Extended Hypercube: A Hierarchical Interconnection Network of Hypercubes

J. Mohan Kumar and L. M. Patnaik, *Senior Member, IEEE*

***Abstract*—A new interconnection topology—the Extended Hypercube-consisting of an interconnection network of k-cubes is discussed. The extended hypercube is a hierarchical, expansive, recursive structure with a constant predefined building block. The extended hypercube retains the positive features of the k-cube at different levels of hierarchy and at the same time has some additional advantages like reduced diameter and constant degree of a node, This paper presents an introduction to the topology of the extended hypercube and analyzes its architectural potential in terms of message routing and executing a class of highly parallel algorithms. Topological properties and performance studies of the extended hypercube are presented.**

***Index* Terms—Hypercube, interprocessor communication, message routing, multiprocessor, parallel algorithms.**

I. INTRODUCTION

In recent years considerable progress made in the design of integrated circuit technology has resulted in the emergence of highly powerful processors. Several new parallel architectures have been proposed [5], [8], [9], [10], [16] to increase computing speeds to complement the advances in VLSI design. But to this day the problem of interconnecting processors to achieve high computational bandwidth has not been fully solved. Increased parallelism means more communication among processors and hence a corresponding increase in overheads. Internode distance, message traffic density, and fault-tolerance are dependent on the diameter and degree of a node. The product (diameter * degree of a node) is a good criterion to measure the cost and performance of a multiprocessor system [5]. An interconnection network with a large diameter has a very low message passing bandwidth and a network with a high degree of node is very expensive. In addition, computing systems should be easily expandable; there should be no changes in the basic node configuration as we increase the number of nodes.

The binary hypercube, henceforth referred to as the hypercube in this paper, has a robust topology [3]. The hypercube has been employed to solve several problems [18], [21]. But the hypercube networks are not truly expandable because we have to change the hardware configuration of all the nodes whenever the number of nodes grows exponentially, as the nodes have to be provided with additional ports. The hypercube network does not have a constant predefined building block [9]. This is a very significant factor from the architectural/hardware point of view. Moreover, an incompletely populated hypercube lacks some of the properties which make the hypercube attractive in the first place. Complicated routing algorithms are necessary for the incomplete hypercube [8]. Several modifications of the hypercube structure have been investigated in recent years to overcome the shortcomings of the topology of the hypercube. In [4] and [5], Bhuyan and Agganval have proposed a general class of hypercube structures to achieve a very good cost factor, (degree of a node) * (diameter). Goodman and Sequin [9] have proposed a truly expansive tree structure—the Hypertree. Hypernet, a network of hypercubes is discussed in [10]. The hypernet is a versatile architecture for a variety of applications. It has the good features of a complete binary tree and a binary hypercube and avoids some of the drawbacks of the two architectures. A new hierarchy of hypercube interconnection topologies has been discussed by Lakshmivarahan and Dhall in [13]. This scheme includes the binary hypercube topology and in general has a high degree of a node and low diameter. Hierarchical Cubic Networks (HCN's) are discussed by Ghose and Desai in [8]. The HCN's are hierarchical networks and use hypercube networks as nodes.

In the schemes reported in [5], [9], and [10] and most of the other popular communication schemes, the processor node has to perform two main functions, viz., 1) computation tasks and 2) communication tasks. An efficient communication scheme is one in which the processor nodes perform more of computation tasks and less of communication tasks. But in many of the Hypercube and related architectures [5], [9], [10] the processor nodes are involved in communicating messages between their neighbors. A good utilization factor defined in Section III has to be achieved to increase the efficiency of a multiprocessor system.

In this paper, we discuss a new interconnection scheme—the Extended Hypercube (EH), earlier discussed in [12] and [15]. This scheme combines some of the topological features of the architectures proposed in [8]–[10] and at the same time retains the attractive features of the hypercube topology to a large extent. The EH is built using basic modules consisting of a k-cube of processor elements (PE's) and a Network Controller (NC) as shown in Fig. 1. The NC is used as a communication processor to handle intermodule communication; 2^k such basic modules can be interconnected via 2^k NC's, forming a k-cube among the NC's. The EH is essentially a truly expansive, recursive structure with a constant predefined building block.

Manuscript received January 1, 1990; revised August 10, 1990.

J. M. Kumar is with the Microprocessor Applications Laboratory, Indian Institute of Science, Bangalore 560 012, India.

L. M. Patnaik is with the Microprocessor Applications Laboratory, Supercomputer Education and Research Center and the Department of Computer Science and Automation, Indian Institute of Science Bangalore 560 012, India.

IEEE Log Number 9103980.

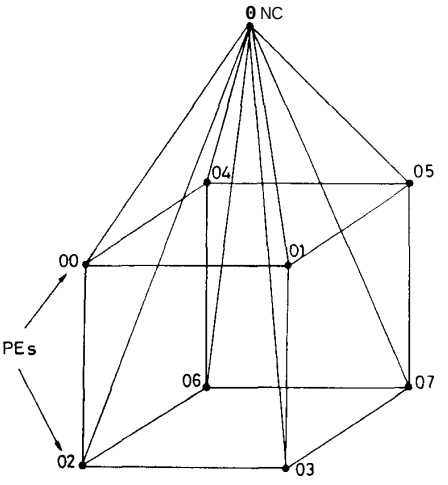


Fig. 1. Basic module of the extended hypercube, e.g., EH(3,1)

The number of 1/0 ports for each PE (and NC) is fixed and is independent of the size of the network. The EH structure is found to be most suited for implementing a class of highly parallel algorithms, viz., the DESCEND and ASCEND class described in [16]. The EH can emulate the binary hypercube in implementing a large class of algorithms with insignificant degradation in performance. The utilization factor of the EH is higher than that of the Hypercube. This paper is organized as follows. In the second section we present a summary of the EH topology and discuss about the features of the EH architecture. A method for addressing the nodes of the EH is also discussed in Section 11. In the third section, we discuss the properties of the EH and performance evaluation of the EH. In the fourth section we discuss message routing procedures and implementation of parallel algorithms on the EH. A comparison of the parameters of the EH architecture with those of the other multiprocessor architectures is done in Section V. The last section concludes the paper with a few comments on the capabilities of this topology.

11. The Extended Hypercube

In this section, a formal introduction to the EH architecture is given and a methodology for addressing the nodes of the EH is also discussed. The EH architecture is suited for hierarchical expansion of multiprocessor systems. The basic module of the EH consists of a k-cube and an additional node for handling communication—the Network Controller (NC). There are a total of $[2^k * (k/2 + 1)]$ links in the basic module, comprising of the $[(2^k * k)/2]$ links of the hypercube and 2^k links for connecting the processor elements to the NC as indicated in Fig. 1. Message passing scheme among the nodes within a h-cube is performed by adopting one of the hypercube message passing algorithms [2], [17]. However, message passing among nodes at different levels of hierarchy is via one or more NC's.

The NC in effect allows the processor element (PE) of each node to concentrate on computation rather than on communication. The PE communicates directly with its neighbors (PE's at a Hamming distance of one) and the NC. All communications between noncontiguous nodes within the same k-cube are via the neighboring nodes, whereas communication among distant nodes belonging to different k-cubes is via one or more NC's. Thus, the PE's of an EH are involved in message transfer operations among the 2^k nodes of the same k-cube only whereas in a hypercube each PE is involved in the communication operations between all pairs of PE's in the hypercube. This improves the computation to communication ratio considerably.

