Tree	$\log N$	$\log N$	N	N	1	N
Chordal Ring	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	1	$\sqrt{N}$
Hypercube	$\log N$	$\log N$	1	$\log N$	$\log N$	$\log^2 N$
PSNN	$\log N$	$\log N$	$\log N$	$\log N$	1	$\log N$

5.1. Maximal Distance *d*

The maximal distance (also called *diameter*) d is the maximal number of links $d(i, j)$ that have to be traversed from any source node i in order to reach any object node j, $i \neq j$, along the shortest path:

$$d \equiv \max_{i, j} \{d(i, j)\}$$

The shorter the distance, the less number of communication steps required for any message in the network. Therefore, we choose to call d a figure of *nonmerit* of the network.

5.2. Fan-out Time *F*

The fan-out time F is the number of steps required to broadcast a message from any node i to all other nodes. The message is spread over the links of some spanning tree of the network, whose root is i. Each intermediate node on the fanout tree duplicates the incoming message onto each one of the outgoing links. If $b(i)$ is the minimal number of steps required to broadcast from node i, then:

$$F \equiv \max_{i} \{b(i)\}$$

Note that F accounts for communication steps, although the time to perform message duplication may be included as well. Note further that according to the communication model presented in Section 3, a node which needs to send out the message over multiple links does so serially, one link after another. It is self evident that F is also a figure of "nonmerit." Furthermore, since $b(i) \geqslant \max_{j} \{d(i, j)\}$, $F \geqslant d$.

The inverse operation is data concentration, where some node i receives the result of an associative operator applied to data elements from all other nodes (e.g. sum). The data elements are collected along the links of a spanning tree whose root is i. Each intermediate node on the tree applies the associative operation to all incoming data elements and to its own element, and then forwards the interim result towards the root. Short of the computation time, the time of concentration is equal to F.

The general case of contention is not modeled by fanout. For instance, contention may occur when $n - 1$ (or less) messages, generated by $n - 1$ nodes, are all directed at the nth node. When two such messages, say M_1 and M_2, meet at some node j along their passage towards their destination, they contend for the services of node j. This is due to the model of Section 3, which allows a node to receive and process only one message at a time. Node j eventually receives both M_1 and M_2, in contrast with the

data concentration case where only one result is transmitted. Thus, general contention resolution may take longer time than F. This also applies to implementation of shared variables on Ensemble Machine, where a concurrent write into a shared variable may result in contending messages.

On the other hand, fanout does apply to contention resolution on combining networks. In a combining network, the two contending messages are replaced by a single, "combined" message, which is similar to data concentration, and hence can be modeled by fan out. Eventhough combining networks were originally proposed for Shared Memory Multiprocessors,[32] they may also be applicable to Ensemble Machines, or the mixed-mode machines such as RP3.[16]

In some CRCW (concurrent reads, concurrent writes) PRAM models,[33] multiple concurrent write attempts to the same variable result in exactly one of them succeeding, and all the others disposed of. With such a model we can assume that the time to resolve a contention of two messages at some node is equal to the time it takes to forward just one message, as if the other one did not exist. With this modification, the fanout time in this case is the same as the maximum distance, d.

Of course, there are other cases where multiple messages exist in the network, each going to its own destination. These cases are modeled by permutations, rather than fanout, as described later.

5.3. Disconnectivity D

The parameters d and F defined earlier predict the complexity of sending a single message, by a single node, to either a single receiver or to all. Our experience[8] shows that the two most important patterns of communications, as required by common algorithms, are the exchange of single messages **and** the simultaneous permutation of N messages, one from each node. The following definition of *disconnectivity* has been initiated by the need to predict the capability of the network to perform permutations.

The definition of disconnectivity is based on the *minimum bisection width* **e** of the network, as defined by Thompson.[34] Let e be the size of the minimal edge cut set which, when removed from the network G, divides it into two approximately equal disconnected components G_1 with $\lfloor N/2 \rfloor$ nodes, and G_2 with $\lceil N/2 \rceil$ nodes. Then

$$D \equiv \frac{N}{e}$$

D measures the worst-case number of communication steps required for permutations. Suppose a permutation π exchanges all $N/2$ elements of G_1 with all $N/2$ elements of G_2. Then N elements must be transferred over e links, requiring at least N/e communication steps.

The computation of the minimum bisection width e is not always a straightforward step. In the Appendix we combine direct analysis with methods developed by Thompson,[34] Leiserson,[35] and Leighton.[28] Specifically, we use Thompson's theorem which states that *the VLSI layout area of a graph with bisection width* B *is at least* $\Omega(B^2)$. We use the inverse result,

The bisection width of a graph with VLSI layout area A *is at most* $O(\sqrt{A})$.

Since some of the networks we analyze have known optimal layouts $\Theta(A)$, we can at least asymptotically estimate their bisection width and, consequently, disconnectivity D.

Since it is clear that performance deteriorates with growing D, disconnectivity is also a "nonmerit" figure.

5.4. Looseness *L*

Thus far we have defined three different measures of performance, d, F, and D. No one of them reflects all aspects of communications; while d and F relate to the time required to forward a single message, D refers to permutations. The definition of *looseness* combines both aspects into a unified measure of communicational performance.

Note that d and F are not mutually exclusive. In particular, $F \geqslant d$. Hence, we ignore d in the following, and observe that F is a reliable substitute for d. F and D, on the other hand, are in a sense "orthogonal," each describing the network under different operations. Furthermore, F and D are exhaustive, in the sense that all the algorithms we have analyzed[8] require either permutations or handling of single messages, or both, and no other types of communications are ever encountered. Consequently we conclude that F and D are necessary and sufficient measures of network communication performance.

