Proceedings of 10th International Conference on Distributed Computing Systems
(ICDCS), May 1990

STRUCTURAL PROPERTIES OF INCOMPLETE HYPERCUBE COMPUTERS

Nian-Feng Tzeng

The Center for Advanced Computer Studies
The University of Southwestern Louisiana
Lafayette, LA 70504, U. S. A.

ABSTRACT

An n-dimensional hypercube can connect exactly 2^n computing nodes only, strongly restricting the possible system sizes and leaving a large gap between two allowable sizes. Consequently, variant hypercube topologies with more flexibility in system sizes, called *incomplete hypercubes,* are appealing and need to be examined. An incomplete hypercube may also come from a complete hypercube which operates in a degraded manner after some nodes become faulty. In this paper, incomplete hypercubes are analyzed for the first time. Structural properties, including diameter, mean message traversal, and traffic density, of incomplete hypercube computers with size 2^n+2^k, $0 \leq k < n$, are investigated and presented. It is interesting to discover that traffic density in such an incomplete hypercube is bounded by 2, despite its structural non-homogeneity. Thus, cube links can easily be constructed so as to avoid any single point of congestion, guaranteeing good performance. With every advantage of complete hypercubes but without a serious restriction on the system size, the incomplete hypercube is an attractive and practical topology.

1. INTRODUCTION

Recently, the binary hypercube has become the most popular message-passing architecture for interconnecting computing nodes [1-4] due to its numerous attractive features, involving a slow (logarithmic) increase in its diameter and degree, structural regularity, simple routing, the existence of rich parallel paths between pairs of nodes, etc. Since it was first introduced in [5], the hypercube structure has been pursued intensively and its many interesting properties have been uncovered [6-11, 14].

A severe restriction on the hypercube topology, however, is that the size of a system has to be a power of 2. It leaves a large gap between two consecutive allowable system sizes. This inspires the consideration of *incomplete hypercubes* where the restriction on the node number is relaxed or totally removed [12]. An incomplete hypercube may also come from a complete hypercube which operates in a degraded manner after some nodes become faulty. Simple and deadlock-free algorithms for routing and for broadcasting messages in the incomplete hypercube with an arbitrary size have been developed by Katseff [12]. They possess characteristics similar to those of the routing algorithms for complete hypercubes described in [6].

No analysis on the incomplete hypercube has ever been carried out so far to unveil its structural properties, which would provide us with a better insight into the structure. In particular, since incomplete hypercube computers are asymmetric in nature and generally an asymmetric structure, such as the tree, star, and any other irregular topology, is likely to have one or several heavy traffic links or nodes that may become vulnerable points with respect to performance and reliability, we are interested in whether an incomplete hypercube exhibits such points of vulnerability. This work is aimed at finding out an answer.

The incomplete hypercube computer under consideration is of size 2^n+2^k, $0 \leq k < n$, i.e., an incomplete hypercube comprising two complete hypercubes. This class of incomplete hypercube computers is particularly appealing and practical because all the nodes in each constituent complete hypercube of such incomplete hypercubes have an identical degree, yielding somewhat regular structures, which ease their packing and setting (note that a hypercube is commonly packed and set in a hierarchical way). Furthermore, incomplete hypercube computers with an arbitrary size may be analyzed in a similar manner since they can be regarded as built from several complete hypercubes with different sizes. An incomplete hypercube with 14 nodes, for example, is composed of three complete hypercubes with sizes 8, 4, and 2, respectively.

Structural properties of such incomplete hypercube computers, including the diameter, mean message traversal, and traffic density, are studied and demonstrated. As will be seen shortly, in any sized incomplete hypercube, the highest traffic density is bounded by 2, despite its non-homogeneity. Cube links can easily be designed to prevent any traffic bottleneck from occurring. This clearly indicates that incomplete hypercubes are superior over other non-homogeneous topologies, such as trees and stars, where points of congestion are likely to exist and serious performance degradation may result because in such a topology, the highest traffic density is proportional to its size. If one wants to put a complete hypercube into operation even after some nodes become faulty, cube links in the complete hypercube can easily be

This work was supported in part by the NSF under Grant MIP-8807761 and by the Louisiana Board of Regents' R&D Program Component under Grant 86-USL(2)-127-03.

Best Paper Award

so designed that every link is still below half saturated in the presence of faults to avoid any single point of congestion, maintaining good performance always.

This paper is organized as follows. Section 2 introduces necessary nomenclature and background to facilitate the subsequent presentation. A brief review on the elementary properties of complete hypercubes and on a routing algorithm for incomplete hypercubes is given in Section 3. Section 4 analyzes incomplete hypercube computers and presents the results. The paper concludes with Section 5.

2. NOMENCLATURE AND BACKGROUND

An n-dimensional complete hypercube, denoted by H_n, comprises 2^n nodes, each with n connection links serving as communication channels directly to n neighbors, called the nearest neighbors. Nodes in H_n are numbered from 0 to 2^n-1, by n-bit binary numbers $(x_{n-1}\cdots x_i\cdots x_0)$ as their *addresses*. A node and a nearest neighbor have exactly one different bit in their addresses. H_n can be viewed as two $(n-1)$-dimensional complete hypercubes, $0H_{n-1}$ and $1H_{n-1}$, such that every node $(0x_{n-2}\cdots x_i\cdots x_0)$ in $0H_{n-1}$ has a link to one and only one node $(1x_{n-2}\cdots x_i\cdots x_0)$ in $1H_{n-1}$. Let $\times^l$ represent l consecutive $\times$'s. Then, the subcubes of H_n, $0H_{n-1}$ and $1H_{n-1}$, can be denoted respectively by $(0*^{n-1})$ and $(1*^{n-1})$, where $*$ is a *don't care* symbol, whose value can be 0 or 1. A d-dimensional subcube in H_n contains exactly d $*$'s in its binary address representation, regardless of their relative positions. For instance, like $(0*^{n-1})$ and $(1*^{n-1})$, $(*0*^{n-2})$ and $(*^i1*^{n-1-i})$ are among $(n-1)$-dimensional subcubes in H_n. Similarly, those m-dimensional subcubes in H_n include $(0^{n-m}*^m)$, $(0101*^m1^{n-m-4})$, etc. Nodes which belong to a subcube can be easily told from the subcube denotation. The 2-dimensional subcube $(0*11*)$ in H_5, for example, contains nodes (00110), (00111), (01110), and (01111).