An EH consisting of a k-cube and one NC will be referred to as the basic module or EH(k,1) (Fig. 1) in the rest of this paper. The EH(k,1) has two levels of hierarchy: the k-cube at the zeroth level and the NC at the first level. The hypercube consisting of the PE's is referred to as the HC(k,0). An EH(k,2) has 2^k k-cubes of PE's at the zeroth level, one k-cube of NC's at the first level, and one NC at the second level. In general an EH(k, 1) (1 is the degree of the EH), consists of one NC at the lth level and a k-cube of 2^k NC's/PE's at the $(1-1)$st level. The NC's/PE's at the $(1-1)$st level of hierarchy form a k-cube. We will refer to this cube as HC(k, $1-1$). The NC's/PE's at the $(1-2)$nd level of hierarchy form 2^k distinct k-cubes which will be called as HC(k, $1-2$)'s. Thus, we have h-cubes at all levels j for $0 \leq j < 1$. The $2^{k*(l-1)}$ number of HC(k,0)'s are all computation HC's and the HC(k, j)'s (for $j \geq 1$) are all communication HC's. The basic module of the EH is a constant predefined building block and the node configuration remains the same regardless of the dimension of the EH. The basic module is said to be an EH of degree one, denoted by EH(k,1). The EH architecture can easily be extended by interconnecting the basic modules. We can interconnect eight EH(3,1)'s (basic modules) to get an EH(3,2) as shown in Fig. 2. The interconnection topology formed by the 3-cube of NC's at the first level and a controller at the second level is identical to that of the basic module. Thus, we have a hierarchical structure consisting of 64 PE's at the zeroth level (lowest level), eight NC's at the first level, and one NC at the second level. In general, the EH is a single rooted, 2^k-ary tree of height 1, with additional horizontal links. Every node in an EH(k, 1) is connected to its neighbors: k siblings at level j $(0 \leq j < l)$, 2^k children at level $(j-1)$, and one parent at level $(j+1)$. The additional horizontal links form k-cubes among the 2^k children of every node. In an EH(k, 1) there are $2^{k*(l-j-1)}$ for $0 \leq j < l$, k-cubes at any level j, with $1/2^k$ as many nodes as at the previous level. Topologically the EH(k, 1) differs from the pyramid topology and the hypertree. In a k-PC Pyramid topology [14], [19] a PE at level 1 is connected to $2 * k$ siblings at level j, 2^k children at level $(j-1)$, and a parent at level $(j+1)$. The additional horizontal links form a k-mesh connected computer. A hypertree is a single rooted, binary tree with every node at level j having j siblings, two children at level $(j+1)$, and one parent at level $(j-1)$. The horizontal links form a hypercube among themselves.

A. Addressing the Nodes of the EH

As explained in Sections I and II, the EH(k, l) consists of several HC(k, j)'s $(0 \leq j < 1)$ each with a topology identical

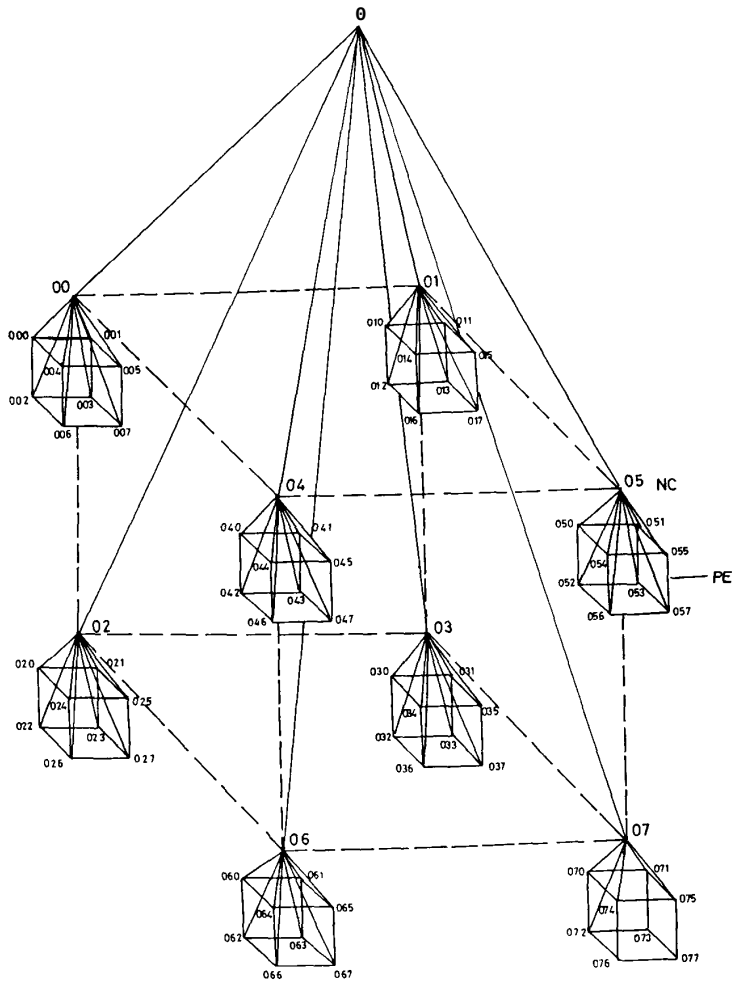


Fig. 2. A 64-node extended hypercube EH(3,2)

to that of a k-cube. In other words, at all levels of hierarchy, we have an interconnection of k-cubes. We will use the term "node" in general for the PE's and NC's of the EH. **As** shown in Fig. 1, the PE's of the basic module or EH(k, 1)'s can be addressed as $0, 1, \cdots, M$ $(M = 2^k - 1)$. Now if the NC of the EH(k, 1) is identified by N, then the PE's of this EH(k,1) are addressed as $NO, N1, N2, \cdots, NM$. The address of the NC precedes the address of the PE. In general, the address for an arbitrary node *"A"* (Fig. 3) at the zeroth level can be written as $D_l D_{l-1} D_{l-2} D_{l-3} \cdots D_0$, each D_i $(0 \leq i \leq$ I) is a k-bit mod 2^k number. Here D_0 corresponds to the address of a node in HC(k, 0), D_1 corresponds to the address of the node (NC) at the first level (to which the first node is connected), D_2 corresponds to the address of the node (NC) at the second level, and so on. Consider an EH of degree "l," a node at the zeroth level will have an (I + 1) digit address, a node at the first level will have an l digit address, a node at the second level will have an $(l - 1)$ digit address, etc. The solitary node at the Ith level has a one digit address, viz., 0.

For example consider the EH(3,3) of Fig. **3.** The NC at the top level, also referred to as the root **NC,** has an address "0." The children of the root NC have eight children at level 2, and the siblings at level 2 are interconnected to form a 3-cube, viz., HC(3,2). The nodes of the HC(3,2) are labeled as 00, 01, 02, ..., 07, and each node has eight children at level 1. The nodes of level 1 form eight hypercubes, viz., HC(3,1)'s. The nodes in the HC(3,1) are addressed as 000, 001, ... 007, 010, 011, ... 017, ..., 077. Finally, the nodes of HC(k, 0)'s are addressed as follows: 0000, 0001, ... ,0007, 0010, 0011, ..., 0017, ... , 0770, 0771, ..., 0777.

