The paper is intended for the Workshop on Algorithms and Computational Molecular Biology

Embedding Quadtrees into Hypercubes

Chui-Cheng Chen
Department of Information Management
Southern Taiwan University of Technology
1 NanTai St., YungKang City Tainan, Taiwan 710, R.O.C.
Tel: +886 6 2533131#4311
E-mail: ccchen@mail.stut.edu.tw

Abstract

The quadtree is useful data structure for a variety of image processing. In this paper, we propose two simple but effective methods for embedding quadtrees into hypercubes. First, we embed a complete quadtree of height h into a (3(h-1)+4)-dimensional hypercube, or into a smaller incomplete hypercube which comprises a (3(h-1)+3)-dimensional hypercube and a (3(h-2)+4)-dimensional hypercube. This embedding preserves the adjacency of the complete quadtree, while the second method does not. The second method is to embed a complete quadtree of height h into an incomplete hypercube of the same node size with the *congestion* 2 and the *dilation* is at most 3.

Keywords: Embedding, Complete quadtrees, Hypercubes

Embedding Quadtrees into Hypercubes

Abstract

The quadtree is useful data structure for a variety of image processing. In this paper, we propose two simple but effective methods for embedding quadtrees into hypercubes. First, we embed a complete quadtree of height h into a $(3(h-1)+4)$-dimensional hypercube, or into a smaller incomplete hypercube which comprises a $(3(h-1)+3)$-dimensional hypercube and a $(3(h-2)+4)$-dimensional hypercube. This embedding preserves the adjacency of the complete quadtree, while the second method does not. The second method is to embed a complete quadtree of height h into an incomplete hypercube of the same node size with the *congestion* 2 and the *dilation* is at most 3.

Keywords: Embedding, Complete quadtrees, Hypercubes

1. Introduction

The hypercube is one of the most popular architectures of parallel machines. The structure of the hypercube presents a rich interconnection topology, a symmetric structure, and a low diameter. It can simulate many computational structures with only small constant factor slowdown, such as array, binary tree and mesh of tree [1]. It also contains many computational structures, such as meshes and rings. While two consecutive dimensional hypercubes leave a large gap. To overcome this restriction, incomplete hypercubes provide more flexibility in the size [2, 3]. An incomplete hypercube can be obtained from a complete hypercube where some nodes/links fail.

Over the years, many algorithms have been designed to embed quadtrees into a hypercube [4-8]. The quadtree is an efficient data structure to represent binary image data [9]. The root of the quadtree represents the entire structure to represent binary image data. The root of the quadtree represents the entire image data, and each internal node has four sons, each son representing a quadrant of its parent node. Since the structure of the quadtree is easy to implement, it is a very useful data structure for a variety of image processing.

Ho and Johnsson [4] have shown a complete quadtree of height h ($h \geqq 0$), which has $(4^{h+1}-1)/3$ nodes, can be embedded into a $(2h+1)$-dimensional hypercube with *dilation* 2, and a specific algorithm to do the embedding has been studied in [5]. Stout [6] has described how to embed a complete quadtree into a hypercube. Yang and Lee

[7] have introduced an efficient algorithm to construct a quadtree in a hypercube. In this paper, first, we present how to embed a complete quadtree into a hypercube, or into a smaller incomplete hypercube, so that the adjacency of the complete quadtree is preserved. Next, we present an algorithm to embed a complete quadtree into an incomplete hypercube with *dilation* 3, *congestion* 2 and *expansion* 1.