The incomplete hypercube of interest is composed of two complete hypercubes H_n and H_k $(0\leq k<n)$. It has 2^n+2^k nodes numbered from 0 to 2^n+2^k-1, by $(n+1)$-bit binary numbers $(x_n x_{n-1}\cdots x_i\cdots x_0)$. Nodes in H_n are numbered from 0 to 2^n-1; whereas in H_k, from 2^n to 2^n+2^k-1. Let I_k^n denote the incomplete hypercube. Subcubes in I_k^n can be defined as in a complete hypercube. As an instance, subcubes which are the constituent complete hypercubes of I_k^n are $(0*^n)$ and $(10^{n-k}*^k)$. Fig. 1 shows I_2^3, where each node is numbered by a 4-bit string and the two constituent hypercubes are $(0*^3)$ and $(10*^2)$.

The link between two neighboring nodes A and B is denoted as λ_B^A. To communicate with a remote node, one node can send a message to that remote node by relaying the message through several intermediate nodes. A message traveling from one node to a nearest neighbor is called a *traversal*. Let $D(A,B)$ represent the number of different bits between the addresses of nodes A and B, i.e., their Hamming distance. It is apparent that the number of traversals required from node A to node B is $D(A,B)$. A *path* from node A to node B, denoted by Λ_B^A, is composed of the sequence of links traversed from A to B. For example, the sequence of links: $\lambda_{A_2}^{A_1}$, $\lambda_{A_3}^{A_2}$, and $\lambda_{A_4}^{A_3}$ constitutes path $\Lambda_{A_4}^{A_1}$. A *shortest path* between two nodes A and B comprises only l links if $D(A,B)=l$. In this paper, we are interested in the shortest paths only, and a shortest path is called a path for simplicity. The *relative address* of two nodes is the bitwise Exclusive-OR of their addresses. A link is assumed to have *link number* i if it connects two nodes whose addresses differ only in the i^{th} bit position (starting with the least significant bit as bit 0). The link between nodes (1101) and (0101), for example, is referred to as link 3.

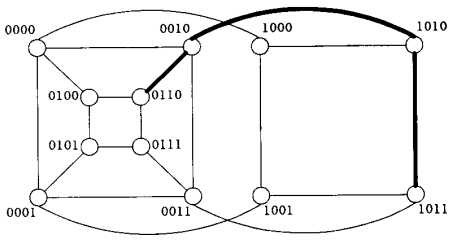

Fig. 1. Incomplete hypercube I_2^3.
(The path from (1011) to (0110) is darkened.)

The maximum number of traversals a message may make between any two nodes in a topology is termed the *diameter* of the topology. It places a lower bound on message transmission latency required to route messages throughout the topology. On the other hand, the mean number of traversals of messages in a topology indicates the transmission latency of a "typical" message. Thus, *mean message traversal* (denoted by T_{mean}) is a fundamental property of a topology. Although mean message traversal probably is a more natural index of latency than the diameter, they both could greatly influence the performance of a topology. Another important property of topologies is *traffic density* over links (denoted by TD), which reflects link utilization. A topology with low traffic density is preferable because it would avoid any potential communication bottleneck and reduce processing/queueing delay when messages visit a node.

While the diameter of a topology is fixed, its mean message traversal and traffic density depend on the message distribution, which describes the probability of message exchanges among different nodes. In this paper, a *uniform* message distribution is assumed, i.e., the average rate at which node A sends messages to node B is the same for all nodes A and B, where $A\neq B$. Besides the diameter, mean message traversal and traffic density in incomplete hypercubes under a uniform message distribution are of interest and will be examined.

As opposed to those in a complete hypercube, nodes in an incomplete hypercube no longer play an identical role. As an instance, every node in subcube (00**) of I_2^3 shown in Fig. 1 has 4 links (while the other nodes have 3

links each) and is connected to a corresponding node in subcube (10**); communication between nodes in subcube (01**) and nodes in subcube (10**) must go through nodes in subcube (00**). Nodes in subcube (00**) are particular and are referred to as *pivot* nodes. Generally, I_k^n has 2^k pivot nodes, which lie in subcube $(0^{n-k+1}*^k)$. There are three different types of links in I_k^n, i.e., links for connecting nodes in the constituent complete hypercube H_n (called L^n); links for connecting nodes in the constituent complete hypercube H_k (called L^k); and links for connecting nodes between pivot nodes and nodes in H_k (called L^c). It is obvious that, in the incomplete hypercube I_k^n, L^n, L^k and L^c consist respectively of $n2^{n-1}$, $k2^{k-1}$ and 2^k links. A link $\in L^n$ and a link $\in L^k$ are the *corresponding links* of each other if they have the same link number. For example, link $\lambda_{(0010)}^{(0000)}$ and link $\lambda_{(1010)}^{(1000)}$ in I_2^3 are the corresponding links (see Fig. 1); so are link $\lambda_{(1011)}^{(1010)}$ and link $\lambda_{(0011)}^{(0010)}$.

Consider the collection of subcubes of I_k^n: $\Phi = \{S_i \mid S_i = (x_n x_{n-1} \cdots x_k *^k)$ such that the value of $x_n x_{n-1} \cdots x_k = i$, $0 \le i \le 2^{n-k}\}$. Apparently, each subcube S_i contains 2^k nodes, whose addresses are from $i2^k$ to $(i+1)2^k - 1$; each node in I_k^n belongs to one and only one subcube. As an instance, the collection of subcubes of I_2^3 shown in Fig. 1 is $\Phi = \{S_0 = (00**), S_1 = (01**), S_2 = (10**)\}$. Notice that node X belongs to S_0 if and only if X is a pivot node in I_k^n. Let the subset of Φ, $\bigcup_{i=1}^{2^{n-k}-1} S_i$, be denoted as Δ. Then, Δ consists of all such nodes in $(0*^n)$ that have no direct links to any node in $(10^{n-k}*^k)$.