Both F and D have been characterized as "nonmerit" figures; i.e., the lower they are the better. A network with low F and low D may be considered "tight," and one with higher F and/or D is more "loose." Therefore we define *looseness L*,

$$L \equiv \alpha F + (1 - \alpha) D$$

Note that the addition of F and D is based on the observation that at any point in time the network performs either single message transfer or permutation, but not both; there is no meaning to doing both simultaneously, since concurrent handling of multiple messages is tantamount to permutation. Thus, $F \times D$ is not an acceptable definition of L. This point is specifically made here because an attempt to define $L = F \times D$ would change the

results of this research in a significant way; in particular, our finding that PSNN is preferable to hypercube depends on this definition. On the other hand, it will be shown below that an allowable alternative definition could be $\sqrt{F^2 + D^2}$.

As long as $0 < \alpha < 1$ in this definition, the asymptotic results do not change. The exact analysis, on the other hand, does depend on the actual value of α. This parameter can be used to associate greater importance to either broadcasts $(\alpha \to 1)$ or permutations $(\alpha \to 0)$. Consequently, we investigate all three cases: $\alpha = 0$, $\alpha = 1$ and $\alpha = 1/2$.

5.5. Degree of Connectivity C

The connectivity degree C is defined as the number of links incident upon each node. C is used as a cost function for networks with the same connectivity over all nodes (e.g., unlike star), as is explained in Section 3. C reflects the network cost per node, and is useful for comparing different networks with the same number of nodes.

5.6. Cost-Effectiveness LC

The product LC of looseness and cost reflects the cost-effectiveness of the network in performing communications. Ordinarily, cost is divided by performance to achieve cost-effectiveness; as explained earlier, looseness may be considered as inversely proportional to performance, and therefore *cost/performance* is proportional to *cost • looseness*.

6. A COMPARATIVE TOPOLOGICAL ANALYSIS OF STATIC NETWORKS

We now apply all the measures defined here to each of the networks described in Section 4. The exact calculations are included in the Appendix, and are summarized there in Table A. To simplify the exposition, we present only asymptotic results in this section. Table I and Figs. 3 and 4 demonstrate the results.

A variety of interesting phenomena may be noted. The d and F columns support our suggestion that F reliably represents d. However, the star network is an exception, due to its serious bottleneck in the central node.

Considering disconnectivity D, note that CCN demonstrate an interesting and unique feature of decreasing D with increase in size. Also note that D is the first measure to distinguish the hypercube from PSNN. Further examination of D versus d and F discovers the disparity between
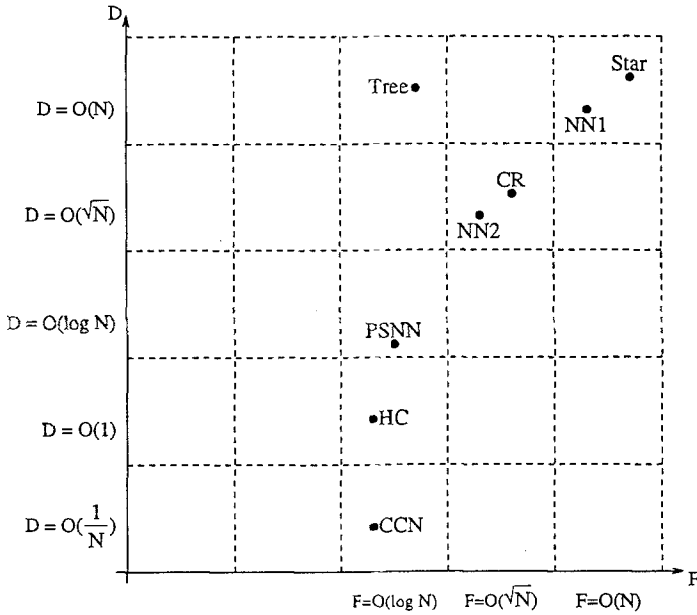
Fig. 3.   Qualitative performance of static networks.

them. Trees, for instance, have low d and F but high D, while chordal rings are low in D but high in d and F. This provides empirical justification for the introduction of the new measure D.

When L encompases both aspects of communications $(0 < \alpha < 1)$, the best performing networks, as measured by L, are CCN, HC and PSNN. When we choose $\alpha = 1$ $(L = F)$, the Tree also joins this class. However, when network cost is also considered, CCN is of course prohibitively expensive, as can be seen from the connectivity column. Another feature of the cost measure is that like D it also distinguishes PSNN from the hypercube.

The ultimate measure, when cost is not overlooked, is the cost-effectiveness LC, as seen in the last column. For instance, the best performing networks (L-wise), CCN, HC and PSNN, are split into three classes; CCN is clearly the least cost-effective network, and HC is found inferior to PSNN.

To emphasize these results, a qualitative graphic representation is adopted. Two graphs are shown in Figs. 3 and 4, exhibiting performance and cost-effectiveness, respectively. The axes of both graphs are marked with asymptotic functional orders of magnitude and are demonstrative
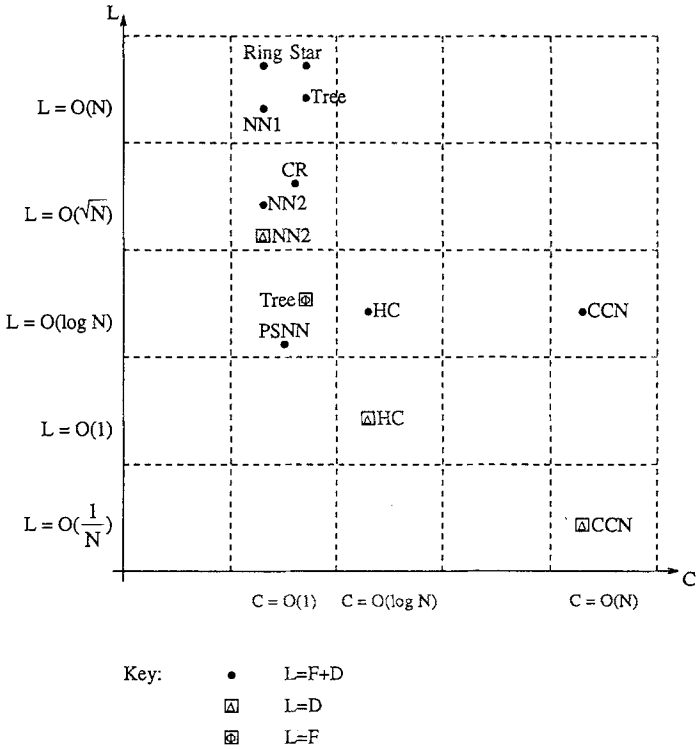
Fig. 4. Qualitative cost-effectiveness of static networks.