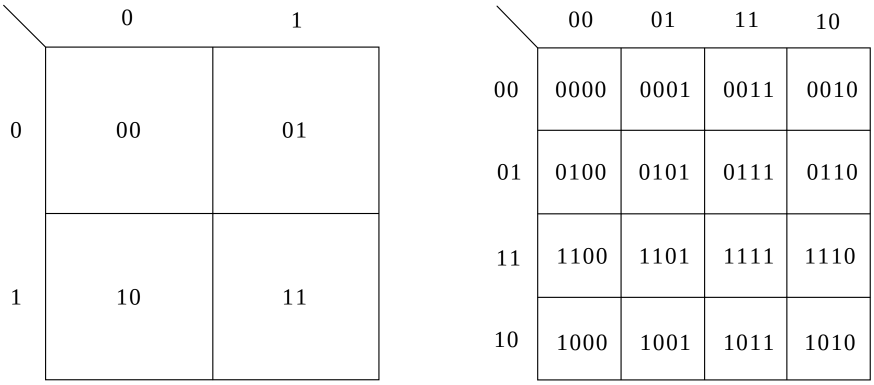
The remainder of this paper is organized as follows. In Section 2, we introduce the notations and definitions for embedding, In Section 3, we present an algorithm to embed a complete quadtree into a hypercube, or into a smaller incomplete hypercube. In Section 4, an algorithm is given to embed a complete quadtree into an incomplete hypercube with the same node size. In Section 5, we summarize the results.

2. Preliminaries

A complete quadtree of height h is a rooted quadtree. The root the complete quadtree is on level 0, four nodes on level 1, 4^i nodes on level i, *etc.*, and we let QT_h denote the complete quadtree of height h with $(4^{h+1}-1)/3$ nodes.

We denote the n-dimensional hypercube with 2^n nodes as H_n. These nodes of H_n are labeled $\{0, 1, \ldots, 2^n-1\}$ with binary number. Two nodes in the hypercube are linked with an edge if their binary numbers differ by a single bit. The *Hamming distance* is the number of different bits between two nodes. If a hypercube misses some certain nodes, it is called an incomplete hypercube [2, 3]. Let $IH(n1, n2, \ldots, nk)$ denote the incomplete hypercube which comprises k complete hypercubes: H_{n1}, H_{n2}, …, H_{nk}, where $nj>ni\geqq 0$, for $j<i\leqq k$.

To conveniently describe the embedding, we can partition a hypercube into four sub-hypercubes by the leftmost two bits of the hypercube, and the binary numbers of the leftmost two bits of the four sub-hypercubes correspond to 00, 01, 11 and 10 (see Figure 1 (a)). Each node of a sub-hypercube has an edge to link a node of adjacent sub-hypercube; the leftmost and the rightmost sub-hypercubes are adjacent, likewise, the top and the bottom sub-hypercubes are adjacent. Similarly, a hypercube can be partitioned into sixteen sub-hypercubes by the leftmost four bits of the hypercube, and the binary numbers of the leftmost four bits of the sixteen sub-hypercubes correspond to 0000, 0001, …, 1111 (see Figure 1(b)). Each node of a sub-hypercube has an edge to link a node of adjacent sub-hypercube.



(a) (b)

Figure 1. A hypercube is partitioned to four or sixteen sub-hypercubes.

In one-to-one node embedding of a graph *G* into a graph *H*, the dilation of an edge in *G* is the length of embedded path in *H*. The dilation of an embedding is the maximum dilation over all edges in *G*. The congestion of an edge in *H* is the number of edges of *G* that are embedded using the same edge of *H*. The congestion of an embedding is the maximum congestion over all edges in *H*. The expansion of an embedding is the ratio of the number of node in *H* to the number of nodes in *G*. Hence, it has to be considered the tradeoff among the *dilation*, the *congestion* and the *expansion* of an embedding.

3. Embedding complete quadtrees into hypercubes with *dilation* 1

In this section we show how to embed QT_h into a hypercube, or into a smaller incomplete hypercube, while the adjacency of QT_h is preserved.

Theorem 1. QT_h ($h \geqq 1$) can be embedded into a (3(h-1)+4)-dimensional hypercube.

Proof. We prove the theorem by induction on h.

Hypothesis: QT_{h-1} can be embedded into a (3(h-2)+4)-dimensional hypercube.

