Original citation:
Ravindran, S. and Gibbons, A. M. (1992) Dense edge-disjoint embedding of complete binary trees in the hypercube. University of Warwick. Department of Computer Science. (Department of Computer Science Research Report). (Unpublished) CS-RR-223

Permanent WRAP url:
http://wrap.warwick.ac.uk/60912

Copyright and reuse:
The Warwick Research Archive Portal (WRAP) makes this work by researchers of the University of Warwick available open access under the following conditions. Copyright © and all moral rights to the version of the paper presented here belong to the individual author(s) and/or other copyright owners. To the extent reasonable and practicable the material made available in WRAP has been checked for eligibility before being made available.

Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.

A note on versions:
The version presented in WRAP is the published version or, version of record, and may be cited as it appears here.For more information, please contact the WRAP Team at: publications@warwick.ac.uk

http://wrap.warwick.ac.uk/

Research Report 223

Dense Edge-Disjoint Embedding of Complete Binary Trees in the Hypercube

Somasundaram Ravindran and Alan Gibbons

RR223

We show that the complete binary tree with $n > 8$ leaves can be embedded in the hypercube with n nodes such that: paths of the tree are mapped onto edge-disjoint paths of the hypercube, at most two tree nodes (one of which is a leaf) are mapped onto each hypercube node, and the maximum distance from a leaf to the root of the tree is $\log_2 n + 1$ hypercube edges (which is optimally short). This embedding facilitates efficient implementation of many P-RAM algorithms on the hypercube.

keywords: parallel algorithms, graph embedding, binary tree, hypercube

Department of Computer Science
University of Warwick
Coventry CV4 7AL
United Kingdom

February 1992

Dense Edge-Disjoint Embedding of Complete Binary Trees in the Hypercube *

Somasundaram Ravindran and Alan Gibbons
Department of Computer Science, University of Warwick, Coventry CV4 7AL, UK.

Abstract

We show that the complete binary tree with $n > 8$ leaves can be embedded in the hypercube with n nodes such that: paths of the tree are mapped onto edge-disjoint paths of the hypercube, at most two tree nodes (one of which is a leaf) are mapped onto each hypercube node, and the maximum distance from a leaf to the root of the tree is $\log_2 n + 1$ hypercube edges (which is optimally short). This embedding facilitates efficient implementation of many P-RAM algorithms on the hypercube.

keywords: parallel algorithms, graph embedding, binary tree, hypercube

1. Introduction

When designing algorithms for feasible (distributed memory) models of parallel computation it is natural to drawn upon the rich literature that has been developed for the (shared memory) P-RAM model over the last decade or so (see [GR], for example). It is well-known [Va], for example, that the P-RAM can be emulated by the hypercube in such a way that the processor-time product (the *work* measure) is preserved although the computation is slowed down by a factor of $O(\log n)$ where n is the problem size. For a large class of computations (or sub-computations) however, this logarithmic loss can be avoided by embedding the P-RAM algorithmic structure in the hypercube. If the algorithmic structure is frequently employed, then an embedding strategy could be usefully automated. Perhaps the most commonly occurring structure in this regard is the complete binary tree. It is precisely because such logarithmic depth structures are used (either explicitly or implicitly) that polylogarithmic time complexities are attained for many P-RAM algorithms. Thus many problems are placed in the complexity class NC, which is the class of efficiently solvable problems in this model.

In the P-RAM model, the complete binary tree is most usually employed as follows. Data for a problem (or sub-problem) are placed at the leaves, and the required result is obtained by performing computations at the internal nodes in one or more sweeps up and down the tree, so that computations at the same depth are performed in parallel. It should be noted however that some algorithms may require simultaneous computation at an arbitrary number of nodes at different depths of the tree. If we are to embed the complete binary tree into some host

*This work was partially supported by the ESPRIT II Basic Research Actions Programme of the EC under contract No.7141 (Project ALCOM II).

topology of a distributed memory machine, we therefore need to observe the following requirements to achieve an *efficient* embedding:

1. *All tree nodes at the same depth should be mapped into disjoint hypercube nodes if (as in the P-RAM computation) computations are to be performed in parallel at these nodes. In addition, P-RAM algorithms may require computation at nodes of the tree which are of different depth. Thus for greatest utility, the embedding should map at most a constant number of tree nodes to any node of the host graph.*