only. Variations within each asymptotic class reflect possible minor changes between networks.

The performance graph plots D versus F. Note that the distance of each network from the origin reflects the looseness L of that network; actually, it was this observation that led us to perceive L in the first place, and to define it originally as $L = \sqrt{F^2 + D^2}$. We observe that CCN promises best performance, and that the predicted performance of HC is better than that of PSNN.

The cost-effectiveness graph shows L versus C. The most cost-effective networks must be closest to the origin. We observe again that for $0 < \alpha < 1$ PSNN is the only member of the most cost-effective class (since Fig. 4 is an symptotic representation, $L = F + D$ for all $0 < \alpha < 1$). When $\alpha = 1$ $(L = F)$, the Tree joins in with PSNN in the most cost-effective class. This agrees with the fact that the tree is competitive on broadcasts but inferior on permutations.
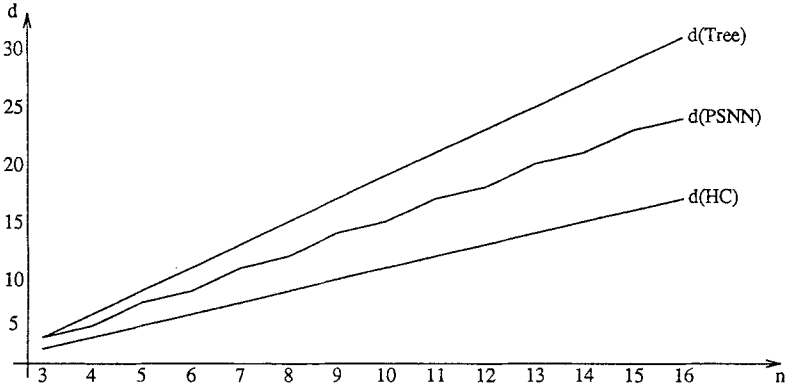
Fig. 5. Distance versus network size.

7. HYPERCUBES VERSUS PERFECT SHUFFLES

We now analyze the topological parameters of the hypercube and PSNN in more detail. In order to be able to make accurate comparison of these very similar networks, the topological measures are computed exactly, rather than asymptotically, and are presented in Table II. As noted in the previous section, if the application at hand is broadcast-intensive (i.e. employs a negligible amount of permutations), the tree network is comparable in some aspects to the PSNN, and is consequently also included in the table. The complete computation is included in the Appendix.

With respect to distance, all three networks belong to the class of $d = O(\log N)$, which is second only to $d = O(1)$. The difference lies in the slope of d as a function of N. Similar, and slightly stronger, effect is observed on the F versus N graph. See Figs. 5 and 6.
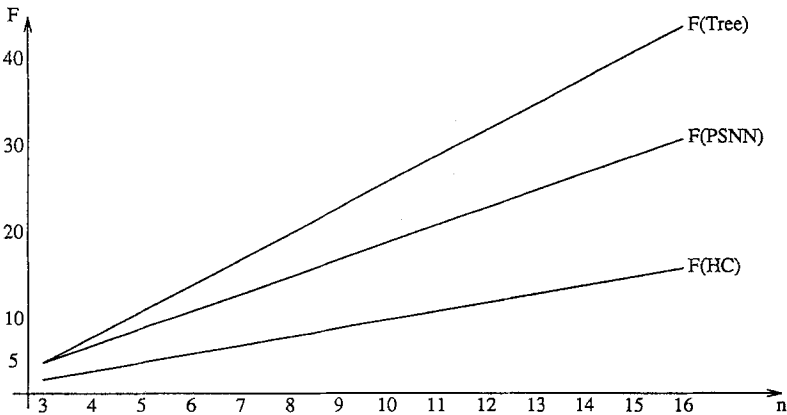
Fig. 6. Fanout versus network size.

Table II. Topological Measures of Hypercube, PSNN, and Tree of $N = 2^n$ Nodes

Network	Distance d	Fanout F	Disconnectivity D	Looseness $L = \frac{1}{2}F + \frac{1}{2}D$	Connectivity C	Cost-Effectiveness LC
Hypercube	n	n	2	$\frac{1}{2}n + 1$	n	$\frac{1}{2}n^2 + n$
PSNN	$n + \left\lceil \frac{n}{2} \right\rceil - 1$	$2n - 1$	$\approx \frac{n}{2}$	$\frac{5}{4}n - \frac{1}{2}$	4	$5n - 2$
Bin. Tree	$\approx 2n - 2$	$\approx 3n - 4$	N	$\frac{3}{2}n - 2 + \frac{1}{2}N$	$\approx 2\frac{1}{2}$	$\approx \frac{15}{4}n - 5 + \frac{5}{4}N$