III. PROPERTIES OF THE EH

The EH is a loosely coupled network of computation and communication processors. The computation/communication processors are interconnected among themselves as k-cubes. In addition, the computation processors (the PE's) of any k-cube are connected to a communication processor (the NC). In this section we shall refer to the processors of the EH as

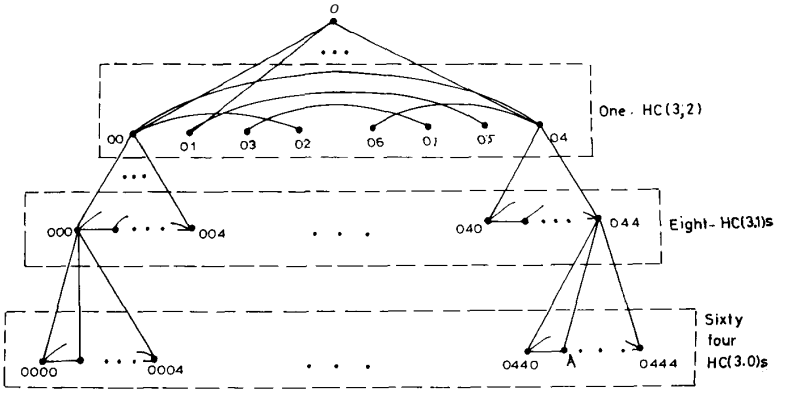


Fig. 3. The EH of degree three—EH(3,3) showing HC(3,0)'s, HC(3,1)'s, and HC(3,2)'s.

nodes. In the case of the basic module, e.g., EH(3,1), there are four parallel paths between any two nodes (Fig. 4). For example, the parallel paths between 00 and 07 are given by

1) 00→ 01→ 03→ 07, 2) 00→ 02→ 06→ 07,

3) 00→ 04→ 05→ 07, and 4) 00→ 0— 07.

Obviously, the last path via "0" is the desired path as it is the shortest path and it does not involve any of the PE's. However, in an EH(k, l), where $1 > 1$, we utilize the NC's only for intermodule and level-to-level communications. In an EH of degree greater than one, there are no parallel paths from one level to another. We shall refer to the paths via the NC or NC's as extended links. For example in Fig. 4, 00 → 0 → 07 is an extended link of the EH whereas 01 → 03 is a link of the EH.

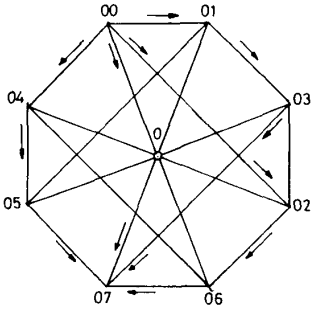


Fig. 4. Parallel paths in EH(3,1)

A. Spanning Tree of the EH

The EH is a connected graph and hence it contains a spanning tree, i.e., a subtree of the graph which contains all the nodes of the graph. The procedure for generating the spanning tree of the basic module is as follows, 1) at the root or the first level, we place the node corresponding to the NC, 2) at the second level two neighboring nodes of the EH are placed as children of the root as shown in Fig. 5(a), then we place all the remaining odd nodes as the childredgrandchild of the odd node at the second level and the rest of the even nodes as the children/grandchild of the even node at the second level (the levels referred here are different from the hierarchical levels of the EH). To obtain the spanning tree of the cube of higher dimensions, identical trees "ST" of Fig. 5(a) are appended to all the nodes of the first level of hierarchy as indicated in Fig. 5(b).

B. Performance Analysis of the EH

In this section the performance of the EH will be compared to that of the other hypercube structures. Performance studies of multiprocessor systems have been carried out in the past. The performance characterestics of the binary hypercube have been discussed by Abraham and Padmanabhan in [1]. The performance of hierarchically interconnected multiprocessors is discussed in [20]. In [11] Indurkhya *et al.* have investigated into the problem of optimal partitioning of distributed programs in a multiprocessor system. The objective of distributed programming is to exploit parallelism and minimize execution time. A task assignment problem assigns every node of the problem graph to some processor element (PE) of the multiprocessor system. Execution times are functions of task computation times and processor-to-processor communication times. Indurkhya *et al.* have modeled the execution time of an assignment by summing all communication times from processor-to-processor and the execution time of the busiest processor. The execution time of the assignment when the PE's execute concurrently is given by

$$\text{Execution time, } T_e = T_r + T_c \tag{1}$$

where T_r is the execution time of the busiest processor and T_c is the communication time, to send a message to all the processors. We define the utilization factor of any PE as, σ = (Time spent by the PE in computation) / (Time spent by the PE in communication). Let R denote the average computation time of a task and C denote the average communication overhead involved in communicating with a neighboring PE. For a fully connected system, there exists a communication channel between every pair of processors. In such a case, the execution time is given by

$$T_e = R + C * (N - 1) \tag{2}$$

where N is the number of PE's.

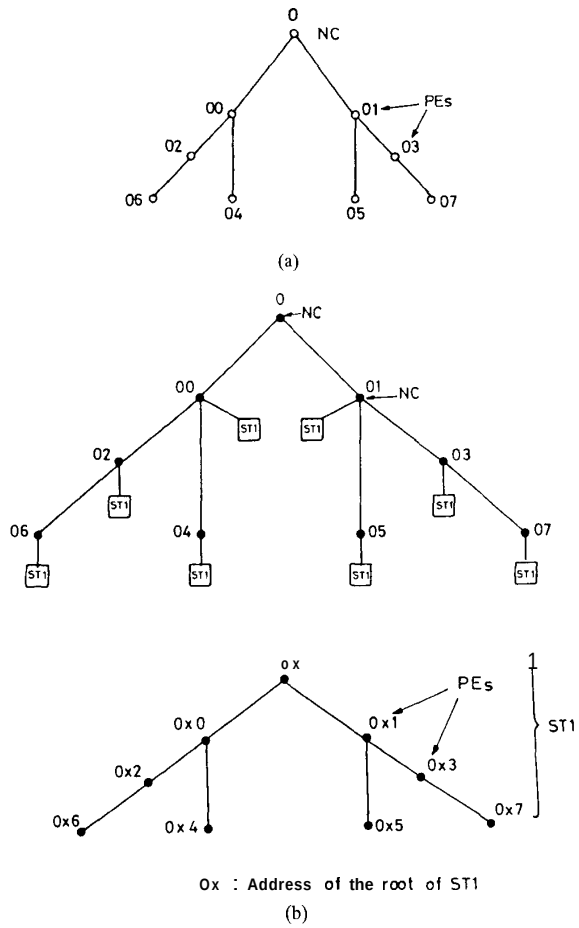


Fig. 5. (a) Spanning tree of EH(3,1). (b) Spanning tree of EH(3,2).

In an n-cube, the PE's have to perform message passing operations between their neighboring PE's, hence

$$T_e = R + C*(N-1) + C\sum_{i=1}^{n}(i-1)^n C_i. \qquad (3)$$

In an EH the PE's perform communication functions related to nodes within the k-cube only, hence $T_e = R + C*(N-1) + \max[C\sum_{i=1}^{k}(i-1)^{*k} C_i, S]$; δ is the switching delay at the NC's, and $\delta \ll (R + (N-1)*C$.

$$= R + C*(N-1) + C\sum_{i=1}^{k}(i-1)^{*k} C_i. \qquad (4)$$

The utilization factor is given by

$$\sigma = \{R/[C*(N-1) + C\sum_{i=1}^{n}(i-1)^{*n} C_i]\} \text{ for the n-cube}$$

$$\sigma = \{R/[C*(N-1) + C\sum_{i=1}^{k}(i-1)^{*k} C_i]\} \text{ for the EH.}$$

The utilization factor of a multiprocessor system is a measure of the efficiency of the internode communication mechanism. However, the overall performance of a multiprocessor system is also affected by factors such as problem partitioning and task allocation, as discussed in [6] and [7].