Basis step (h=1, 2): When h=1, QT_1 can be embedded directly into H_4 as show in Figure 2. When h=2, we partition H_7 into 16 sub-hypercubes H_3's. Since a tritree of height 1 can be embedded into H_3 as shown in Figure 3, and each node of sub-hypercube H_3 has an edge to link a node of another adjacent H_3, we can embed QT_2 into H_7 (see Figure 4).

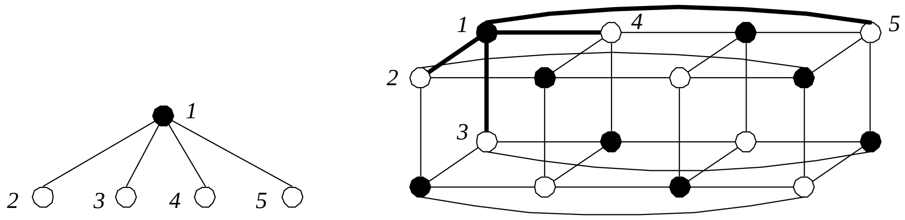


Figure 2. QT_1 is embedded into H_4. The embedded QT_1 is depicted by the solid lines in H_4.

Figure 3. A tritree of height 1 is embedded into H_3.

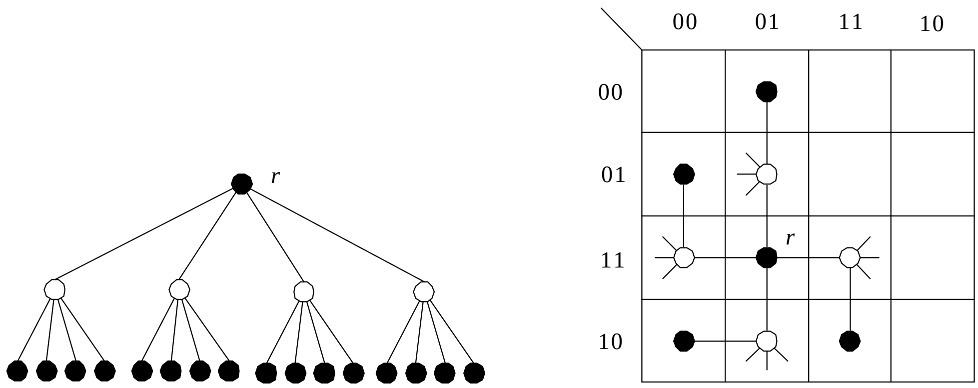


Figure 4. QT_2 is embedded into H_7.

Induction step: we denote $QT_h^{3/4}$ as the induced graph of QT_h by deleting a QT_{h-1} from itself (see Figure 5). Then we can construct $QT_{h-1}^{3/4}$ in a ((3(h-2)+4-1)-dimensional hypercube. Moreover, we partition the (3(h-1)+4)-dimensional hypercube into 16 sub-hypercubes $H_{(3(h-2)+3)}$'s by the leftmost four bits of the hypercube. Each $H_{(3(h-2)+3)}$ contains a $QT_{h-1}^{3/4}$. Now, we embed QT_h into the (3(h-1)+4)-dimensional hypercube as Figure 6 shows. Let the root of QT_h be embedded into sub-hypercube R, and two adjacent sub-hypercubes A and E embed QT_{h-1} by linking the root of $QT_{h-1}^{3/4}$ in sub-hypercube A to a subtree QT_{h-2} of $QT_{h-1}^{3/4}$ in sub-hypercube E. Likewise, two adjacent sub-hypercubes B and F (C and G, D and H) can embed QT_{h-1}. The embedding works because of the symmetry of the hypercube. Thus QT_h can be embedded into the (3(h-1)+4)-dimensional hypercube. □

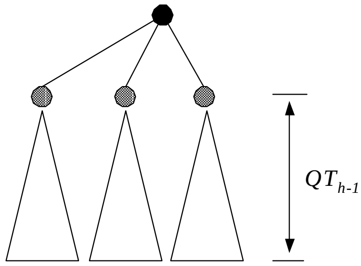


Figure 5. $QT_h^{3/4}$.