Fig. 7. Disconnectivity versus network size.
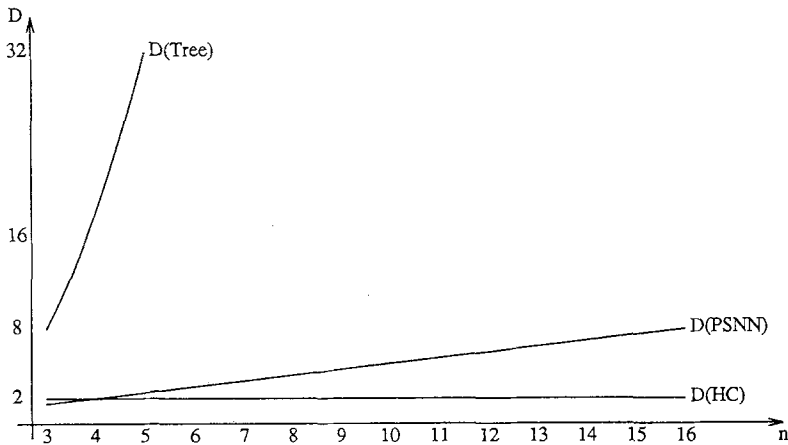
The different disconnectivity measures are shown in Fig. 7. As mentioned earlier, the tree demonstrates a prohibitive level.

While HC and PSNN are quite similar performance-wise, the cost difference is more significant since they belong to different classes. The cost of an HC node grows logarithmically with N, while the cost of a PSNN node (as that of a tree) is constant for any network size. As long as the number of processors is smaller than 16 the cost of HC is lower than PSNN. For $N > 16$ the cost of HC is higher, and the difference increases with N. See Fig. 8.

Note that this break-even point generates similar break-even points for the distance d and fanout F, when considered weighted by cost. On the other hand, disconnectivity D induces no such break-even point (for $n = 3, 4\ D\ (\text{PSNN}) = 2$; $D\ (\text{PSNN}) = n/2$ holds only for $n \geqslant 4$). HC is
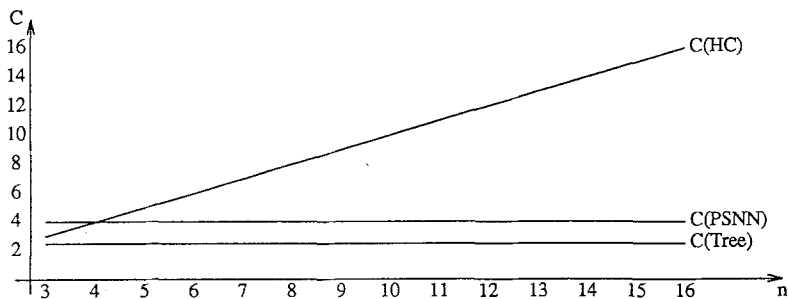
Fig. 8. Network cost per node versus network size.

always preferred since it has constant D, whereas the disconnectivity of PSNN grows logarithmically with N.

The combined effect of F (as well as d) and D is reflected in L, the looseness of the network, which is asymptotically similar for both PSNN and HC. When considered with cost in mind, however, the networks differ. See Fig. 9. The $L*C$ graph refers to $\alpha = 1/2$. Fig. 9 also contains the graphs for $\alpha = 0$ ($D*C$ instead of $L*C$), and $alpha = 1$ ($F*C$ instead of $L*C$). It also shows the graphs for the tree, which is more cost-effective than both PSNN and HC for applications with no permutations. The break-even

Fig. 9. Cost-effectiveness versus network size.

point of LC for the hypercube and PSNN lies between $N = 128$ and 256. Hence we conclude that for communications which include 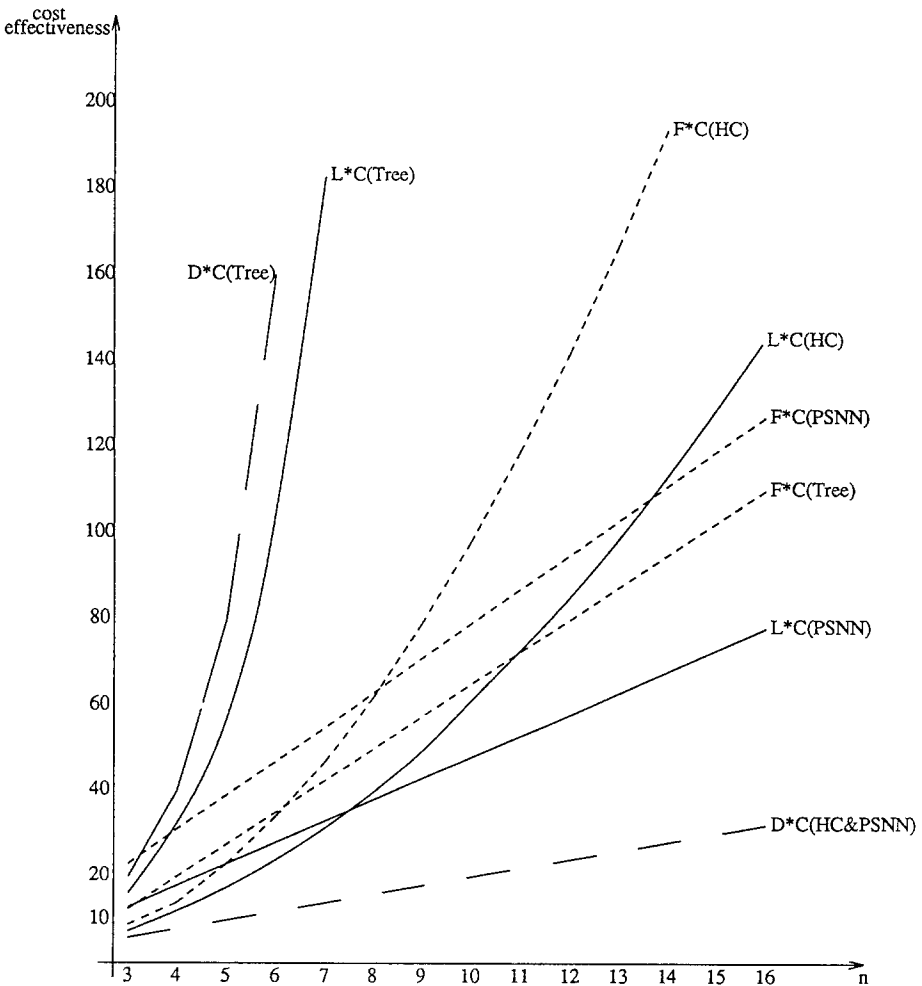permutations, broadcast (or concentration) and possibly transfers of single messages, PSNN is preferable to HC for $N > 128$. This also applies to broadcast-intensive communication (F^*C), where the break-even point is $N = 256$. For communications consisting mainly of permutations, the two networks are comparable in cost-effectiveness: HC permutes in constant time at a logarithmic cost per node, while PSNN permutes in logarithmic time at a constant cost per node.