An abstract structure of I_k^n with $n-k \ge 1$ is sketched in Fig. 2, where arrows indicate links between subcubes with a number next to an arrow denoting the amount of actual links present. For the case of $n = k$, the circle containing Δ and the arrow incident to it are removed altogether, yielding a simpler figure, which shows the case of H_{n+1}. There are $(n-k)2^k$ links between S_0 and Δ; whereas only 2^k links (which constitute L^c) exist between S_0 and $S_{2^{n-k}}$. It is thus expected that the traffic density over a link $\in L^c$ would be higher.

Before the analysis of incomplete hypercubes is conducted, we briefly review the elementary properties of complete hypercubes and a routing algorithm for incomplete hypercubes for later reference and comparison.

3. PREVIOUS RESULTS

3.1. Properties of Complete Hypercubes

The diameter of a complete hypercube H_n is n because the distance between node $X = (x_{n-1}x_{n-2} \cdots x_0)$ and node $\overline{X} = (\overline{x}_{n-1}\overline{x}_{n-2} \cdots \overline{x}_0)$ in H_n is exactly n.

T_{mean} of H_n can be derived [13] by using the equation

$$T_{mean} = \sum_{l=1}^{n} l \times \phi(l), \tag{1}$$

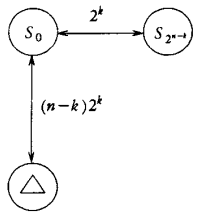

Fig. 2. An abstract structure of I_k^n with $n-k \ge 1$.

where $\phi(l)$ is the probability of an arbitrary message traversing l links. Since the message distribution in H_n is assumed to be uniform, an issued request terminates at every node with an equal probability of $\frac{1}{2^n-1}$. There are $\binom{n}{l}$ nodes whose distance to any given node in H_n is exactly l, where $\binom{n}{l}$ is the combination of n taking l at a time. Thus, the probability of an issued message traversing l links is expressed by

$$\phi(l) = \binom{n}{l} \times \frac{1}{2^n-1}.$$

From Eq. (1), we have

$$T_{mean} = \frac{n2^{n-1}}{2^n-1}. \tag{2}$$

It is clear that T_{mean} approaches $(n/2)$ as n increases for the case of uniform message distributions.

Traffic density on each link of H_n is identical because of symmetry. To calculate TD on a link, consider a time of 2^n-1 units during which each node on an average will have sent a message to each of the others. Equivalently, each node will issue a message heading for an arbitrary other node with probability $\frac{1}{2^n-1}$ during a time unit. Every message behaves statistically equivalently; so it is easy to see that, on an average, the number of total traversals made by all the generated 2^n messages during a time unit is $2^n \times T_{mean}$, which equals $2^n \times \frac{n2^{n-1}}{2^n-1}$. This amount of traversals is shared evenly by $n2^{n-1}$ links, yielding

$$TD = \frac{2^n}{2^n-1}. \tag{3}$$

3.2. Routing in Incomplete Hypercubes

A simple and deadlock-free algorithm for routing messages from one node to another in incomplete hypercubes has been presented in [12]. It always finds a shortest path for messages. The routing algorithm with a minor modification is given below.

Algorithm A: (send or forward a message from node *src* to node *dest* with *tag* ← *src* ⊕ *dest* in an incomplete hypercube):

if ($tag = 0$)
{ send message to local processor. }
else
{ starting with the least significant bit of *tag*:
let i be the bit number of the first 1 in *tag* and link i exists.
send the message on link i and set bit i in *tag* to zero. }

Every issued message carries the relative address of its source node and its destination node as the routing tag. A message is sent through link i only if bit i is the least significant non-zero bit in the tag *and* link i exists. Algorithm A checks the routing tag leftwards, starting from the least significant bit; whereas the routing algorithm given in [12] checks the routing tag in a reverse order. This modification does not alter the characteristics of the original algorithm, but it makes the analysis of traffic density easier, as will be seen later.

To gain a better insight into the algorithm, consider a message issued at node (1011) with tag 13 (= 1101). As can be seen in Fig. 1, the message is first routed through link 0 to (1010), where link 2 does not exist and hence link 3 is taken to forward the message to (0010). Finally, it is routed from (0010) through link 2 to its destination (0110). The length of this chosen path equals 3, which is exactly the number of bit ones in the message tag. It should be noted that Algorithm A is valid actually for any arbitrary incomplete hypercubes [12], not just for incomplete hypercubes I_k^n.

4. ANALYSIS OF INCOMPLETE HYPERCUBES

4.1. Diameter

The diameter of I_k^n can be obtained by finding out the pairs of nodes which are the farthest away from each other. For any two nodes in I_k^n, the number of different bits between their addresses (which reflects the distance between them) is $\leq n+1$. Since the pair of nodes (01^n) and (10^n) always exist and their addresses differ in exactly $n+1$ bits, the diameter of I_k^n is clearly $n+1$.

4.2. Mean Message Traversal

Unlike in a complete hypercube, the mean traversal of messages issued from a node in an incomplete hypercube is not fixed; namely, that issued from one node may be different from that issued from another node. For example, in I_2^3 given in Fig. 1, the mean traversal of messages issued from node (0110) is larger than that issued from node (0000). Due to this asymmetry, it seems unlikely that T_{mean} can be obtained from Eq. (1) directly because finding $\phi(l)$ is fairly difficult. Fortunately, T_{mean} of a topology is also equivalent to the average value of mean traversal of messages issued from all nodes in the topology [14]. We can calculate T_{mean} more easily from this angle.