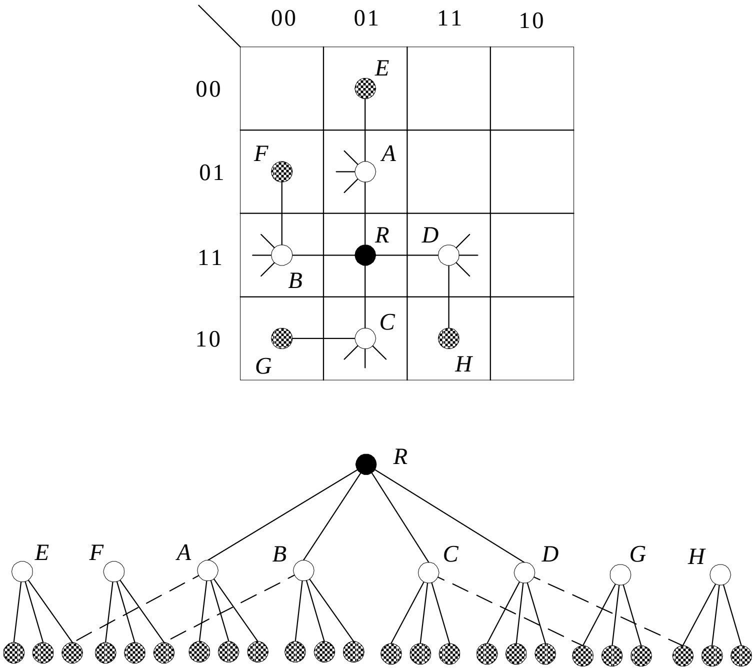


Figure 6. QT_h is embedded into a (3(h-1)+4)-dimensional hypercube, where each represents a QT_{h-2}.

By the preceding construction, the embedded complete hypercube $H_{3(h-1)+4}$ can be reduced to a smaller incomplete hypercube.

Corollary 1. QT_h ($h \geqq 1$) can be embedded into IH(3(h-1)+3, 3(h-2)+4). □

4. Embedding quadtrees into incomplete hypercubes with *expansion* 1

We have show that a complete quadtree can be embedded into hypercube, or into a smaller incomplete hypercube, with *dilation* 1 and *congestion* 1 in the previous section. In this section we discuss how to embed a complete quadtree into an incomplete quadtree into an incomplete hypercube with the same node size (*expansion* 1), considering the *dilation* and the *congestion* when doing the embedding.

Here, denote $IH(n1, n2, \ldots, nk)$ as an incomplete hypercube, which can be obtained by deleting the largest 2^{n1}-$(2^{n2} + \ldots + 2^{nk})$ nodes (in binary number) and their neighboring edges from an ($n1$+1)-dimensional hypercube.

Theorem 2. QT_h ($h \geqq 1$) can be embedded into $IH(2h, 2(h\text{-}1), \ldots, 2, 0)$ with *dilation* 3, *congestion* 2 and *expansion* 1.

Proof. We prove the theorem by induction on h.

Stronger Hypothesis: QT_{h-1} can be embedded into $IH(2(h\text{-}1), 2(h\text{-}2), \ldots, 2, 0)$ with *dilation* 3, *congestion* 2 and *expansion* 1, and the *dilation* is equal to 2 for embedding the edges between level h-2 and level h-1 of QT_{h-1}.

Basis step (h=1 and h=2): When h=1, we can embed QT_1 into $IH(2, 0)$ with *dilation* 2 and *congestion* 2 (see Figure 7), since the dilation of edge (a, d) in QT_1 is 2 and the congestion of either edge (a, b) or edge (a, c) is 2 in $IH(2, 0)$. Thus the *dilation* is equal to 2 for embedding the edges between level 0 and level 1.