8. CONCLUSIONS

We have applied topological measures to multiprocessor static inter-connection networks in order to predict their performance. This research complements the study of an algorithmic benchmark experiment, which provided similar findings.[8]

Novel measures have been developed. They include the disconnectivity of the network, which measures the ability to perform permutations; loose-ness, which combines measures of different patterns of communications; and cost-effectiveness of the network.

An abstract communication model of the network has been defined. That model was instrumental in both the topological and algorithmic analyses, and helped to isolate relevant issues.

The new topological measures, as well as some known ones, were applied to a large set of static networks, and provided comprehensive evaluation of those networks. Special attention was directed at the two most promising networks, namely the hypercube and the perfect-shuffle-nearest-neighbor networks. In conclusion we found that while the hyper-cube promises slightly superior performance, the costeffectiveness of a large $(N \geqslant 256$ nodes) perfect-shuffle-nearest-neighbor network is preferred over that of the hypercube, and is actually second to none.

ACKNOWLEDGMENT

APPENDIX: COMPUTING THE TOPOLOGICAL PARAMETERS

A.1 MAXIMAL DISTANCE d

1. CCN

Since every node is connected to every other node, $d = 1$.

2. Ring

In a unidirectional ring the distance between node i and node j is $|i - j| \bmod N$. For $0 \leqslant i, j < N$ the maximal value of $|i - j| \bmod N$ is $N - 1$. Hence $d = N - 1 = 0(N)$.

3. NN₁

In a bidirectional ring there are two paths from node i to node j, $i \neq j$. Let us denote these paths $l_{i,j}^{cw}$, $l_{i,j}^{ccw}$. Then

$$d = \max_{i,j} \left\{ \min(l_{i,j}^{cw}, l_{i,j}^{ccw}) \right\} = \begin{cases} \dfrac{N-1}{2} & N \text{ odd} \\[2mm] \dfrac{N}{2} & N \text{ even} \end{cases}$$

Or, asymptotically, $d = 0(N)$.

4. NNₙ

In the k-coordinate, n-dimensional case the maximal shortest path consists of n path segments in the n dimensions, where in each dimension the segment is of maximal distance. NN_n contains multiple bidirectional rings (NN_1 networks), each having $k = N^{1/n}$ nodes. Hence, the maximal length of each NN_1 segment is $\lfloor k/2 \rfloor$. There are n such segments, hence $d = n \lfloor k/2 \rfloor$. Since $n = \log_k N$, $d = 0(k \log N) = 0(N^{1/n} \log N)$.

5. Star

In a star network the length of the path from one node to another, when neither of them is the central node, is 2. The first step is from the source node to the central node, and the second step is from the central node to the destination node. The central node is exactly one step away from any other node. Hence $d = 2 = 0(1)$.

6. Tree

In a fully balanced tree network with branching ratio (number of children per node) b and k leaf nodes, the maximal distance is $2 \log_b k$.

Proof.   The root has b children at the first level down, b^2 at the second level down, and $k = b^m$ leaves at the last, or mth, level. Hence there $m = \log_b k$ links between the root and each leaf. The longest path starts at one leaf, passes the root and ends at another leaf, and is thus $2 \log_b k$.   ▮

The maximal distance can be also expressed as a function of N and b. For m levels,

$$N = 1 + b + b^2 + \cdots + b^m = \frac{b^{m+1} - 1}{b - 1} \tag{1}$$

$$b^{m+1} = N(b - 1) + 1$$

$$m = \log_b[N(b - 1) + 1] - 1 \tag{2}$$

$$d = 2m = 2 \log_b[N(b - 1) + 1] - 2 \tag{3}$$

Asymptotically, $d = 0(\log N)$.

7. Chordal Ring

Assume the chords are of optimal length $\sqrt{N}$. Arden and Lee[30] show that the maximal length of the optimal path between two nodes is $0(\sqrt{N})$.

8. Hypercube

The distance between two nodes in HC is the number of different bits in the binary representation of the two node indices. Hence the maximal distance in the network is the number of bits in the index, $d = \log_2 N$. We also get this result when we substitute $k = 2$ in the expression for d of NN_n.

9. PSNN

Hockney and Jesshope[1] show that for PSNN $d = n + \lceil n/2 \rceil - 1$, because the most distant node from node i is node $i + j$ where $j = \{10101 \cdots\}$ and the path from i to j consists of $n - 1$ σ-steps and $\lceil n/2 \rceil$ nearest-neighbor steps.

A.2 FANOUT TIME F

1. CCN

Each node in a CCN network is connected to all other nodes, so there are no communication conflicts. Still, in a single communication step a node can transfer information to only one other node, according to the model presented in Section 3. Hence, each communication cycle at most

doubles the number of nodes which have received the information. There-fore the total number of steps required for fanout is $F = \log_2 N$.

2. Ring

In a unidirectional ring the only way to broadcast is by going from node to node in the proper direction. The number of steps required is $F = N - 1 = 0(N)$.