Let $T_{mean}(X)$ denote the mean traversal of messages issued from node X. Then, T_{mean} of I_k^n is expressed by

$$T_{mean} = \frac{1}{N} \sum_{X \in I_k^n} T_{mean}(X). \tag{4}$$

For any incomplete hypercube I_k^n, we have the following lemma, which enables us to calculate T_{mean} efficiently (consult the appendix for a proof of this lemma).

Lemma 1: $T_{mean}(X) = T_{mean}(Y)$ for all nodes X, $Y \in S_i$, where $S_i \in \Phi$.

The above lemma describes that the mean traversal of messages issued from any node in the same subcube ($\in \Phi$) is the same. For simplicity, let T_{mean}^i represent $T_{mean}(X \in S_i)$. Then, we have Lemma 2 below, which characterizes the mean traversal of messages issued by a node in any subcube. A proof of this lemma is provided in the appendix.

Lemma 2: For any subcube $S_i \in \Phi$,

$$T_{mean}^i = \left(\sum_{l=1}^{n} \binom{n}{l} \times l + \sum_{l=0}^{k} \binom{k}{l} \times (1 + |i| + l) \right) \times p,$$

$$\text{for } 0 \leq i < 2^{n-k}, \tag{5}$$

and

$$T_{mean}^i = \left(\sum_{l=0}^{n} \binom{n}{l} \times (1+l) + \sum_{l=1}^{k} \binom{k}{l} \times l \right) \times p,$$

$$\text{for } i = 2^{n-k}, \tag{6}$$

where $|i|$ is the number of bit ones in the binary representation of i and $p = \frac{1}{2^n + 2^k - 1}$, which is the probability of an issued message terminating at any node (other than the issuing node) under a uniform message distribution.

From Lemmas 1 and 2 together with Eq. (4), we arrive at the following theorem.

Theorem 1: T_{mean} of I_k^n is given by

$$T_{mean} = \frac{2^k \times \left(2^n (2 + n + n 2^{n-k-1}) + k 2^{k-1} \right)}{(2^n + 2^k - 1)(2^n + 2^k)}. \tag{7}$$

Proof: It is obvious from Eq. (4) and Lemma 1 that $T_{mean} = \frac{2^k \times \sum_{i=0}^{2^{n-k}} T_{mean}^i}{2^n + 2^k}$. By utilizing the result of Lemma 2 and following a simple calculation, we are able to obtain Eq. (7) because of $\sum_{l=0}^{n} \binom{n}{l} = 2^n$ and $\sum_{l=0}^{n} \binom{n}{l} \times l = \sum_{l=1}^{n} \binom{n}{l} \times l = \sum_{i=0}^{2^n - 1} |i| = n 2^{n-1}$. □

It is interesting to note that when $k = n$, Eq. (7) is reduced to Eq. (2) with n being replaced by $n+1$, which denotes T_{mean} of H_{n+1}. This is expected because when $k = n$, I_k^n becomes H_{n+1}.

T_{mean} versus k in various incomplete hypercubes is illustrated in Fig. 3. For a fixed n, T_{mean} virtually stays unchanged when k is small. This is understandable because at that time the number of nodes in the constituent complete hypercube, $(10^{n-k} *^k) = H_k$, is much less than 2^n and the majority of communication takes place

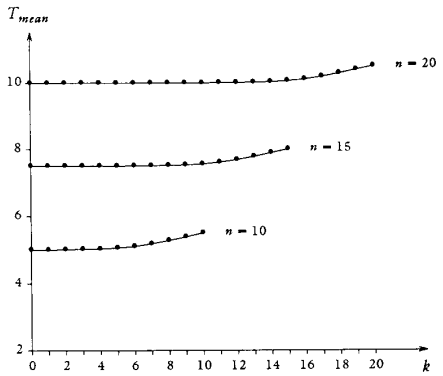

Fig. 3. Mean message traversal (T_{mean}) versus k in I_k^n under a uniform message distribution.

inside H_n, giving rise to T_{mean} near $n/2$ (from Eq. (2)). As k is big enough (roughly, $\geq n-5$), T_{mean} starts to grow noticeably and approaches $(n+1)/2$ as $k = n$. Like in a complete hypercube, T_{mean} of an incomplete hypercube grows only at a logarithmic rate of the network size, a desirable property.

4.3. Traffic Density

A message on an average involves T_{mean} link traversals, and in total there are $(n2^{n-1} + k2^{k-1} + 2^k)$ links serving the $(2^n + 2^k)$ nodes issuing messages in I_k^n; so the average traffic density over all links in the system, TD_a, is

$$TD_a = \frac{2^k \times \left(2^n(2+n+n2^{n-k-1}) + k2^{k-1}\right)}{(2^n+2^k-1)(n2^{n-1}+k2^{k-1}+2^k)}. \quad (8)$$

In fact, traffic density over each link in an incomplete hypercube varies because the number of links connected to a node is not fixed. To obtain actual traffic density over an individual link requires a detailed traffic pattern analysis, which is closely related to the routing algorithm employed. Here, traffic density with respect to Algorithm A is examined. Traffic density with respect to another routing algorithm would be similar and can be examined in the same way.

Since traffic density indicates the average number of messages per link per time unit [7], one can derive traffic density over a link by estimating the average number of messages over the link during a time unit. Traffic density over a link comprises all *traffic density components* due to messages sent from a node A to another node B during a time unit such that Λ_B^A contains the link. Let the traffic density component over link λ_D^C due to messages sent from A to B be denoted by $TDC<\lambda_D^C, \Lambda_B^A>$, then, traffic density over link λ_D^C equals

$$\sum_{\lambda_D^C \subset \Lambda_B^A} TDC<\lambda_D^C, \Lambda_B^A>.$$

Traffic density over links in I_k^n is widely dispersed. Instead of pursuing expressions which govern traffic density over an arbitrary link, we investigate the highest traffic density (TD_h) over links in an incomplete hypercube below. To find out the highest traffic density is important because overall system performance tends to rely upon the highest traffic density, which determines the worst case in communication delay. The subsequent lemmas serve as the basis to arrive at TD_h (for a proof of them, please refer the appendix).

