ELSEVIER

Information Processing Letters 54 (1995) 69–72

Information Processing Letters

Compact embedding of binary trees into hypercubes

Chui-Cheng Chen, Rong-Jaye Chen *

Department of Computer Science and Information Engineering, National Chiao Tung University, 1001 TA Hsueh Road, Hsinchu 30050, Taiwan, ROC

Communicated by R.S. Bird; received 19 October 1994; revised 24 January 1995

Keywords: Embedding; Binary tree; Hypercube

1. Introduction

Over the years, many authors have discussed the embedding of binary trees in hypercubes [2–7]. They studied the one-to-one node embedding of binary trees into hypercubes. Wu [7] has shown that a complete binary tree of height h ($h \geqslant 0$), which has $2^{h+1}-1$ nodes, can be embedded into an $(h+2)$-dimensional hypercube and that the adjacency of the complete binary tree is preserved. In [2,4], it has been proved that a double-rooted complete binary tree of height h is a subgraph of an $(h+1)$-dimensional hypercube. Tzeng et al. [5] have shown that a complete binary tree of height h can be embedded into an incomplete hypercube which comprises an $(h+1)$-dimensional hypercube and an h-dimensional hypercube, and that an incomplete binary tree with $2^{h-1}+2^{h-2}-1$ nodes can be embedded into an h-dimensional hypercube. Wagner [6] described the embedding of a binary tree of height h into an h-dimensional hypercube, which was complete for the first $h-2$ levels.

The objective of this paper is to show how to embed a complete binary tree of height h into an incomplete hypercube of the smallest size and to look in a hypercube for an incomplete binary tree that is larger than the incomplete binary tree in [5]. In Section 2, we describe some preliminaries for embedding. In Section 3, we prove that the complete binary tree can be embedded into an incomplete hypercube, then prove that the size of the incomplete hypercubes is the smallest. In Section 4, we look for an incomplete binary tree in a hypercube.

2. Preliminaries

A complete binary tree of height h, T_h, is a rooted binary tree. The root of the complete binary tree is in level 0, two nodes in level 1, four nodes in level 2, 2^i nodes in level i, etc., and the total number of any complete binary tree of height h is $2^{h+1}-1$. A double-rooted complete binary tree is a complete binary tree with the root replaced by a path of length two [2].

We denote the n-dimensional hypercube with 2^n nodes as H_n. These nodes of H_n are labeled $\{0, 1, \ldots, 2^n-1\}$ as binary number. Two nodes in the hypercube are linked with an edge if and only if their binary numbers differ by a single bit. The *Hamming distance* is the number of different bits

* Corresponding author.

0020-0190/95/$09.50
SSDI 0020-0190(95)00010-0

Fig. 1. Coloring of T_2 and T_3.

between two nodes. If a hypercube misses some certain nodes, it is called an incomplete hypercube [1]. Let $IH(n_1, n_2, \ldots, n_i)$ denote the incomplete hypercube comprising i complete hypercubes: $H_{n_1}, H_{n_2}, \ldots, H_{n_j}, \ldots, H_{n_i}$, $n_j > n_i \geqslant 0$, which can be obtained by deleting the largest $2^{n_1} - (2^{n_2} + \cdots + 2^{n_j})$ nodes (in binary number) and their neighboring edges from an $(n_1 + 1)$-dimensional hypercube.

To conveniently describe the embedding, we use two colors, black and white, to correspond to the binary number of each node. If the binary number contains an even number of 1's, we color the node black. Otherwise, we color the node white. Since the hypercube has a perfect matching, the n-dimensional hypercube has 2^{n-1} black nodes (with an even number of 1's) and 2^{n-1} white nodes (with an odd number of 1's). If T_h is embedded into a hypercube, the nodes of two consecutive levels of T_h have to be mapped to the nodes with different colors in the hypercube. The nodes with the same color as the leaf nodes are more than the nodes with the other color. Without loss of generality, we color the leaf nodes black, their parents white, and so the root black if the height h is even, white if h is odd (see Fig. 1 for coloring of T_2 and T_3).