3. NN₁

On a bidirectional ring, the broadcasting node starts sending the message to one direction at step 1, and to the opposite direction at step 2. From then on, the message advances in parallel on both sides, until the whole ring is exhausted. Clearly this process takes $F = \lceil N/2 \rceil = 0(N)$.

4. NNₙ

The source node broadcasts the data in a one-dimensional ring as explained above for NN_1. Subsequently, all nodes of that ring spread the data along the rings of the second dimension, and so on. The fanout time for a ring with $k = N^{1/n}$ nodes has been found to be $\lceil k/2 \rceil$. This is multi-plied by n, the number of dimensions, yielding

$$F = n \left\lceil \frac{k}{2} \right\rceil = 0(N^{1/n} \log N)$$

5. Star

The central node needs $N - 1$ steps to deliver its message to its $N - 1$ neighbors. If the source were another node, it would send its message to the central node at step 1, and the central node would use $N - 2$ steps to send the data to all other nodes. Thus $F = N - 1 = 0(N)$.

6. Tree

In a balanced tree network with branching ratio b and m levels, the longest fanout is incurred by the leaves. The source leaf sends its message to the root, taking m steps. The root, and every nonleaf node below the root, distribute the message to their b children at b steps. There are m steps down to the leaves, so the last leaf receives the message $b \times m$ steps after the root started the broadcast. However, the root can skip that one subtree from which the message originated, so broadcasting from the root to the rest of the tree takes $b \times m - 1$. Adding the initial m steps, we get

$$F = (b + 1) m - 1$$

When we substitute the expression for m derived in Section A.1.6,

$$F = (b+1)\{\log_b[N(b-1)+1]-1\}-1$$

If we consider b constant, $F = 0(\log N)$.

7. Chordal Ring

In a small chordal ring where $N \leqslant 16$ and each node is connected to 4 other nodes, the number of nodes that receive the data is doubled every step. Hence for $N \leqslant 16$, $F = \lceil \log_2 N \rceil$. When $N > 16$, after the first four steps there are nodes which have no more neighbors to send the data to, and the fanout process slows down from the initial exponential rate. Therefore, $F > \log N$. Since each chord bridges over $\sqrt{N}$ nodes in a minimal-distance chordal ring, the network contains a cycle of $\sqrt{N}$ chords, and at most $0(N)$ step are required for broadcast. Lacking any better algorithm, we assume that $F = 0(\sqrt{N})$.

8. Hypercube

Substituting $k = 2$ in the expression of F for NN_n network, we get

$$F = n\lceil k/2 \rceil = \log_2 N$$

In other words, the number of nodes receiving the message doubles at every step. Counting the dimensions $i = 1,..., \log N$, at each step the message is sent over the links of the ith dimension.

9. PSNN

The broadcast algorithm on PSNN consists of a loop including two steps:

loop:

1. *if (fanout not complete)*
   Each node i with the message sends it to its exchange-neighbor $\varepsilon(i)$.

2. *if (fanout not complete)*
   Each node i with the message sends it to its perfect shuffle-neighbor $\sigma(i)$.
   end loop;

The number of 1-steps in the algorithm is n, the number of 2-steps is $n-1$, and therefore the total number of steps is $F = 2n - 1 = 0(\log N)$.[36] Note that this is in contrast with Hockney and Jesshope.[1]

A.3 DISCONNECTIVITY D

1. CCN

In a completely connected network $N-1$ links go out of each node to the other $N-1$ nodes. We bisect the networks into two groups of size $N_1 = N_2 = N/2$. The number of links in the bisecting cut set separating the two groups is therefore

$$e = N_1 \times N_2 = \frac{N^2}{4}$$

$$D = \frac{4}{N} = 0\left(\frac{1}{N}\right)$$

2. Ring

In a unidirectional ring each cut set includes a single directed link from the first group to the second, and a single directed link from the second group to the first one.

$$e = 1$$

$$D = N$$

3. NN_1

In a bidirectional ring each cut set includes two links.

$$e = 2$$

$$D = N/2$$

4. NN_n

The k-coordinate, n-dimensional NN_n hypertorus may be considered as comprising k hyperplanes, each of $n-1$ dimensions and N/k nodes. All the hyperplanes are connected together by N/k rings (NN_1 networks), each ring including k nodes, one node per hyperplane. The edges of the rings constitute the nth dimension of the hypertorus. The minimal bisecting cut set includes the N/k cut sets of the N/k rings, i.e., $2N/k$ edges altogether:

$$e = \frac{2N}{k}$$

$$D = \frac{k}{2} = 0(N^{1/n})$$

5. Star

The star network does not have an edge-separator which produces two connected subnetworks of approximately equal sizes. To evaluate the star

network in performing permutations, we find it more appealing to examine the minimal *node cut set* instead. Clearly that cut set is the central node, so define $e = 1$.

$$e = 1$$
$$D = N/1 = N$$

6. Tree

Similar to the star network, we consider the node cut set consisting of the root, and again,

$$e = 1$$
$$D = N$$

7. Chordal Ring

We consider the case of single-chord rings where the arc bridged by the chord is of length $\sqrt{N}$. Note that $\lceil N/2 \rceil > \sqrt{N}$ for $N > 4$. Suppose the nodes are numbered $0, \ldots, N-1$ around the ring, and that we cut the ring between 0 and $N-1$ and between $N/2$ and $N/2-1$. Then the minimal cut set includes links of three types:

1. The two ring segments, 0-to-$N-1$ and $N/2-1$-to-$N/2$.
2. The chords from $0, 2, 4,\ldots, \sqrt{N}-1$ to $N-\sqrt{N},\ldots, N-1$, counting approximately $\sqrt{N}/2$ chords.
3. A similar group of $\sqrt{N}/2$ chords around the $N/2$ cut.