Lemma 3: For an arbitrary node $A_1 = (10^{n-k}a_{k-1}a_{k-2}\cdots a_0) \in S_{2^{n-k}}$, and the pivot node $A_2 = (0^{n+1-k}a_{k-1}a_{k-2}\cdots a_0)$, we have (i) for all $X \in (0*^n)$ and $\lambda_D^C \in L^n$, $TDC<\lambda_D^C, \Lambda_{A_1}^X> = TDC<\lambda_D^C, \Lambda_{A_2}^X>$, (ii) for all $Z = (0z_{n-1}\cdots z_k z_{k-1}z_{k-2}\cdots z_0) \in \Delta$ and $\lambda_D^C \in L^n$, $TDC<\lambda_D^C, \Lambda_Z^{A_1}> = TDC<\lambda_D^C, \Lambda_Z^J>$, where $J = (0^{n+1-k}z_{k-1}z_{k-2}\cdots z_0) \in S_0$, and (iii) for all $X \in (0*^n)$ and $\lambda_F^E \in L^k$, $TDC<\lambda_F^E, \Lambda_X^{A_1}> = TDC<\lambda_D^C, \Lambda_X^{A_2}>$, where λ_D^C is the corresponding link of λ_F^E.

This lemma describes the equality of traffic density components over a link due to messages sent from one node to another node.

Lemma 4: For any link $\lambda_F^E \in L^k$ and its corresponding link λ_D^C, we have

$$\sum_{\substack{X_1 \in S_{2^{n-k}}, \\ Y_1 \in (0*^n)}} TDC<\lambda_F^E, \Lambda_{Y_1}^{X_1}> = \sum_{\substack{X_2, Y_2 \\ \in (0*^n)}} TDC<\lambda_D^C, \Lambda_{Y_2}^{X_2}>.$$

Now, we need to evaluate the expression $\sum_{X_2, Y_2 \in (0*^n)} TDC<\lambda_D^C, \Lambda_{Y_2}^{X_2}>$ for any $\lambda_D^C \in L^n$. Actually, it is the same as traffic density over a link in H_n, except that each node now takes on an average 2^n+2^k-1 (instead of 2^n-1) time units to send a message to each of the other nodes, or equivalently, the probability of an issued message terminating at each of the other nodes is $\frac{1}{2^n+2^k-1}$. Based on Eq. (3), we know that traffic density over link $\lambda_D^C \in L^n$ is equal to $\frac{2^n}{2^n+2^k-1}$.

Lemmas 3 and 4 together with the above result lead to the following important theorem, whose proof is given in the appendix.

Theorem 2: For I_k^n, traffic density over each link in L^n or L^k does not exceed $\frac{2^{n+1}}{2^n+2^k-1}$.

In the following theorem, we prove that traffic density on each link $\in L^c$ is exactly $\frac{2^{n+1}}{2^n+2^k-1}$. Since traffic density on each link $\in L^n$ or L^k is shown to be no higher than $\frac{2^{n+1}}{2^n+2^k-1}$, the highest traffic density in I_k^n is thereby $\frac{2^{n+1}}{2^n+2^k-1}$.

Theorem 3: For I_k^n, traffic density over each link in L^c is $\frac{2^{n+1}}{2^n+2^k-1}$, which equals TD_h.

Proof: Traffic density over a link $\lambda_Q^P \in L^c$ is given by $\sum_{\substack{X_1 \in S_{2^{n-k}}, \\ Y_1 \in (0*^n)}} TDC<\lambda_Q^P, \Lambda_{Y_1}^{X_1}> + \sum_{\substack{X_2 \in (0*^n), \\ Y_2 \in S_{2^{n-k}}}} TDC<\lambda_Q^P, \Lambda_{Y_2}^{X_2}>$.

Based on the facts that (i) there are 2^k nodes in $S_{2^{n-k}}$ and 2^n nodes in $(0*^n)$, (ii) the probability of a message issued by a node in $S_{2^{n-k}}$ to a node in $(0*^n)$ is $\frac{1}{2^n+2^k-1}$, (iii) a message sent from a node in $S_{2^{n-k}}$ to a node in $(0*^n)$ must pass through one of the links in L^c (see Fig. 2), and (iv) all the messages sent from $S_{2^{n-k}}$ to $(0*^n)$ are through the 2^k links in L^c equally likely due to symmetry, the first term is equal to $\frac{2^k \times 2^n}{2^k \times (2^n+2^k-1)} = \frac{2^n}{2^n+2^k-1}$. Similarly, the second term equals $\frac{2^n}{2^n+2^k-1}$.

From Theorem 2 and the above statement, it is clear that the highest traffic density (TD_h) in I_k^n equals $\frac{2^{n+1}}{2^n+2^k-1}$.

□

When $k=n$, TD_h and TD_a agree as expected because I_k^n then becomes H_{n+1}, where each link has the same traffic density, whose value is given by Eq. (3). From this theorem, we know that traffic density over any link in the incomplete hypercube is no larger than 2, irrespective of the system size and despite the fact that the number of links between S_0 and $S_{2^{n-k}}$ is confined to 2^k (see Fig. 2). This is an interesting property. Compared with other asymmetric structures, such as trees or stars, in which certain points (links or nodes) have traffic density as high as $O(N)$ for a size N system [7], the incomplete hypercube clearly is more applicable in practice because a structure with very high traffic density over certain points is likely to suffer from degraded performance and reliability. It is much easier to design a link with enough bandwidth for incomplete hypercubes than for other asymmetric structures so as to keep every link below half saturated and yield a short queueing time (note that the queueing knee is near half saturation). If we want to maintain good performance of a complete hypercube even after some nodes become faulty, cube links can easily be so designed that every link is still below half saturated after faults arise.

Traffic density is plotted as a function of k for $n = 10$ and 20 in Fig. 4, where both TD_h and TD_a are given. It can be seen that as k is small, the number of messages passing through a link with the densest traffic during a time unit is about 2. When k is large enough (roughly, $\geq n-5$), TD_h starts to improve rapidly, and as $k = n$, it eventually reaches the point where every link has the same traffic density, which is close to 1. TD_a, however, always stays pretty much around 1, indicating that the majority of links has only one message traversed each during a time unit.