C. *Choice of k and 1 for an EH*

In the following analysis, we shall refer to the communication among the nodes of a k-cube as local communication and communication among nodes of different basic modules, and among nodes at different levels as global communication. The dimension of the basic module is determined on the basis of C_l , which is given by,

$$C_l = \frac{\text{(number of communication cycles with PE's of the same k-cube)}}{\text{(number of communication cycles with PE's in other } k\text{-cubes)}}.$$

For a large value of C_l, it is desirable to have a large k and for low values of C_l, k should be small. The time spent by the PE in handling communication is proportional to the value of k, the dimension of the basic module. In a hypercube system, the time spent by the PE's is proportional to the value n, the dimension of the n-cube. In an EH, the NC's are used primarily to handle all global communications. All PE's transmit their nonlocal messages to the NC. In a communication-intensive problem such as total exchange, an NC at level j of an EH(k,l) $(1 \le j < l)$, handles $2*[\{2^{k*(j+1)}-1\}/(2^k-1)]*[\{2^{k*(l+1)}-2^{k*(j+1)}\}/(2^k-1)]$ messages. In the following analysis, we compare EH(k,l)'s with hypercubes having 2^{k*l} nodes considering only the PE's of the EH. The number of NC's in an EH(k.l) is $[2^{k*l}-1]/[2^k-1] \simeq 1/[2^k-1]$ times the number of PE's, a fraction of the number of PE's, hence we are justified in comparing EH(k,l)'s having 2^{k*l} PE's with hypercubes having $2^n (n = k*l)$ nodes. The diameter of the EH is given by $[k + 2*(l-1)]$, whereas the diameter of a hypercube with identical number of nodes is equal to $(k*l)$. For $k = 2$ the diameter of the EH$(2,l)$ is equal to that of the n-cube, where $n = 2*l$. For $l = 1$, the diameter of the EH(k,l) is equal to that of the n-cube, since $n = k*1$. For $k > 2$ and $l > 1$, the diameter of the EH(k,l) is always less than that of the n-cube, $(n = k*l)$. The diameters of EH(k,l)'s with varying values of k and 1 are tabulated in Table I. It is clear that the cost factor given by (diameter * degree of a node) for an EH(k,l) for $k > 2$ and $l > 1$, is less than that of the n-cube $(n = k*l)$. If k is very high and l is small, then the properties of the EH topology are similar to those of the hypercube. The utilization factor of the EH decreases with increase in the value of k. The NC's are used primarily to handle all global communications. All PE's transmit their nonlocal messages to the nearest NC. In a communication-intensive problem such as total exchange, where every node sends a different message to every other node in the network, total number of messages to be handled by the NC's of an EH at different hierarchical levels and for varying values of k are tabulated in Table II.

Let us consider EH's EH(2,6), EH(3,4), and EH(4,3) having identical number of PE's, viz., N = 4096. The volume of message handled by the NC's at lower levels is more than that handled by NC's at higher levels since there are more NC's

TABLE I
DIAMETERS OF EH(k,1)'s

l / K	1	2	3	4	5	6	7	8
2	2	4	6	8	10	12	14	16
3	3	5	7	9	11	13	15	17
4	4	6	8	10	12	14	16	18
5	5	7	9	11	13	15	17	19
6	6	8	10	12	14	16	18	20

TABLE II
NUMBER OF MESSAGES HANDLED BY NC'S AT DIFFERENT HIERARCHICAL LEVELS

EH Dimension	Total-No. of Messages	Hierarchical Levels				
		$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
EH(2,2)	210	160	-	-	-	-
EH(2,3)	2752	800	2016	-	-	-
EH(2,4)	64K	3360	13k	43K	-	-
EH(2,5)	1M	14K	56K	217K	698K	-
EH(2,6)	4M	53K	223K	906K	3M	32M
EH(3,2)	4032	1152	-	-	-	-
EH(3,3)	25K	10K	74K	-	-	-
EH(3,4)	4M	82K	667K	4.7M	-	
EH(3,5)	32M	658K	5.3M	42M	300M	-
EH(3,6)	256M	5.1M	42M	334M	2.7G	19.1G
EH(4,2)	64K	8740	-			
EH(4,3)	16M	148K	2.2M	-	-	-
EH(4,4)	4G	2.26M	36.4M	146M	-	-

at lower levels. Also, the degree of a node for both PE's and NC's increase with k, hence for a given value of N, each NC of an EH(k_1, l) handles less communication tasks than the NC of an EH(k_2, l) does, where $k_1 > k_2$. With $k = 6$, the degree of a node of the NC is seventy one! hardware implementation of the NC's with D_{nc} the degree of an NC given by $(2^m + k + 1)$ is difficult for large values of k. From the above discussion and Table II we conclude that it is desirable to have $2 \le k \le 6$.

D. Parameters of the EH(k,1)

In this section we analyze the EH(k, 1) and determine the network parameters of the EH(k, 1) topology. The analysis for the EH(k,1) facilitates the understanding of the local communication among the PE's of a k-cube. The EH(k, 1) of Fig. 4 consists of nine nodes; eight PE's and one NC, which is essentially the topology of the basic module. The diameter of the EH(k, 1) defined as the maximum distance between any two nodes in an EH(k, 1) is k if the NC is not used for local communication within the k-cube. For EH(k, j)'s with $j > 1$ it is not desirable to use the NC's for communication among the nodes of the same k-cube, as the NC has to handle intermodule communication. However, if $l = 1$ then the NC can **be** effectively used for communication among noncontiguous nodes of the same k-cube. In the following we determine the parameters of EH(k, 1) with the NC utilized for communication among nodes of the same k-cube. In Section III-E we determine the parameters for the EH(k, 1) in general. The diameter of the EH(k, 1) is $\{[k * 1) + (2^k - k - 1) * 2]/(2^k - 1)\}$ and the EH(k, 1) has $2^k * (k/2 + 1)$ links: $(2^k * k)/2$ to form HC(k, 0) and 2^k links connected to the NC. The number of links in the EH(k, 1) denoted by I_1 is given by

$$I_1 = 2^k * (k/2 + 1)$$

In an EH(k, 1) the degree of a PE is always $(k + 1)$: k links to the neighbors and an additional link to the NC. The degree of the PE, D_{n1}, is thus equal to $(k + 1)$. The average message distance in EH(k, 1) is given by $d_1 = (k/2) * [(2^k)/(2^k - 1)]$, similar to that of a hypercube if the NC is not used for message passing within the HC(k, 0), whereas $d_1' = [k + (2^k - k - 1) * 2]/(2^k - 1)$, if the NC is used for message passing within the HC(k, 0). The average message density in EH(k, 1) is given by

ρ = [(average path length) * (number of nodes)] / (total number of links).

$\rho = \{(k/2) * [(2^k)/(2^k - 1)] * (2^k)\}/\{2^k * (k/2)\}$ when NC is not used for message passing within the HC(k, 0), and

$\rho = [\{k + 1 + (2^m - k - 1) * 2\} * \{(2^k + 1)\}/[2^k * \{2^k * (k/2 + 1)\}]$ when NC is used for message passing within the HC(k, 0). The variation in the average distance of HC(k, 0) for different values of k is shown in Fig. 6.

The analysis of the EH with L hierarchical levels ($L = l + 1$, and $l > 0$) is carried out hereunder. The various parameters of the EH are defined and presented in Table III.

E. Parameters of the EH(k, l)

The number of nodes or PE's in the EH is equal to a power of 2^k. The number of PE's in the EH, denoted by N, is given by

$N = 2^{k*l}$, where 1 is the degree of the EH and $n = k * l$.