Altogether there are about $\sqrt{N}+2$ edges making the cut set:

$$e = \sqrt{N} + 2 = 0(\sqrt{N})$$
$$D = \frac{N}{\sqrt{N}+2} = 0(\sqrt{N})$$

The case of doubly-chorded ring is very similar $(e = 2\sqrt{N}+2)$.

8. Hypercube

The disconnectivity of NN_n was shown to be $2N/k$. Here, instead of rings, there are single links in each dimension, and therefore the factor of 2 should be removed. Substituting $k = 2$,

$$e = N/2$$
$$D = 2$$

Table A: Exact Topological Measures

Network	Distance d	Fanout F	Disconnectivity D
CCN	1	$\log N$	$4/N$
NN_1	$\lfloor N/2 \rfloor$	$\lceil N/2 \rceil$	$N/2$
NN_2	$2\lfloor \sqrt{N}/2 \rfloor \approx \sqrt{N}$	$2\lceil \sqrt{N}/2 \rceil \approx \sqrt{N}$	$\sqrt{N}/2$
NN_n	$n\lfloor N^{1/n}/2 \rfloor$	$n\lceil N^{1/n}/2 \rceil$	$N^{1/2}/2$
Star	2	$N-1$	N
Binary Tree	$2\log(N+1)-2$	$3[\log(N+1)-1]-1$	N
Chordal Ring	$O(\sqrt{N})$	$O(\sqrt{N})$	$\approx \dfrac{N}{\sqrt{N}+2}$
Hypercube	$\log N$	$\log N$	2
PSNN	$\log N + \lceil 1/2 \log N \rceil - 1$	$2\log N - 1$	$\dfrac{N}{\dfrac{2N}{\log N}-1}$

Network	Looseness $L = 1/2F + 1/2D$	Connectivity C	Cost-Effectiveness LC
CCN	$1/2 \log N + 2/N$	$N - 1$	$(N-1)(1/2 \log N + 2/N)$
NN_1	$N/2$	2	N
NN_2	$3\sqrt{N}/4$	4	$3\sqrt{N}$
NN_n	$\dfrac{n+1}{4} N^{1/n}$	$2n$	$\dfrac{n(n+1)}{2} N^{1/n}$
Star	$N - 1/2$	N	$N^2 - 1/2N$
Tree	$\dfrac{N}{2} + \dfrac{3}{2}[\log(N+1)-1] - 1/2$	$\approx 2\tfrac{1}{2}$	$\dfrac{5}{4}\{N + 3[\log(N+1)-1] - 1\}$
Chordal Ring	$\approx\sqrt{N}$	3	$\approx 3\sqrt{N}$
Hypercube	$1 + 1/2 \log N$	$\log N$	$\log N + 1/2(\log N)^2$
PSNN	$\dfrac{\dfrac{3}{2} N - \dfrac{N}{\log N} - \log N + 1/2}{\dfrac{2N}{\log N} - 1}$	4	$\dfrac{6N - \dfrac{4N}{\log N} - 4\log N + 2}{\dfrac{2N}{\log N} - 1}$

9. PSNN

It has been shown by Leighton[28] that the VLSI area of the PSE, as well as the PSNN (also called *shuffle-shift* in Ref. 28), networks is $\Theta(N^2/\log^2 N)$. We have shown in Section 5.3 that this implies a minimum bisection width $e = 0(N/\log N)$. Although this result is sufficient for asymptotic estimation of D, we now attempt to arrive at a more accurate result, for the sake of Table II.

When we arrange all the nodes in σ-necklaces, we get mostly necklaces of size $\log N$ except for $0(\sqrt{N})$ degenerate necklaces with less nodes.[28] It has been shown in Ref. 28 that it is possible to arrange the PSE network on a two-dimensional *level-necklace grid* such that the necklaces occupy two adjacent columns each, and each exchange link can be drawn as a horizontal line at some *level*. The two degenerate necklaces of nodes $0, N-1$ contain one node each and should be counted out.

An approximate bisection can be achieved by cutting along a horizontal line between two adjacent levels, such that the two subnetworks are as equal to each other as possible. The cut set consists of two edges per necklace, except the two degenerate ones 0, $N-1$ explained earlier. This cut set consists therefore of $2N/\log N - 2$ links. A more accurate bisection will be achieved by a jug which crosses exactly one level, cutting at most one additional, exchange link. Thus, for the PSE

$$e \leqslant \frac{2N}{\log N} - 1 = 0\left(\frac{N}{\log N}\right)$$

$$D = \frac{N}{e} \geqslant \frac{N}{\dfrac{2N}{\log N} - 1} \geqslant \frac{\log N}{2} = \Omega(\log N)$$

As for the PSNN, the exact minimum bisection width is still an open problem. It is not clear that the PSNN network can be arranged on a necklace-level grid, similar to PSE. For PSNN, we suspect that e may be larger than $0(N/\log N)$, and thus D may turn out to be better (i.e. smaller) than $\Omega(\log N)$, possibly even $D = 0(1)$ as for the hypercube. In any case, note that the asymptotic figure for looseness L remains the same.

REFERENCES

1. R. W. Hockney and C. R. Jesshope, Parallel Computers, Adam Hilger Ltd., Bristol (1980).
2. K. Hwang and F. A. Briggs, Computer Architecture and Parallel Processing, McGraw-Hill Book Co., New York (1984).
3. C. L. Seitz, Ensemble Architectures for VLSI-A Survey and Taxanomy, *Proc. Conf. Adv. Res. in VLSI*, Massachusetts Institute of Technology, pp. 130–135 (1982).