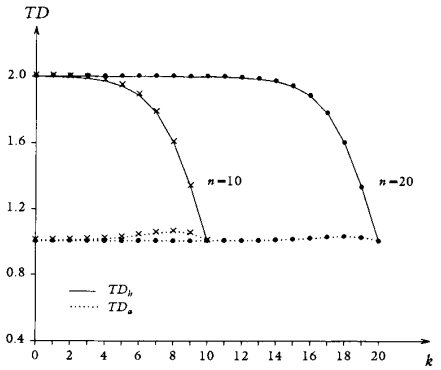

Fig. 4. Traffic density (TD) versus k in I_k^n under a uniform message distribution.

5. CONCLUSIONS

In this paper, we have investigated the properties of incomplete hypercubes for the first time. The incomplete hypercube topology potentially not only allows the construction of systems with arbitrary sizes but also exposes the nature of hypercube systems operating in a gracefully degraded manner. Structural properties, including diameter, mean message traversal, and traffic density, of the incomplete hypercube computer with size 2^n+2^k, $0 \leq k < n$, are analyzed and demonstrated. The analysis may be extended to incomplete hypercubes with any arbitrary size.

It is found that mean message traversal in such an incomplete hypercube increases only at a logarithmic rate of the system size. More importantly, traffic density over links under the uniform message distribution is shown to be bounded by 2, irrespective of the system size and despite its structural non-homogeneity. Compared to other asymmetric topologies, such as trees or stars where the densest traffic over a point (link or node) in a system with size N can be as high as $O(N)$ [7], the incomplete hypercube computer is much less likely to cause any single point of congestion, avoiding any potential degradation in performance and reliability. Cube links can easily be so constructed that every link is below half saturated in order to assure a short mean message queueing time, maintaining good performance. With virtually every advantage of complete hypercubes but without a serious restriction on the system size, the incomplete hypercube is an attractive and practical topology.

APPENDIX

Proof of Lemma 1: Consider the distances from nodes X and Y respectively to all the nodes in S_j (where j may be equal to i). Let X, Y, and any arbitrary node Z in S_j be respectively $(x_n \cdots x_k x_{k-1} \cdots x_0)$, $(y_n \cdots y_k y_{k-1} \cdots y_0)$, and $(z_n \cdots z_k z_{k-1} \cdots z_0)$. Since both nodes X and Y belong to S_i, $x_n \cdots x_k$ equals $y_n \cdots y_k$. So, the number of different bits between $x_n \cdots x_k$ and $z_n \cdots z_k$ is the same as that

between $y_n \cdots y_k$ and $z_n \cdots z_k$ for any node $Z \in S_j$.

Next, let us examine the number of different bits between $x_{k-1} \cdots x_0$ and $z_{k-1} \cdots z_0$ (which is a part of the address of an arbitrary node $Z \in S_j$). Since the number of different bits between $x_{k-1} \cdots x_0$ and $z_{k-1} \cdots z_0$ varies from 0 to k (because the value of $z_{k-1} \cdots z_0$ varies from 0 to 2^k-1), the total of different bits between $x_{k-1} \cdots x_0$ and $z_{k-1} \cdots z_0$ of all nodes Z in S_j amounts to $\sum_{l=0}^{k} \binom{k}{l} \times l$, which can also be shown as the total of different bits between $y_{k-1} \cdots y_0$ and $z_{k-1} \cdots z_0$ of all nodes Z in S_j. From the above statement, it is clear that the total distance to all the nodes in S_j from node X is the same as from node Y. This is true for any $S_j \in \Phi$.

Thus, the total distance to all nodes in I_k^n from node X is identical to that from node Y. This implies that $T_{mean}(X)$ equals $T_{mean}(Y)$ under a uniform message distribution environment.

□

Proof of Lemma 2: Eq. (5) is derived first. Since $0 \leq i < 2^{n-k}$, S_i is in $(0*^n)$, a constituent complete hypercube. The term $\sum_{l=1}^{n} \binom{n}{l} \times l$ in Eq. (5) accounts for the sum of distances from a given node $X \in S_i$ to all the other nodes in $(0*^n)$.

The next term accounts for the sum of distances from the given node X to all the nodes in $(10^{n-k}*^k)$, the other constituent complete hypercube, and can be derived by considering the sum of different bits between the address of node $X = (0x_{n-1} \cdots x_k x_{k-1} \cdots x_0)$ and the address of each node in $(10^{n-k}*^k)$: (1) their most significant bits are always different; (2) for the next $(n-k)$ bits, $x_{n-1} \cdots x_k$ differs from 0^{n-k} in $|i|$ bits, i.e., those bit ones; and (3) for the least significant k bits, $x_{k-1} \cdots x_0$ and $*^k$ may be different from 0 bit up to k bits; and, in fact, the total number of different bits between $x_{k-1} \cdots x_0$ and all the 2^k possibilities of $*^k$ is $\sum_{l=0}^{k} \binom{k}{l} \times l$. By summing up all the cases from X to every node in $(10^{n-k}*^k)$, we have the second term of Eq. (5).

Since a uniform message distribution is assumed, messages issued from node X terminate at any other node equally likely, i.e., with probability p. Consequently, T_{mean}^i is as expressed in Eq. (5).

The proof of Eq. (6) follows the same line as that of Eq. (5). The term $\sum_{l=0}^{n} \binom{n}{l} \times (1+l)$ in Eq. (6) accounts for the sum of distances from a given node $X \in S_{2^{n-k}}$ $(= (10^{n-k}*^k))$ to all the nodes in $(0*^n)$; while the next term, from X to all the other nodes in $S_{2^{n-k}}$.