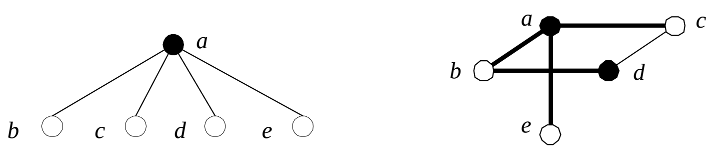


Figure 7. QT_1 is embedded into $IH(2, 0)$ with *dilation* 2 and *congestion* 2.

When h=2, there are 4^2 leaf nodes, so we have to add H_4 to $IH(2, 0)$. H_4 can be partitioned into four sub-hypercubes by the leftmost two bits of H_4, hence, the binary numbers of the leftmost two bits of the four sub-hypercubes correspond to 00, 01, 11 and 10. We can construct four H_2's from these four sub-hypercubes to make the binary numbers of the rightmost two bits of each H_2 the same. We use Figure 8 to describe how to embed QT_2 into $IH(4, 2, 0)$.

Figure 8 illustrates the embedding of QT_2 into $IH(4, 2, 0)$. The four H_2's comprise respectively the nodes (b, g, h, i), (u, e, s, t), (c, k, l, m) and (d, o, p, q). Each leaf node of QT_1 in $IH(2, 0)$ has an edge to link a node of an H_2, such as leaf nodes f, j, n and r linking respectively nodes b, c, d and e. We take leaf nodes f, j, n and r of QT_1 in $IH(2, 0)$ as leaf nodes of QT_2, nodes b, c, d and e as the parents of the nodes (f, g, h, i), (j, k, l, m), (n, o, p, q) and (r, s, t, u). The congestion of the edges which link $IH(2, 0)$ and H_4, such as edges (b, f), (c, j), (d, n) and (e, r), is 2. The dilation of edges between level 0 and level 1 increases by 1, compared to the dilation of the edges between level 0 and level 1 for embedding QT_1 into *IH*(2, 0), such as the dilation of edge (*a*, *b*), (*a*, *c*), (*a*, *d*) and (*a*, *e*) become 2, 2, 3 and 2, respectively. Therefore, QT_2 can be embedded into *IH*(4, 2, 0) with *dilation* 3, *congestion* 2 and *expansion* 1, and the *dilation* is equal to 2 for embedding the edges between level 1 and level 2 of QT_2.

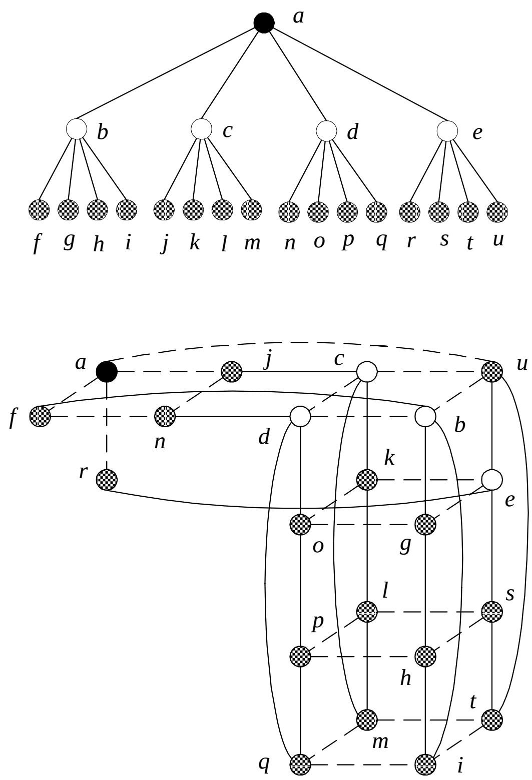


Figure 8. QT_2 is embedded into *IH*(4, 2, 0). Four H_2's, depicted respectively by solid lines, are constructed from H_4 in *IH*(4, 2, 0).