Of course it is also possible to have EH's with 2^n number of processors, where $n \# k * 1$. But in such cases, the NC's at the top two levels of hierarchy do not form an EH (Fig. 7). In an EH(k, 1) the PE's reside at the zeroth level of hierarchy. There are no NC's at the zeroth level, i.e., an EH with only one PE does not have an NC. All the NC's reside at hierarchical levels one and upwards. The number of NC's in the EH denoted by M is given by

$$M = 0, \text{ for } l = 0 \text{ and}$$
$$M = (2^{k*l} - 1)/(2^k - 1), \text{ for } l \ge 1.$$

For $l \ge 1$, the number of links in the EH is given by

$$I = 2^k * (k/2 + 1) * 2^{k(l-1)}[(1 - 2^{-k*l})/(1 - 2-91$$

The diameter of the EH is given by

$$D = 0, \text{ for } l = 0 \text{ and}$$
$$D = k + 2 * (l - 1). \text{ for } l \ge 1.$$

Each PE of the EH is connected to k neighbors and the NC. The degree of a node is a constant for all values of 1. Each NC is connected to 2^k NC's/PE's at a lower level, k of its neighbors, and an NC at the next higher level. The degree of a PE in an EH is $D_n = (k + 1)$ (for all 1). Whereas the degree of an NC is, $D_{nc} = (2^k + k + 1)$ (a constant for **all** l). The

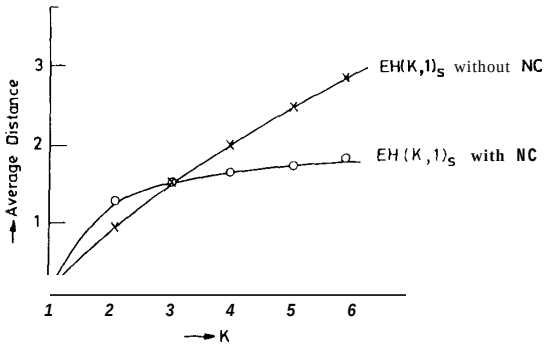


Fig. 6. Average distance for EH(k,l)'s.

TABLE III
PARAMETERS OF THE EH, E.G , EH$(3,l)$

l	1	2	3	4	5
n	3	6	9	12	15
M					
I)	20	180	1460	11700	93620
D	3 1	5 1	7 1	9 1	11
D_n	4	4	4	4	4
D_{nc}	12	12	12	12	12
C_m	14.64	22.54	35	45	57.31
d	1.58	2.91	4.46	6.01	10.07
ρ	0.711	1.326	1.792	2.40	3.49

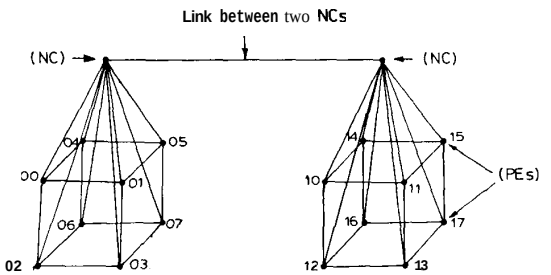


Fig. 7. Example of an incomplete EH comprising two EH(3,1)'s

cost factor of the network is given by the product, (degree of a node)*(diameter) [5].

In an EH the degree of node of the PE and the NC are different. The average degree of node is $D_{\text{nav}} = [(k+1)*2^{k*l} + (2''' + k + 1)^*(2^{k*l}-1)/(2^k-1)]/[2^{k*l} + (2''' + k + 1)*(2^{k*l}-1)]$. Hence, the cost factor $= [k + 2*(l-1)]*[(k+1)*2^{k*l} + (2''' + k + 1)*(2^{k*l}-1)/(2^k-1)]/[2^{k*l} + (2^k + k + 1)^*(2^{k*l}-1)]$.

The average distance in the EH: The average distance within any HC$(k,j)(1 \le j < l)$ is a very small value. But due to the fact there is only one link from one level to another, the average distance increases with the degree of the EH. The general expression for the average distance in an EH of degree l is given by

$$d = 2\sum_{i=0}^{\text{I}-1}[2^{k*(l-i)} \sum_{j=i+1}^{1} 2^{k(l-j)} * (j-2)+ \sum_{x=0}^{l-1}[2^{k*l} * (2''' - 1)]/[2^{k*x}]$$
$$*[\sum_{y=0}^{l-x-1} (2^{k*y}) * [(2*y) + (k/2) * [(2^k)/(2^k - 1)]]$$

In an EH of degree l and consisting of N nodes, for any arbitrary node there are $\sum_{j=0}^{l}\sum_{l=1}^{k} 2^{k*j} * {}^kC_i * (2*j+1)$ nodes at a distance of $(2*j+i)$ for $1 \le i \le k$, and $0 < j < l$. **As** an example, the number of nodes and their distances from a source node say $(0000)_8$ to other nodes in **an** EH of degree three is shown in Fig. 3. The average distance is very close to one for an EH of degree one, but increases with the increase in the degree of the EH. The variation in the value of *"d"* with the number of nodes is indicated by the plot in Fig. 10(a).

Message Traffic Density: The message traffic density in the EH is given by

$$\rho = (\text{Average message distance} \times \text{Number of nodes}) / (\text{Number of links})$$
$$\rho = [d*(N+M)]/I.$$

Table III gives the various parameters N, n, M, D, d, and ρ for an EH(k.l). In Section V we compare the afore-said parameters with those of other hypercube related topologies.

IV. MESSAGE ROUTING IN THE EH

Using the NC, the interprocessor message traffic of a module gets redistributed into two categories, viz., local communication and global communication. Communication among the PE's belonging to the same k-cube is categorized as local communication. Local communication is handled by employing hypercube message passing algorithms. At zeroth level, the local communication is taken care of by the PE's: the time spent by each PE in communicating messages is proportional to the value of k. The NC's are involved in global communication. The single network controller at the topmost level retransmits messages to/from the host system from/to the PE's via the network of NC's. The message passing operation in local communication involves 1) the source PE, 2) upto $(k-1)$ PE's within the k-cube, and **3)** the destination PE. The message passing operation in the global communication involves 1) the source PE, 2) upto $2(l-1)$ NC's, and 3) the destination PE.

A. *Message Routing Procedure*

The message is formatted at the source node with the source address, destination address, message length, and a few semaphore bits [8]. The addresses are indicated in the octal number system for an EH(3,3) as described earlier. Let us consider the topology given in Fig. 8. The address of the node marked "A" is 0435. Here "0" corresponds to the address of the NC at the third level, "04" corresponds to the address

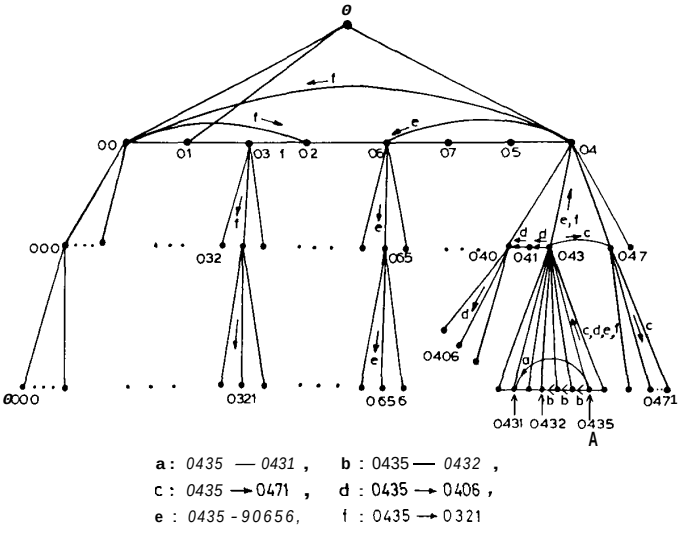


Fig. 8. Message routing in EH(3,3).

TABLE **IV**
MESSAGE ROUTING SEQUENCE

Destination	Routing Sequence from 0435	Distance
0431	0435 → 0431	1
0432	0435 → 0437 → 0433 → 0432	3
0471	0435 → 043 → 047 → 0471	3
0406	0435 → 043 → 04 → 040 → 0406	4
0656	0435 → 043 → 04 → 06 → 065 → 0656	5
0321	0435 → 043 → 04 → 00 → 02 → 03 → 032 → 0321	7

of the NC at the first level and "0435" corresponds to the address of the PE at the zeroth level. The routing sequence for different destination nodes at different hierarchical levels is shown in Table IV. The transfer of message between two nodes at different levels of hierarchy is referred to as the vertical shift and the transfer of message within a hypercube is referred to as the hypercube shift.