3. Embedding complete binary trees into incomplete hypercubes

In this section we show how to embed a complete binary tree into an incomplete hypercube of the smallest size. For the embedding, we need the following lemmas.

Lemma 1. *The total number of black nodes on the tree T_h is $(2^{h+2} - 2)/3$ if h is odd, and $(2^{h+2} - 1)/3$ if h is even* [3,6,7].

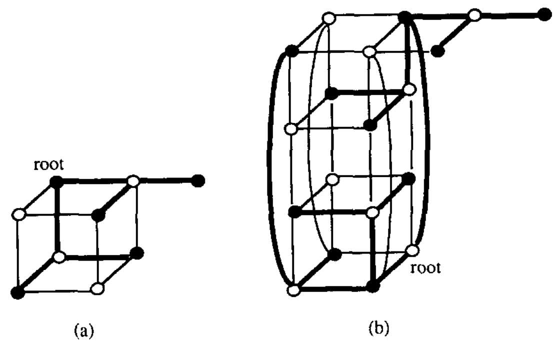

Fig. 2. (a) T_2 is embedded into $IH(3, 0)$. (b) T_3 is embedded into $IH(4, 1, 0)$. (All the embedded tree nodes are linked by solid lines in the incomplete hypercube.)

Lemma 2. *A double-rooted complete binary tree of height h can be embedded into an $(h + 1)$-dimensional hypercube* [2,4].

Now we show that an incomplete hypercube of a specified size can be embedded by T_h.

Theorem 3. *T_h can be embedded into $IH(h + 1, h - 1, h - 3, \ldots, 5, 3, 0)$ if h is even, and $IH(h + 1, h - 1, h - 3, \ldots, 4, 1, 0)$ if h is odd, where $h \geqslant 0$.*

Proof. We will prove the theorem by induction on h.

Hypothesis: T_{h-1} can be embedded into $IH(h, h - 2, \ldots, 5, 3, 0)$ if $h - 1$ is even, and T_{h-1} can be embedded into $IH(h, h - 2, \ldots, 4, 1, 0)$ if $h - 1$ is odd.

Basis step ($h = 0, 1, 2, 3$): When $h = 0$ and 1, T_0 and T_1 can be embedded directly into $IH(0)$ and into $IH(1, 0)$, respectively. Moreover, when $h = 2$ and 3, T_2 and T_3 can be embedded into $IH(3, 0)$ and $IH(4, 1, 0)$ (see Fig. 2).

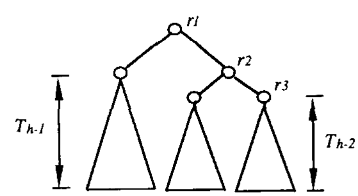

Fig. 3. T_h is partitioned into one T_{h-1} and two T_{h-2}'s; the three subtrees are linked by double roots $r1$ and $r2$.

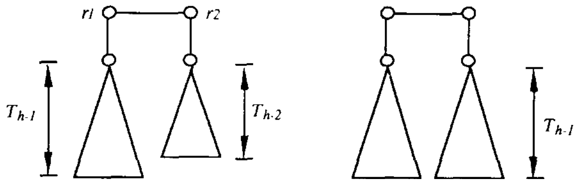

Fig. 4. (a) One T_{h-1} and one T_{h-2} linked by double roots $r1$ and $r2$, contained in a double-rooted complete binary tree of height h as (b).