Induction step: There are 4^h nodes which are added to QT_{h-1} as the leaf nodes of QT_h, thus we have to add H_{2h} to *IH*(2(*h*-1), …, 2, 0). Likewise, using the approach of basis step, we partition H_{2h} into four sub-hypercubes by the leftmost two bits of H_{2h} to construct $2^{2(h-1)}$ H_2's, and the binary numbers of the rightmost 2(*h*-1) bits of each H_2 are the same. Each leaf node of QT_{h-1} in *IH*(2(*h*-1), …, 2, 0) has an edge to link a node of an H_2 in H_{2h}, since leaf nodes of QT_{h-1} are embedded into either the nodes of $H_{2(h-1)}$ in *IH*(2(*h*-1), 2(*h*-2), …, 2, 0) or the adjacent nodes of $H_{2(h-1)}$ in *IH*(2(*h*-2), …, 2, 0), and the binary numbers of the rightmost 2(*h*-1) bits of both the adjacent nodes are the same. We take leaf nodes of QT_{h-1} as leaf nodes of QT_h and the nodes, which are adjacent with leaf nodes of QT_{h-1}, of H_{2h} as the parents of leaf nodes of QT_h. We use Figure 9 to show how to embed QT_h into *IH*(2*h*, 2(*h*-1), …, 2, 0).

Figure 9 illustrates the embedding of QT_h into $IH(2h, 2(h\text{-}1), \ldots, 2, 0)$. We label the nodes of $IH(2h, 2(h\text{-}1), \ldots, 2, 0)$ by $2h+1$ bits and partition $IH(2h, 2(h\text{-}1), \ldots, 2, 0)$ into eight sub-hypercubes by the leftmost three bits of the incomplete hypercube. Hence, the eight sub-hypercubes can be labeled as 000, 001, 011, 010, 100, 101, 111 and 110. Let *n1*, *n2*, *n3* and *n4* construct H_2 from these four sub-hypercubes 101, 100, 110 and 111 in H_{2h}. If a leaf node of QT_{h-1} is embedded into *n5* in the sub-hypercube 001, we take *n5* as leaf node of QT_h and *n1* as the parent of *n5, n2, n3* and *n4*. Similarly, if a leaf node of QT_{h-1} is embedded into n6 in the sub-hypercube 000, *n6* has an edge to link *n2* because of the symmetry of the hypercube. We take *n6* as leaf node of QT_h and *n2* as the parent of *n6, n1, n4* and *n3*. By using the same method, we take the remaining leaf nodes of QT_{h-1} as leaf nodes of QT_h and their adjacent nodes in H_{2h} as the parents of leaf nodes of QT_h.