The procedure for routing a message from the host system to a PE involves a vertical shift from D_l^s to $D_l^d D_{l-1}^d D_{l-2}^d \cdots D_0^d$ via $D_l^d D_{l-1}^d, D_l^d D_{l-1}^d D_{l-2}^d, D_l^d D_{l-1}^d D_{l-2}^d \cdots D_1^d$. (note that $D_l^s = D_l^d$). D_{l-i}^s is the $(1-i)$th digit of the address of the source node and D_{l-i}^d is the $(l-i)$th digit of the address of the destination node. In general the procedure for routing a message from a source PE with address $D_l^s D_{l-1}^s D_{l-2}^s \ldots D_0^s$ to a destination PE with address $D_l^d D_{l-1}^d D_{l-2}^d \cdots D_0^d$ is as follows:

```
Proc Message Rout
 begin
   if [D_l^s D_{l-1}^s D_{l-2}^s ... D_0^s] = [D_l^d D_{l-1}^d D_{l-2}^d ... D_0^d]
      then destination is source; terminate.,
    else set j = 0 ;
   while j = 0
    do for i = 1 to 1
      begin
         if D_{l-i}^s = D_{l-i}^d then j = i;
      end
    begin
      vertical shift from D_l^s D_l^s - 1 D_l^s - 2 · · · D_0^s to
          D_l^s D_{l-1}^s D_{l-2}^s ... D_j^s
      hypercube shift from D_l^s D_{l-1}^s D_{l-2}^s · · · D_j^s to
          D_l^d D_{l-1}^d D_{l-2}^d ... D_j^d
      vertical shift from D_l^d D_{l-1}^d D_{l-2}^d ... D_j^d to
          D_l^d D_{l-1}^d D_{l-2}^d ... D_0^d
   end
end.
```

Theorem: The diameter of the EH topology is $[k+2(l-1)]$. For message transfer between two PE's located at different $\mathrm{HC}(k,0)$'s there are two vertical shifts and one hypercube shift. Each vertical shift involves a maximum of $(l-1)$ hops and the hypercube shift involves a maximum of k hops. Hence, the diameter of the $\mathrm{EH}(k,l)$ is $[k+2(l-1)]$.

B. Execution of Parallel Algorithms on the EH

The EH can emulate the performance of the binary hypercube in executing the DESCEND/ASCEND class of algorithms described in [16]. Data items $t0, t_1, \cdots, t_{N-1}$ are stored in locations $T[0], T[1], \ldots, T[N-1]$. **An** algorithm is in the DESCEND class if it performs a sequence of basic operations on pairs of data that are successively $2^{n-1}, 2^{n-2}, \cdots, 2^0 = 1$ locations apart, where $2^n = N$. A basic operation of

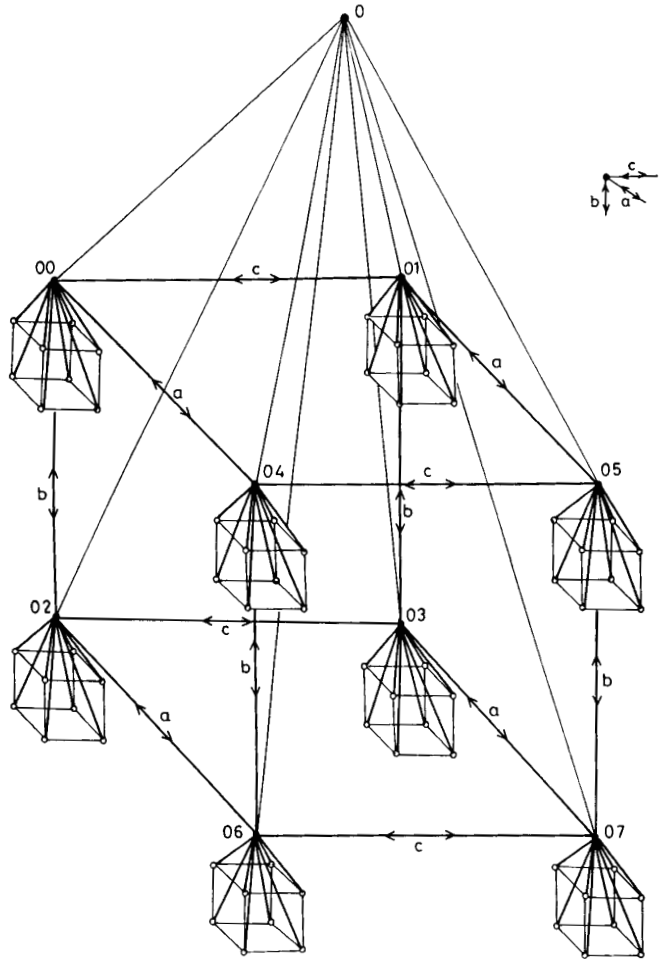


Fig. 9. (a) Implementation of DESCEND algorithm on EH(3,2). Stage 1: K = 5 to 3.

DESCEND, OPER$(m,h;U,\mathrm{V})$ modifies the data items of locations U and V; OPER may depend on m and h, and m and h are integers such that $0 \leq \mathrm{m} < N$, $0 \leq h < n$, and $N = 2^n$. Binary representation of $\mathrm{m} = \mathrm{bit}_h(m) * 2^h$. An algorithm of the DESCEND class is specified below [16]:

Algorithm 1

proc DESCEND
  for $h \leftarrow n - 1$ step -1 until $h = 0$
  do for each m: $0 \leq \mathrm{m} < N$
    pardo if $\mathrm{bit}_h(m) = 0$ then OPER $(\mathrm{m}, h; T[m], T[m+2^h])$
      fi
    odpar
  od
corp DESCEND

In the dual class, ASCEND OPER is performed on data items that are successively $1 = 2^0, 2^1, \ldots, 2^{n-1}$ locations apart and the control of the algorithm is "for $h \leftarrow 0$ step 1 until $h = n - 1$."

Algorithms for bitonic merge and cyclic shift are directly within the ASCEND and DESCEND classes. Applications such as permutation, shuffle, unshuffle, bit-reversal, odd-even-merge, FFT, convolution, matrix transposition have programs consisting of sequences of algorithms in the ASCEND/DESCEND class and run in $O(\log N)$ parallel steps while other applications such as bitonic sort, odd-even-sort, and calculation of symmetric functions involve recursive execution of ASCEND/DESCEND algorithms and run in $O((\log N)^2)$ parallel steps. The n-dimensional binary hypercube is the most suitable topology for executing the algorithms of this class in $O(\log N)$ parallel steps. But the demand for high degree of a node, $n = \log N$ is not only expensive in terms of cost but also difficult for hardware implementation. The unfolded n-cube and the shuffle-exchange networks have been used in the past to overcome the disadvantages of the binary hypercube topology. Hwang and Ghosh [10] have discussed the complexity of executing various algorithms on the hypernet and other topologies.