4. J. T. Schwartz, Ultracomputers, *ACM Trans. Prog. Lang. Sys.*, **2**(4): 484–521 (1980).
5. H. S. Stone, Parallel Processing with the Perfect Shuffle, *IEEE Trans. on Computers*, Vol. C-20, No. 2, pp. 153–161 (1971).
6. C. L. Seitz, The Cosmic Cube, *Comm. ACM*, **28**(1): 22–33 (1985).
7. C. L. Seitz, Concurrent VLSI Architectures, *IEEE Trans. on Computers*, **C-33** (12):1247–1265 (1984).
8. R. Ginosar, and D. Egozi, Algorithmic Comparison of Hypercubes and Perfect-Nearest-Neighbor Networks, Technical Report, Dept. of Electrical Engineering, Technion, Haifa, Israel, (1987).
9. C. H. Ng, FIFO Buffering Transceiver: A Communication Chip Set for Multiprocessor Systems, MSc Thesis, Dept. of Computer Science, California Institute of Technology, Pasadena, California (1982).
10. Y. Rimoni, I. Zisman, R. Ginosar, and U. Weiser, Communication Element for the Versatile MultiComputer, *Proc. 15th IEEE Israel Conf.*, pp. 2.1.2 (1987).
11. R. J. Swan, S. H. Fuller, and D. P. Siewiorek, Cm*: A Modular Multimicroprocessor, *Proc. AFIPS NCC*, No. 46, pp. 637–644 (1977).
12. C. H. Sequin, and R. M. Fujimoto, X-Tree and Y-Components, *VLSI Architecture*, (eds.), B. Randell and P. C. Treleaven, pp. 471–480, Prentice-Hall, New Jersey (1983).
13. A. M. Despain, and D. A. Patterson, X-TREE: A Tree Structured Multiprocessor Computer Architecture, *Proc. 5th An. Symp. Computer Architecture*, pp. 144–151 (1978).
14. Intel. Corp., iPSC Documentation (1985).
15. Bolt Beranek and Newman Inc., Butterfly Parallel Processing Computer, Cambridge, Massachusetts (1986).
16. G. F. Pfister, W. C. Brantley, D. A. George, S. L. Harvey, W. J. Kleinfelder, K. P. McAuliffe, E. A. Melton, V. A. Norton, and J. Weiss, The IBM Research Parallel Prototype (RP3): Introduction and Architecture, *Proc. Int'l. Conf. on Parallel Processing*, pp. 764–771 (1985).
17. M. C. Sejnowski, E. T. Upchurch, R. N. Kapur, D. P. S. Charlu, and G. J. Lipovski, An Overview of the Texas Reconfigurable Array Computer (TRAC), *Proc. AFIPS NCC*, No. 49, pp. 631–641 (1980).
18. D. Gajski, D. Kuck, and D. Lawrie, CEDAR: A Large Scale Multiprocessor, *ACM SIGARCH Computer Architecture News*, **11**(1):7–11 (1983).
19. B. W. Arden and R. Ginosar, MP/C: A Multiprocessor/Computer Architecture, *IEEE Trans. on Computers*, **C-31**(5):455–473 (1982).
20. C. K. Baru and S. Y. W. Su, The Architecture of SM3: A Dynamically Partitionable Multicomputer System, *IEEE Trans. on Computers*, **C-35**(9):790–802 (1986).
21. P. Kermani, and L. Kleinrock, Virtual Cut-Through: A New Computer Communication Switching Technique, *Computer Networks*, 3(4):267–286 (1979).
22. T. Y. Feng, A Survey of Interconnection Networks, *IEEE Computer*, pp. 12–27 (December 1981).
23. F. P. Preparata and J. E. Vuillemin, The Cube-connected Cycles: A Versatile Network for Parallel Computation, *Comm. ACM*, **24**(6):300–309 (1981).
24. B. W. Arden and H. Lee, A Regular Network for Multicomputer Systems, *IEEE Trans. on Computers*, **C-31**(1):60–69 (1982).
25. J. R. Goodman and C. H. Sequin, Hypertree: A Multiprocessor Interconnection Topology, *IEEE Trans. on Computers*, **C-30**(12):923–933 (1981).
26. R. A. Finkel and M. H. Solomon, The Lens Interconnection Strategy, *IEEE Trans. on Computers*, **C-30**(12):960–965 (1981).
27. K. W. Doty, New Designs for Dense Processor Interconnection Networks, *IEEE Trans. on Computers*, **C-33**(5):447–450 (1984).

28. F. T. Leighton, Complexity Issues In VLSI, MIT Press, Cambridge, Massachusetts (1983).
29. S. A. Browning, The Tree Machine: A Highly Concurrent Computing Environment, Ph. D. Thesis, Dept. of Computer Science, Californa Institute Technology, Pasadena, California (1980).
30. B. W. Arden and H. Lee, Analysis of Chordal Ring Networks, *IEEE Trans. on Computers*, **C-30**(4):291–295 (1981).
31. C. R. Lang, The Extension of Object Oriented Languages to a Homogeneous Concurrent Architecture," Ph. D. Thesis, California Institute Technology, Pasadena, California (1982).
32. A. Gottlieb, R. Grishman, C. P. Kruskal, K. P. McAuliffe, L. Rudolph, and M. Snir, The NYU Ultracomputer-designing an MIMD shared memory parallel computer, *IEEE Trans. on Computers*, **C-32**(2):175–189 (1983).
33. Y. Shiloach and Vishkin U., Finding the Maximum, Merging and Sorting in a Parallel Computation Model, *J. of Algorithms*, **2**(1):88–102 (1981).
34. C. D. Thompson, A Complexity Theory for VLSI, Ph. D. Thesis, Dept. of Computer Science, Carnegie-Mellon Univ., Pittsburgh, Pennsylvania (1980).
35. C. E. Leiserson, Area-Efficient VLSI Computation, MIT Press, Cambridge, Massachusetts (1983).
36. C. R. Jesshope, private communication (December 1985).