2. *Tree edges at the same depth should correspond to edge-disjoint paths in the hypercube if the commonest types of P-RAM algorithm employing this technique are to be simulated. For greatest flexibility, all tree paths should be mapped to disjoint paths in the host graph.*

3. *The maximum distance from the root to a leaf of the tree (in terms of edges of the host graph) in the embedding should be minimised in order that the routing time is minimised.*

4. *Consistent with satisfying the above points, the size of the host graph should be a minimum in the interests of processor economy.*

Recently [GP] described how the complete binary tree may be embedded in the two-dimensional mesh so as to optimise the above efficiency requirements. For that embedding, each mesh edge was regarded as two anti-parallel directed edges. The purpose of this paper is to show that the complete binary tree can be embedded with similar efficiency in the undirected hypercube. To this end, in the following section, we describe an edge-disjoint embedding of the complete binary tree with $n > 8$ leaves in the n node hypercube such that the maximum distance from a leaf to the root is $\log_2 n + 1$ (which is optimally short). Indeed, in every respect, the embedding fulfills the above efficiency requirements.

Other embeddings have been described which, in various ways, full short of the efficiency requirements stated above, for example:

1. The embedding of [BI] meets all the efficiency requirements except that the host graph is twice as large as it need be. In fact [BI] embeds the n leaf complete binary tree in the hypercube with $2n$ nodes.

2. In [Le] (pages 407-410), an embedding is described in which the n leaf tree is embedded in the n node hypercube. However, in this embedding, up to $\log_2 n$ tree nodes of different depths are mapped to a single node of the hypercube. Although the embedding is such as to facilitate the efficient implementation of most P-RAM algorithms, there may be difficulties in the exceptional cases when simultaneous computation is required to take place at an arbitrary number of different levels within the tree.

2. The embedding

In this section we describe our embedding of the complete binary tree in the hypercube. Recall that a hypercube is a graph with n nodes (where $n = 2^m$, for some

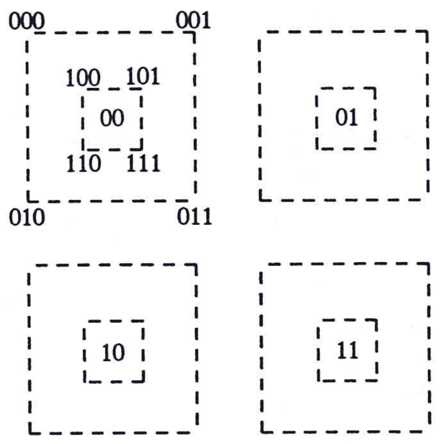


Figure 1.

positive integer m) labelled from 0 to $n-1$ in binary notation and such that there is an edge between two nodes iff their binary labels differ in exactly one bit. For the proof of theorem 1 we require to define layouts of nodes of those hypercubes with 16 and 32 nodes. For $n = 32$, we adopt the scheme of figure 1. Nodes occur at the corners of the squares defined by the dashed lines and the labels of nodes in the top left-hand quarter of the figure are shown. The first two binary bits of the labels of nodes in the other quarters are shown at the centre of their quarter of the figure. The last three binary bits of such an address will be the same as the corresponding node in the top left-hand quarter. Generally speaking, figures will only shown some edges of the hypercube, just those that are of interest. Dashed edges, such as those of figure 1, happen to correspond to certain hypercube edges but are used merely as an aid in locating nodes in the layout. For $n = 16$, the hypercube layout corresponds to the top half of figure 1. Theorem 1 is our main result.

Theorem 1 *For $n \geq 16$, there exists an embedding of the complete binary tree with n leaves in a hypercube with n nodes having the following properties:*

1. *Each hypercube node is assigned exactly one leaf of the tree.*
2. *Each hypercube node, except one, is also assigned exactly one internal node of the tree.*
3. *Distinct tree edges are mapped onto edge-disjoint (possibly null) paths in the hypercube.*
4. *The length of the image in the hypercube of a tree path from a leaf to the root is at most $\log_2 n + 1$ hypercube edges.*

Proof: We first prove the theorem for $n > 16$. For these values of n, we embed the *double-rooted complete binary tree* (denoted by DRCB tree) in the hypercube. The DRCB tree is a complete binary tree in which the path (of length 2) connecting the two sons of the root is replaced by a path of length 3. Either of the two internal