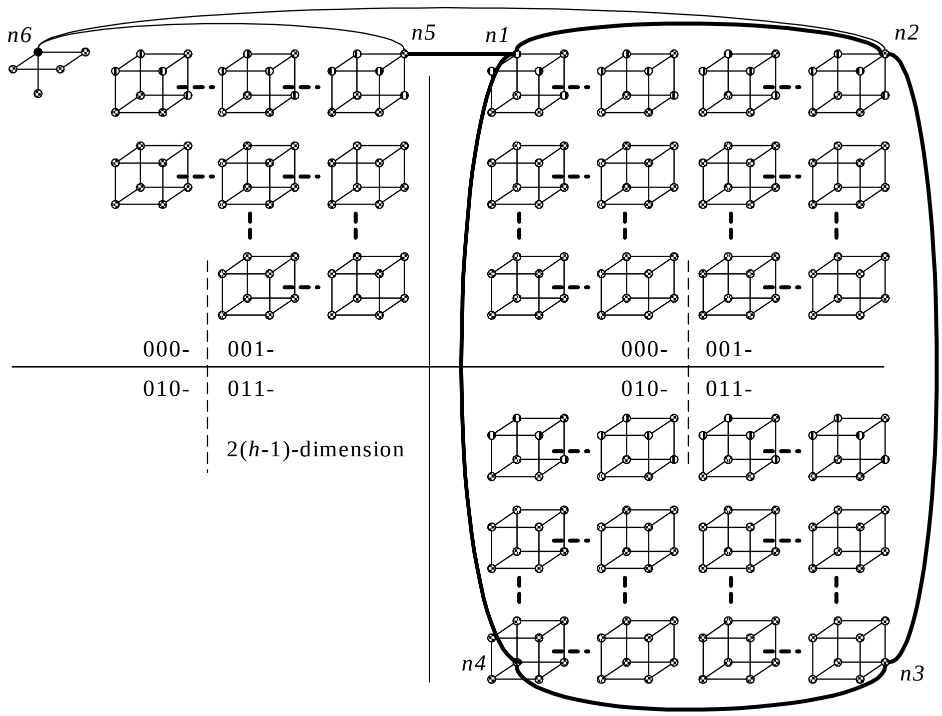


Figure 9. Embedding QT_h into $IH(2h, 2(h\text{-}1), \ldots, 2, 0)$. The sub-hypercube H_{2h} is partitioned into four sub-hypercubes $H_{2(h-1)}$'s, and the bold lines depict an H_2 from these four $H_{2(h-1)}$'s.

The dilation of the edges between level h-2 and level h-1 of QT_h increases by 1, compared to the dilation of the edges between level h-2 and level h-1 for embedding QT_{h-1} into $H(2(h\text{-}1), \ldots, 2, 0)$, while the congestion of the edges between $IH(2(h\text{-}1), \ldots, 2, 0)$ and H_{2h} is held on 2. Therefore, QT_h can be embedded into $IH(2h, 2(h\text{-}1), \ldots, 2, 0)$ with *dilation* 3, *congestion* 2 and *expansion* 1, and *dilation* is equal to

2 for embedding the edges between level h-1 and level h of QT_h. □

5. Conclusions

We have presented two simple but effective algorithms for embedding quadtrees into hypercubes. First, we have shown hat a complete quadtree of height h ($h \geqq 1$) can be embedded into a (3(h-1)+4)-dimensional hypercube, or into a smaller incomplete hypercube IH(3(h-1)+3, 3(h-2)+4), so that the adjacency of the complete quadtree is preserved. Then a complete quadtree of height h can be embedded into an incomplete hypercube IH(2h, 2(h-1), …,2 ,0) with *expansion* 1, *congestion* 2 and *dilation* 3.

Reference

1. T. Leighton, *Introduction to parallel Algorithms and Architectures: Array, Tree, Hypercubes* (Morgan Kaufmann, Reading, MA,1992).
2. H. P. Katseff, Incomplete Hypercubes, *IEEE Trans Computer* 37 (1988) 604-608.
3. N. F. Tzeng, H. L. Cheng and P. J. Chuang, Embeddings in Incomplete Hypercube, *Proc. of Int. Conf. on Parallel Processing 3* (1990) 335-339.
4. C. T. Ho and S. L. Johnsson, Dilation d Embedding of a Hyper-Pyramid into a Hypercube, *Proc. of the Supercomputing*'89 (1989) 294-303.
5. N. Krishnakuma, V. G. Hegde and S. S. Iyengar, Fault Tolerant Based Embedding of Quadtrees into Hypercubes, *Proc. of Int. Conf. on Parallel Processing* 3 (1991) 244-249.
6. Q. F. Stout, Hypercubes and Pyramids, *Pyramidal Systems for Computer Vision*, (Springer-Verlag, 1986) 75-89.
7. S. N. Yang and R. R. Lee, *IEEE Region 10 Conference, Tencon 92 11th-13th* (1992) 131-135.
8. F. Dehne, A. G. Ferreira and A. Rau-Chaplin, Efficient Parallel Construction and Manipulation of Quadtrees, *Proc. of Int. Conf. on Parallel Processing* 3 (1991) 255-262.
9. H. Samet, The Quadtree and Related Hierarchical Data Structures, *Computing Surveys* 16 (1984) 187-260.