□

Proof of Lemma 3: Consider any intermediate node Q visited by a message sent from $X = (0x_{n-1} \cdots x_0)$ to A_1. From Algorithm A, we know that the message is forwarded to link i of node Q if bit i is the least significant non-zero bit of the tag and link i exists. Each node in $(0*^n)$ has at least n connected links, i.e., link 0, link 1, $\cdots$, and link $n-1$; so if the message is to be forwarded to link i, all the least significant $(i-1)$ bits of the tag must be zero, which implies $Q = (0x_{n-1} \cdots x_{i+1} x_i a_{i-1} \cdots a_0)$ and $x_i \neq a_i$. After making this traversal, the message arrives at node $(0x_{n-1} \cdots x_{i+1} a_i a_{i-1} \cdots a_0)$. This routing process repeats, and the message eventually will be forwarded to node $(00^{n-k} a_{k-1} a_{k-2} \cdots a_0)$, which is node A_2. From there, the message will be sent through link n of A_2 to A_1. Hence, path $\Lambda_{A_2}^X$ is always a part of path $\Lambda_{A_1}^X$ and a message sent from X to A_1 must pass through A_2. This implies that $TDC < \lambda_D^C, \Lambda_{A_1}^X >$ always equals $TDC < \lambda_D^C, \Lambda_{A_2}^X >$, as stated in (i).

Again, consider a message sent from A_1 to $Z = (0z_{n-1} \cdots z_k z_{k-1} z_{k-2} \cdots z_0) \in \Delta$. In accordance with Algorithm A, the message will be forwarded to node $(10^{n-k} z_{k-1} z_{k-2} \cdots z_0)$. Since each node in $(10^{n-k}*^k)$ has only $k+1$ connected links, i.e., link 0, link 1, $\cdots$, link $k-1$, and link n, the message must then go through link n to enter $(0*^n)$. After making the traversal, the message arrives at node $(00^{n-k} z_{k-1} z_{k-2} \cdots z_0) = J \in S_0$. From there, the message is routed to node Z via the same path as that traversed by a message generated at J with destination Z. This means that $TDC < \lambda_D^C, \Lambda_Z^{A_1} >$ is equal to $TDC < \lambda_D^C, \Lambda_Z^J >$.

To prove (iii), let us first examine how a message M_1 is routed from A_1 to a node $X = (0x_{n-1} \cdots x_k x_{k-1} \cdots x_0) \in (0*^n)$. As stated previously, M_1 reaches node $(10^{n-k} x_{k-1} \cdots x_0)$ after, say, q steps (where q is the number of different bits between $x_{k-1} \cdots x_0$ and $a_{k-1} \cdots a_0$) and then traverses link n to node $(00^{n-k} x_{k-1} \cdots x_0)$. If node $X \in S_0$, M_1 has reached its final destination X; otherwise, M_1 starts to traverse links $\in L^n$ until it finally arrives at X.

Next, let us consider how a message M_2 is routed from A_2 to X. Because the tag of M_2 differs from that of M_1 only in the most significant bit, according to Algorithm A, M_2 will be forwarded to node $(00^{n-k} x_{k-1} \cdots x_0)$ after q steps. If node $X \in S_0$, M_2 has reached its final destination; otherwise, M_2 follows the same path for M_1 from there on. Hence, $TDC < \lambda_F^E, \Lambda_X^{A_1} >$ is equal to $TDC < \lambda_D^C, \Lambda_X^{A_2} >$ for all $\lambda_F^E \in L^k$ and its corresponding link λ_D^C.

□

Proof of Lemma 4: Summing up both sides of the equation given in (iii) of Lemma 3 over all nodes, we have that traffic over a link $\in L^k$ due to messages sent from all nodes $X_1 \in S_{2^{n-k}} = (10^{n-k}*^k)$ to all nodes $Y_1 \in (0*^n)$ equals traffic over its corresponding link due to messages sent from all nodes $X_2 \in S_0$ to all nodes $Y_1 \in (0*^n)$. The latter traffic amount is then proved to be equivalent to the amount of traffic over the same link due to messages sent from all nodes $W \in (0*^n)$ to all nodes $Y_1 \in (0*^n)$ as follows.

Consider all traffic components over link λ_D^C. The two nodes connected by this link, C and D, belong to $(0^{n+1-k}*^k) = S_0$. Let us observe a message sent from a node $Z = (0z_{n-1} \cdots z_k z_{k-1} z_{k-2} \cdots z_0) \in \Delta$ to a node $W = (0^{n+1-k} w_{k-1} w_{k-2} \cdots w_0) \in S_0$. Assuming that the most significant different bit between $(0z_{n-1} \cdots z_k z_{k-1} z_{k-2} \cdots z_0)$ and $(0^{n+1-k} w_{k-1} w_{k-2} \cdots w_0)$ is j, then j is greater than k. If the message passes through λ_D^C, then, it must visit node C (or D) before traversing λ_D^C. According to Algorithm A, however, all intermediate nodes visited by this message must be in subcube $(0^{n+1-j}1*^{j-1})$, which does not contain S_0 where nodes C and D reside. Thus, the message cannot traverse link λ_D^C and contributes nothing to traffic over link λ_D^C. Likewise, it can be proved that messages sent from a node Z to another node $V \in \Delta$ cannot traverse the link. The proof of this lemma is thus completed.

□

Proof of Theorem 2: First, let us consider traffic density over a link λ_D^C in L^n. It is clear that traffic density over link λ_D^C equals $\varsigma = \sum_{X_1, Y_1 \in (0*^n)} TDC < \lambda_D^C, \Lambda_{Y_1}^{X_1} > + \sum_{X_2 \in (0*^n), Y_2 \in S_{2^{n-k}}} TDC < \lambda_D^C, \Lambda_{Y_2}^{X_2} > + \sum_{X_3 \in S_{2^{n-k}}, Y_3 \in (0*^n)} TDC < \lambda_D^C, \Lambda_{Y_3}^{X_3} >$.