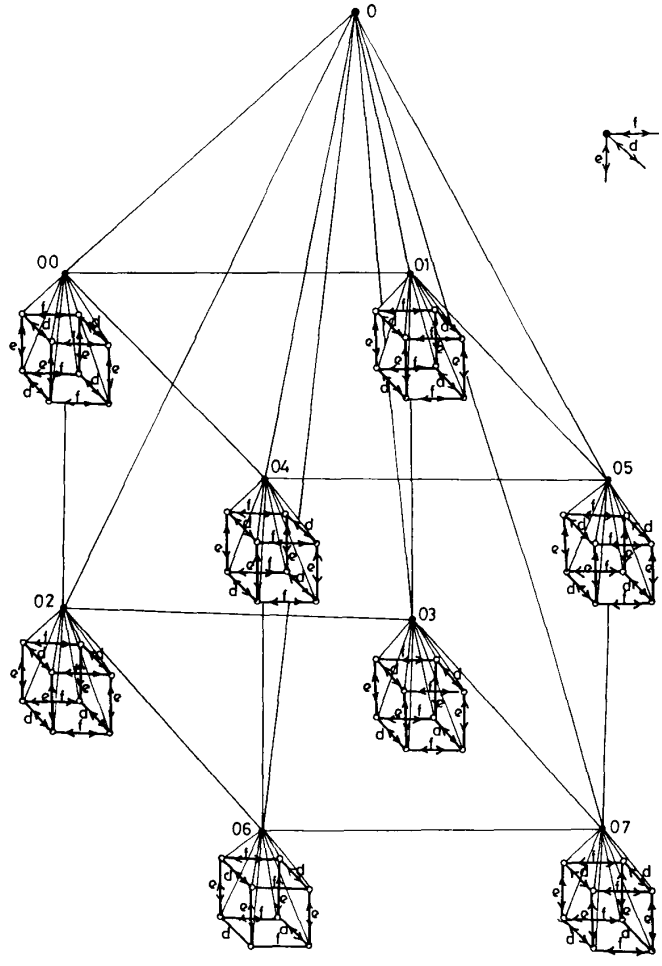


Fig. 9. *(Continued).* **(b)** Implementation of DESCEND algorithm on EH(3,2). Stage 2: K = 2 to 0

The cube connected cycles (CCC) described by Preparata and Vuillemin in [16] combines the principles of parallelism and pipelining to emulate the binary hypercube and the shuffle-exchange network with no significant degradation in performance but with a more compact structure. In a CCC the degree of a node is reduced to three. Preparata ***et al.*** have described the execution of the ASCEND/DESCEND class of algorithms on the CCC in $O(\log N)$ parallel steps. The function OPER described earlier can be executed by performing repeated cyclic shifts within each cycle of the CCC to emulate the binary hypercube. This limitation is overcome to a certain extent by pipelining analogous operations OPER (, i; ,,) for $r < i < n$ where 2^r is the length of the cycle. All the processor elements of the CCC are involved in the cyclic shift operations: in other words, the PE's of the CCC are used in shifting data items of other PE's resulting in a low utilization factor.

In an EH$(k,1)$ the DESCEND/ASCEND class of algorithms can be executed in $O(\log N)$ parallel steps and the PE's of the EH do not perform data transfer operations of other PE's. The Algorithm **1** can be used to implement DESCEND in the EH(k,l), where $n = k * 1$. However, as the connectivity of the EH(k,l) is not equal to $\log N$, the basic operation OPER $(m,h;T[m],T[m+2^h])$ is not as straightforward as it is in the case of the hypercube. In the EH, for $0 \leq h < k$, each pair $(T[m],T[m+2^h])$ is separated by a single link of the k-cube, whereas for $k \leq h < n$, each pair $(T[m],T[m+2^h])$ is separated by an extended link of the EH$(k,1)$. The basic operation OPER is performed $\log N = n$ times: for an EH(3,2) with 2^6 PE's, OPER is performed six times, i.e., for $h \leftarrow 5$ step -1 until 0. The six steps are shown in Figs. 9(a) and (b). In Fig. 9(a) the extended links indicated as *a*, *b*, and *c* are active and in Fig. 9(b) links of the k-cubes indicated by *d*, *e*, and *f* are active. OPER for $3 \leq h \leq 5$ involves PE's which are separated by extended links. As indicated in Fig. 9(a), for $h = 5$, PE's 001,002, ..., 007 and PE's 040,041, ..., 047 are separated by extended link *"a"* connecting NC's 00 and 04.

The address of a PE of an EH(k,l) is given by $D_l D_{l-1} \ldots D_0$ in digit form and $b^l_{k-1} b^l_{k-2} \cdots b^l_0 b^{l-1}_{k-1} b^{l-1}_{k-2} \ldots b^{l-1}_0 \ldots b^0_{k-1} b^0_{k-2} \ldots b^0_0$ in bit form, and the address of an NC at level j is given by $D_l D_{l-1} \cdots D_j$. We define

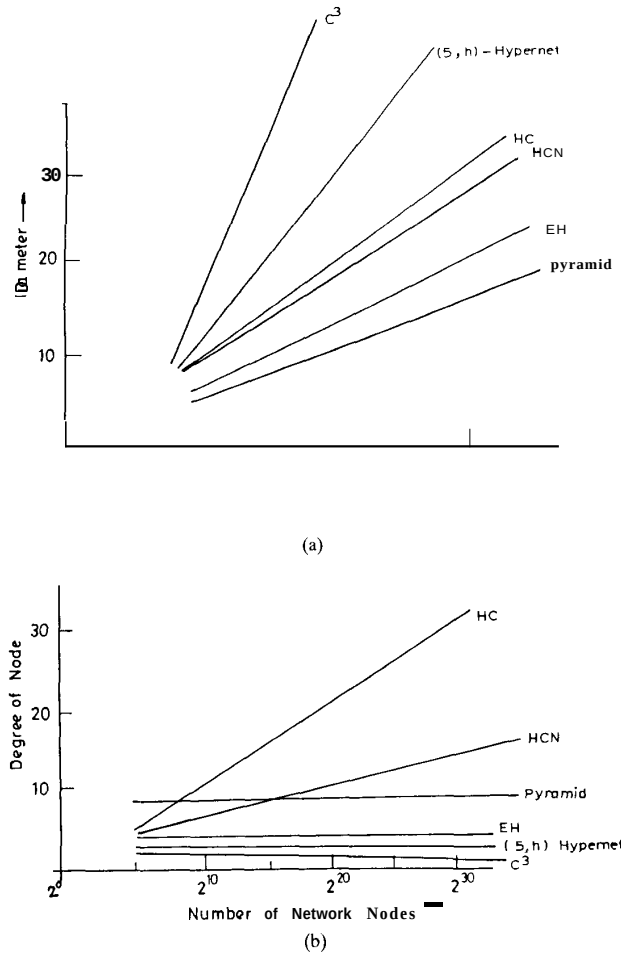


Fig. 10. (a) Comparison of diameters. (b) Comparison of degree of node.

OPERS a modification of the basic operation OPER as OPERS$(m, q; H[m], H[m+2^q])$, where $q = (k*j)+h$. OPERS is equivalent to

pardo for all pairs seperated by 24
    OPER$(m, q; T[m], T[m + 1]$
odpar

Algorithm 2 is the modified version of algorithm 1 for the implementation of the DESCEND in an EH(k, l). OPERS$(m, j, H[m], H[m + 2^j])$ involves $2 * (j - 1) + \log 2k$ steps. Hence, the DESCEND algorithm can be implemented in $(\quad + \quad * \log 2^k) = (\quad * \quad) = O(\log N)$ parallel steps.

Algorithm 2

proc EHDESCEND
  for $j \leftarrow l - 1$ step -1 until $j = 0$
    pardo for each HC(k, j)
      $D_j = b^j_{k-1} b^j_{k-2} \cdots b^j_0$
      for h $\leftarrow k1$ step -1 until $k = 0$
    do for each m: $0 \leq m < z$; $z = 2^k - 1$
      pardo if bit$_h(m) = 0$ AND $j = 1$
        then OPERS $(m, q; H[m], H[m + 2^q])$
        else OPER $(m, h; T[m], T[m + h])$
        fi
      odpar
    od
  odpar
corp EHDESCEND

V. COMPARISON OF PARAMETERS

In the following analysis, we compare the parameters of certain topologies, viz., linear array with N nodes, fully connected network with N nodes, hypercube(HC) with $2^n = N$ nodes, hierarchical cubical networks (HCN), cube connected cycles $[C^3]$, pyramid and EH$(3, l)$ with $[2^{3*l} + 2^{3*(l-1)}]$ nodes.