Induction step: (1) When h is odd, T_h can be partitioned into three subtrees, one T_{h-1} and two T_{h-2}'s as shown in Fig. 3. By Lemma 2, H_{h+1} contains a double-rooted complete binary tree of height h, which contains T_{h-1} and one T_{h-2} using double roots $r1$ and $r2$, as shown in Fig. 4. By hypothesis, T_{h-2} can be embedded into $IH(h-1, h-3, \ldots, 4, 1, 0)$, since $h-2$ is odd. Hence, we can find the other T_{h-2} of T_h in $IH(h+1, h-1, h-3, \ldots, 4, 1, 0)$.

Since any hypercube is symmetric, we can adjust the double-rooted complete binary tree in H_{h+1}. Let the edge $(r2, r3)$ of T_h be mapped to the node $r2$ which is in H_{h+1} and the node $r3$ which is in $IH(h-1, h-3, \ldots, 4, 1, 0)$; that is, the node $r2$ is established at the certain node in H_{h+1}, then the double-rooted complete binary tree can be constructed in H_{h+1} based on the node $r2$. Thus T_h can be embedded into $IH(h+1, h-1, h-3, \ldots, 4, 1, 0)$.

(2) Likewise, when h is even, T_h can be embedded into $IH(h+1, h-1, \ldots, 5, 3, 0)$. □

The compactness of the embedding is proved by the following theorem.

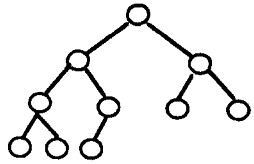

Fig. 5. $IT(2, 3)$.

Theorem 4. *$IH(h+1, h-1, \ldots, 5, 3, 0)$ if h is even, or $IH(h+1, h-1, \ldots, 4, 1, 0)$ if h is odd, is the smallest incomplete hypercube that contains T_h.*

Proof. The cases for $h = 0$, 1, 2 and 3 are trivial. For $h \geqslant 4$, if h is odd, and the total number of black nodes of T_h are $(2^{h+2}-2)/3$. The total number of black nodes in the embedded $IH(h+1, h-1, \ldots, 4, 1, 0)$ is:

$$2^{h+1}/2 + 2^{h-1}/2 + 2^{h-3}/2 + \cdots + 2^4/2 + 2^1/2 + 2^0$$
$$= 2^h + 2^{h-2} + 2^{h-4} + \cdots + 2^3 + 2^0 + 2^0$$
$$= (2^{h+2}-2)/3.$$

So this embedded incomplete hypercube is the smallest.

Similarly, when h is even, $IH(h+1, h-1, h-3, \ldots, 5, 3, 0)$ with $(2^{h+2}-1)/3$ black nodes is the smallest into which T_h can be embedded. □

4. Embedding incomplete binary trees into complete hypercubes

We denote an incomplete binary tree of height $h+1$ as $IT(h, n)$, which is a complete binary tree of height h plus its leftmost n leaf nodes in level $h+1$, where $1 \leqslant n < 2^{h+1}$ (see Fig. 5 for $IT(2, 3)$).

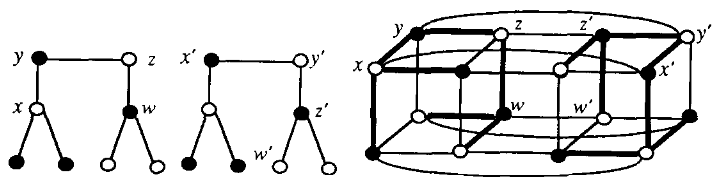

Fig. 6.

Tzeng et al. [6] have shown that $IT(h-2, 2^{h-2})$ can be embedded into H_h. In this section, we present an embedding algorithm in Theorem 6 to find an incomplete binary tree in a hypercube, which is larger than the incomplete binary tree in [6]. To prove this embedding theorem, we need the following lemma.

Lemma 5. *Two double-rooted complete binary trees of height* 2 *in Fig.* 6(a) *can be embedded into* H_4 *as shown in Fig.* 6(b), *where both the embedded trees are linked by solid lines.*

Theorem 6. $IT(h-2, 2^{h-2}+2^{h-4})$ *with* $2^{h-1}+2^{h-2}+2^{h-4}-1$ *nodes can be embedded into an h-dimensional hypercube, where* $h \geqslant 4$.