From (i) of Lemma 3, we can see that the second term of ς is the same as $\sum_{X_2 \in (0*^n), Y_2 \in S_0} TDC < \lambda_D^C, \Lambda_{Y_2}^{X_2} >$, which in turn equals $\sum_{X_2, Y_2 \in S_0} TDC < \lambda_D^C, \Lambda_{Y_2}^{X_2} > + \sum_{X_2 \in \Delta, Y_2 \in S_0} TDC < \lambda_D^C, \Lambda_{Y_2}^{X_2} >$. If both nodes C and D are in S_0, this term reduces to $\sum_{X_2, Y_2 \in S_0} TDC < \lambda_D^C, \Lambda_{Y_2}^{X_2} >$ because $\sum_{X_2 \in \Delta, Y_2 \in S_0} TDC < \lambda_D^C, \Lambda_{Y_2}^{X_2} >$ equals 0. On the other hand, if either node C or node D does not belong to S_0, this term reduces to $\sum_{X_2 \in \Delta, Y_2 \in S_0} TDC < \lambda_D^C, \Lambda_{Y_2}^{X_2} >$. Likewise, we can see from (ii) of Lemma 3 that the third term of ς is identical to $\sum_{X_3 \in S_0, Y_3 \in \Delta} TDC < \lambda_D^C, \Lambda_{Y_3}^{X_3} >$.

If both nodes C and D are in S_0, the summation of the last two terms of ς is $\sum_{X_2, Y_2 \in S_0} TDC < \lambda_D^C, \Lambda_{Y_2}^{X_2} > + \sum_{X_3 \in S_0, Y_3 \in \Delta} TDC < \lambda_D^C, \Lambda_{Y_3}^{X_3} > = \sum_{X_2 \in S_0, Y_2 \in (0*^n)} TDC < \lambda_D^C, \Lambda_{Y_2}^{X_2} >$, which amounts to $\frac{2^n}{2^n + 2^k - 1}$, as can be found in the proof of Lemma 4. If either node C or node D does not belong to S_0, the summation of the last two terms of ς becomes $\sum_{X_2 \in \Delta, Y_2 \in S_0} TDC < \lambda_D^C, \Lambda_{Y_2}^{X_2} > + \sum_{X_3 \in S_0, Y_3 \in \Delta} TDC < \lambda_D^C, \Lambda_{Y_3}^{X_3} >$, which is surely no larger than $\sum_{X_2, Y_2 \in (0*^n)} TDC < \lambda_D^C, \Lambda_{Y_2}^{X_2} > = \frac{2^n}{2^n + 2^k - 1}$.

In either case, traffic density on λ_D^C given by ς is $\leq 2 \times \frac{2^n}{2^n + 2^k - 1} = \frac{2^{n+1}}{2^n + 2^k - 1}$ because the first term of ς equals $\frac{2^n}{2^n + 2^k - 1}$.

Next, let us examine traffic density over a link λ_F^E in L^k. Traffic density over λ_F^E is $\sum_{X_1, Y_1 \in S_{2^{n-k}}} TDC < \lambda_F^E, \Lambda_{Y_1}^{X_1} > + \sum_{X_2 \in S_{2^{n-k}}, Y_2 \in (0*^n)} TDC < \lambda_F^E, \Lambda_{Y_2}^{X_2} >$.

The first term amounts to $\frac{2^k}{2^n + 2^k - 1}$. From Lemma 4, it is clear that the second term is $\frac{2^n}{2^n + 2^k - 1}$. Since $0 \leq k \leq n-1$, we have traffic density on $\lambda_F^E = \frac{2^k + 2^n}{2^n + 2^k - 1} < \frac{2^{n+1}}{2^n + 2^k - 1}$.

□

REFERENCES

[1] C. L. Seitz, "The Cosmic Cube," *Communications of the ACM,* vol. 28, No. 1, pp. 22-33, Jan. 1985.

[2] Intel Corporation, *A New Direction in Scientific Computing,* Order #28009-001, Intel Corporation, 1985.

[3] W. D. Hillis, *The Connection Machine,* Cambridge, MA: The MIT Press, 1985.

[4] J. P. Hayes, T. N. Mudge, and Q. F. Stout, "Architecture of a Hypercube Supercomputer," *Proc. 1986 Int'l Conf. Parallel Processing,* Aug. 1986, pp. 653-660.

[5] J. Squire and S. M. Palais, "Programming and Design Considerations of a Highly Parallel Computer," *Proc. AFIPS Spring Joint Comput. Conf.,* vol. 23, 1963, pp. 395-400.

[6] H. Sullivan and T. R. Bashkow, "A Large Scale Homogeneous, Fully Distributed Parallel Machine, I," *Proc. 4th Symp. Computer Architecture,* Mar. 1977, pp. 105-117.

[7] L. D. Wittie, "Communication Structures for Large Networks of Microcomputers," *IEEE Trans. Computers,* vol. C-30, pp. 264-273, Apr. 1981.

[8] A. Y. Wu, "Embedding of Tree Networks into Hypercubes," *J. Parallel Distributed Comput.,* vol. 2, pp. 238-249, 1985.

[9] T. F. Chan and Y. Saad, "Multigrid Algorithms on the Hypercube Multiprocessor," *IEEE Trans. Computers,* vol. C-35, pp. 969-977, Nov. 1986.

[10] M.-S. Chen and K. G. Shin, "Processor Allocation in an N-Cube Multiprocessor Using Gray Codes," *IEEE Trans. Computers,* vol. C-36, pp. 1396-1407, Dec. 1987.

[11] Y. Saad and M. H. Schultz, "Topological Properties of Hypercubes," *IEEE Trans. Computers,* vol. C-37, pp. 867-872, July 1988.

[12] H. P. Katseff, "Incomplete Hypercubes," *IEEE Trans. Computers,* vol. C-37, pp. 604-608, May 1988.

[13] D. A. Reed and D. C. Grunwald, "The Performance of Multicomputer Interconnection Networks," *IEEE Computer,* vol. 20, pp. 63-73, June 1987.

[14] D. A. Reed and R. M. Fujimoto, *Multicomputer Networks: Message-Based Parallel Processing,* Cambridge, MA: The MIT Press, 1987.