Diameter: The diameter of a multiprocessor system is defined as Maximum $\{d_{ij}\}$, where d_{ij} is-the shortest distance between nodes i and j. The diameter of the multiprocessor system is an important parameter in determining the performance of the network. The diameter of a linear array is $(N - 1)$ and on the other extreme the diameter of a fully connected network is unity. The diameters for the Hypercube and EH lie between

there need be no change in the hardware configuration of any node as the EH has a recursive structure.

VI. Conclusion

The extended hypercube is a hierarchical, extensive, recursive structure with a predefinable building block. The use of network controller nodes for each k-cube ensures efficient communication of messages via the interconnections among the network controllers. Further, the processor nodes perform more of computation tasks rather than communication tasks resulting in a high computation to communication ratio. The diameter and degree of a node of the EH have low values and hence the cost factor given by (diameter * degree of a node) of the EH is less than those of other topologies such as the binary hypercube, the cube connected cycles, the hypernet, the pyramid, etc. The ASCEND/DESCEND class of algorithms can be executed in $O(\log N)$ parallel steps on the EH.

Acknowledgment

The authors would like to thank the reviewers whose suggestions have improved the quality of the paper.

References

[1] S. Abraham and K. Padmanabhan, "Performance of direct binary n-cube network for multiprocessors," *IEEE Trans. Comput.*, vol. 38, pp. 1000–1011, July 1989.

[2] W. C. Athas and C. L. Seitz, "Multicomputers: Message-passing concurrent computers," *IEEE Comput. Mag*, pp. 9–24, Aug. 1988.

[3] B. Becker and H.-V. Simon, "How robust is the N-cube?," Tech. Rep., 05/1986, SFB124,BL.

[4] L.N. Bhuyan and D.P. Aggarwal, "Design and analysis of a general class of interconnection networks," in *Proc. IEEE Int. Conf. Parallel Processing*, Baltimore, MD, Aug. 1982.

[5] ——, "Generalized hypercube and hyperbus structures for a computer network," *IEEE Trans. Cornput.*, vol. C-33, pp. 323–333, Apr. 1984.

[6] Z. Cvetanovic, "The effects of problem partitioning, allocation, and granularity on the performance of multiple-processor systems," *IEEE Trans. Cornput.*, vol. C-36, pp. 421432, Apr. 1987.

[7] M. A. Franklin and S. Dhar, " Interconnection network: Physical design and performance analysis," *J. Parallel Distributed Comput.*, vol. 3, pp. 352–372, 1986.

[8] K. Ghose and K.R. Desai, "The HCN: A versatile interconnection network based on cubes," in Proc. Intl. Conf. on Supercomputers, 1989.

[9] J. R. Goodman and C. H. Sequin, "Hypertree: A multiprocessor interconnection topology," *IEEE Trans. Comput.*, vol. C-30, pp.923–932, Dec. 1981.

[10] K. Hwang, and J. Ghosh, "Hypernet: A communication efficient-architecture for constructing massively parallel computers," *IEEE Trans. Comput.*. vol. C-36, no.12, pp. 1450–1466, 1987.

[11] B. Indurkhya, H. S. Stone, and L. Xi-Cheng, "Optimal partitioning of randomly generated distributed programs," *IEEE Trans. Software Eng.*, vol. SE-12, pp. 483–495, Mar. 1986.

[12] N. Jagadish, J. Mohan Kumar, and L. M. Patnaik "An efficient scheme for interprocessor communication using dual-ported RAMs," *IEEE Micro*, pp. 10–19, Oct. 1989.

[13] S. Laksmivarahan, and S. K. Dhall, "A new hierachy of hypercube interconnection schemes for parallel computers," *J. Supercornput.*, pp. 81–107, Sept. 1988.

[14] R. Miller, and Q. F. Stout, "Data movement techniques for the pyramid computer," *SIAMJ. Comput.*, vol. 16, no. 1, pp. 38–60, Feb. 1987.

[15] J. Mohan Kumar, L. M. Patnaik, and D. K. Prasad, "A transputer-based Extended Hypercube," *Microprocessing and Microprogramming*, to be published.

[16] F. P. Preparata and J. Vuillemin, "The cube connected cycles: A versatile network for parallel computation," *Commun. ACM*, vol. 24, no. 5, pp. 300–309, May 1981.

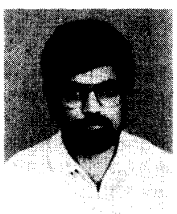
[17] Y. Saad and M. H. Schultz, "Topological properties of hypercubes," Res. Rep. YALEU/DCS/RR-389, Yale Univ., June 1985.

[18] C.L. Sietz, "The Cosmic Cube," *Commun. ACM*, vol. 28, no. 1, pp. 22–33, Jan. 1985.

[19] Q. F. Stout, "Hypercubes and pyramids," NATO ASI Series Vol. F 25, *Pyramidal Systems for Computer Vision*, pp. 75–89.

[20] A. N. Tantawi, "Performance of a hierarchically interconnected multiprocessor," IBM Res. Rep. 13335, IBM Research Division, 1988.

[21] A. S. Wagner, "Embedding trees in the hypercube," Ph.D dissertation, Dep. Comput. Sci., Univ. of Toronto.



J. Mohan Kumar received the B.E degree from Bangalore University, Bangalore, India, in 1982 and the M.Tech. degree from the Indian Institute of Science, Bangalore, India in 1985

Currently he is working toward the Ph.D degree in computer science at the Indian Institute of Science During 1982–1983 and 1985–1986 he was a Lecturer at the Bangalore University Since 1986 he has been a Scientific Officer at the Microprocessor Applications Laboratory, Indian Institute of Science His research interests include multiprocessor systems, neural networks, and parallel computing.

L. M. Patnaik (S'57–M'76–SM'86) received the Ph D degree in 1978 in the area of real-time control software and the D.Sc degree in 1989 in the areas of computer systems and architectures, both from the Indian Institute of Science, Bangalore, India

Currently, he is a Professor with the Department of Computer Science and Automation, and Supercomputer Education and Research Center, and Convenor of the Microprocessor Applications Laboratory, at the Indian Institute of Science, Bangalore, India. During the last two decades of his service at the Institute, his teaching, research, and development interests have been in the areas of parallel and distributed computing, computer architecture, CAD of VLSI systems, computer graphics, theoretical computer science, and neural computing. In these areas, he has published more than 90 papers in refereed International Journals and more than 95 papers in refereed International Conference Proceedings, on the theoretical, software, and hardware aspects of the above areas of computer science and engineering. He is a co-editor of a book on VLSI system design published by the Tata McGraw Hill, co-author of a book on functional programming to be published by Springer-Verlag

Dr. Patnaik is a member the Computer Society of India, a founder Member of the Executive Committee of the Association for the Advancement of Fault-Tolerant and Autonomous Systems, a Fellow of the Indian Academy of Sciences, National Academy of Sciences, the Institution of Electronics and Telecommunications Engineers, the Institution of Engineers, all in India He is a Distinguished Lecturer of the IEEE Region 10. He was awarded the Dr Vikram Sarabhai Research Award for the year 1989 for contributions in the areas of Electronics, Informatics,Telematics, and Automation He has been serving as the Program dnd General Chairs for the International Conference on VLSI Design, member of the Editorial Board of the *International Journal of High Speed Computing*, and *International Journal of Computer Aided VLSI Design*. He is editor of the *Computer Science and Informatics Journal* published by the Computer Society of India