Proof. First, H_h can be partitioned into two H_{h-1}'s which are denoted H_r and H_l respectively according to the most significant bit. Each node of H_r has an edge to link a certain node of H_l. Applying Lemma 5 by letting $x = n_{l3}$, $y = n_{l1}$, $z = n_{l2}$, $w = n_{l4}$, $x' = n_{r1}$, $y' = n_{r2}$, $z' = n_{r4}$ and $w' = n_{r5}$, respectively, and based on the symmetry of the hypercube, two double-rooted complete binary trees of height $h-2$ can be embedded into H_h as shown in Fig. 7 (The solid lines depict the edges which link n_{l3}, n_{l1}, n_{l2} and n_{l4} to n_{r1}, n_{r2}, n_{r4} and n_{r5}, respectively).

We let A, B and C be the complete binary subtrees of height $h-3$ rooted by n_{l3}, n_{l4} and n_{r3}, respectively, B_1 be the complete binary subtree of height $h-4$ rooted by n_{l4}, D_1 and D_2 be the complete binary subtrees of height $h-5$ rooted by n_{r7} and n_{r8}, respectively, and E be the complete binary subtree of height $h-4$ rooted by n_{r6}. The incomplete binary tree constructed as shown in Fig. 8 can be embedded into H_h.

In addition, as there are $h-1$ levels from root n_{r2} to either the leaves of its left subtree or the leaves of B_1, and there are also $h-2$ levels from root n_{r2} to either the leaves of D_1 or the leaves of E, the height of the incomplete binary tree rooted by n_{r2} is $h-1$. So, the embedded incomplete binary tree is $IT(h-2, 2^{h-2}+2^{h-4})$. □

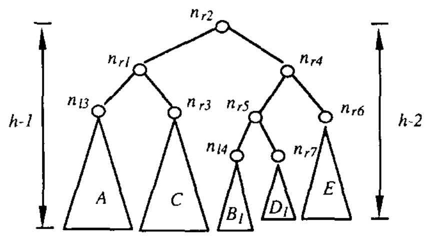

Fig. 8. The embedded incomplete binary tree $IT(h-2, 2^{h-2}+2^{h-4})$.

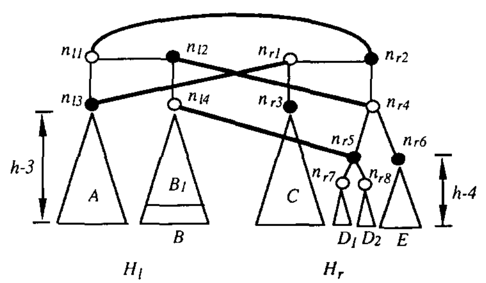

Fig. 7.

References

[1] H.P. Katseff, Incomplete hypercubes, *IEEE Trans. Comput.* **37** (5) (1988) 604–608.

[2] T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes* (Morgan Kaufmann, Reading, MA, 1992) 406–408.

[3] E.L. Leiss and H.N. Reddy, Embedding complete binary trees into hypercubes, *Inform. Process. Lett.* **38** (1991) 197–199.

[4] L. Nebesky, On cubes and dichotomic Trees, *Časopis Pěst. Mat.* **99** (1974) 164–167.

[5] N.F. Tzeng, H.L. Cheng and P.J. Chuang, Embeddings in incomplete hypercubes, in: *Proc. Internat. Conf. on Parallel Processing* 3 (1990) 335–339.

[6] A.S. Wagner, Embedding the complete tree in the Hypercube, *J. Parallel Distributed Comput.* **20** (1994) 241–247.

[7] A.Y. Wu, Embedding of tree networks into hypercubes, *J. Parallel Distributed Comput.* **2** (1985) 238–249.