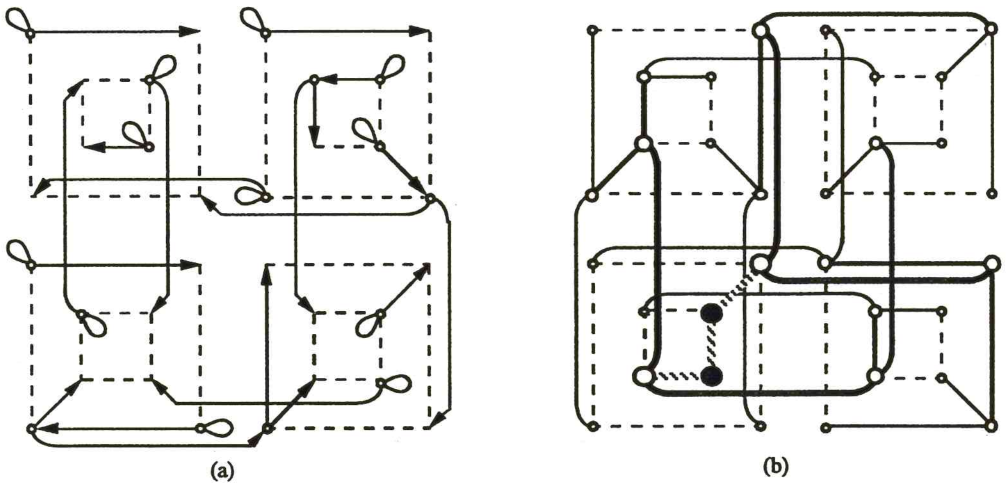
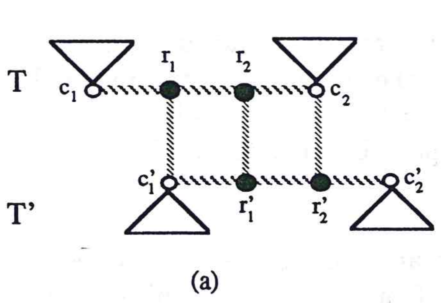


Figure 2.

nodes of this path may be regarded as the root of an embedded complete binary tree.

We inductively construct the embedding starting with the base case of $n = 32$. For clarity, we employ two figures (2(a) and (b)) to describe this case. Figure 2(a) shows the embedding of those tree edges which have leaves as end-points. For clarity, some embedded tree edges point towards that endpoint which is a leaf of the tree. Some tree edges are mapped to null paths which are indicated by loops. Figure 2(b) shows the embedding of all other tree edges. Notice that hashed edges are used for the path of length 3 on which full circles denote possible roots of the embedded complete binary tree. Also, notice that the three edges on this path belong to three different dimensions of the hypercube. In figure 2(b), the internal nodes are drawn with increasing size and the tree edges are drawn with increasing boldness the nearer they are to the root. It is easy to see that this base case satisfies the theorem in all respects, in particular the maximum root to leaf distance is 6 hypercube edges.

Figure 3 illustrates the inductive step in the construction of the embedding of the DRCB tree with n leaves in the hypercube with n nodes from two embeddings of $n/2$ leaf DRCB trees in hypercubes with $n/2$ nodes. These two embeddings are denoted by T and T' in figure 3(a). The hashed vertical edges in that figure are edges of the new dimension of the constructed hypercube. The hashed horizontal paths ((c_1,r_1,r_2,c_2) and (c'_1, r'_1, r'_2, c'_2)) are the paths of length 3 which have as internal nodes the possible roots of the embedded complete binary trees with $n/2$ leaves. The triangular shapes attached to children of these possible roots represent the embedded subtrees rooted at these children. The two smaller hypercubes are orientated so that r_1 and c'_1 are made to correspond, then the dimension corresponding with the edge (r_1,r_2) is made to correspond with the dimension of the edge (r'_1,r'_2). In this way, the nodes r_2 and r'_1 are made to correspond. Similarly, the dimension of (r_2,c_2) is made to correspond with the dimension of (r'_2,c'_2) and so node c_2 is brought into correspondence with r'_2. This is always possible given



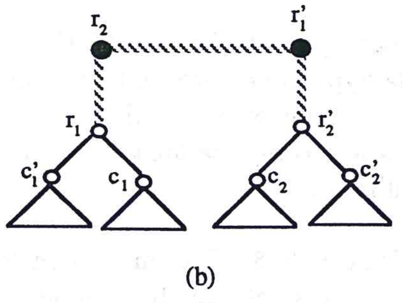
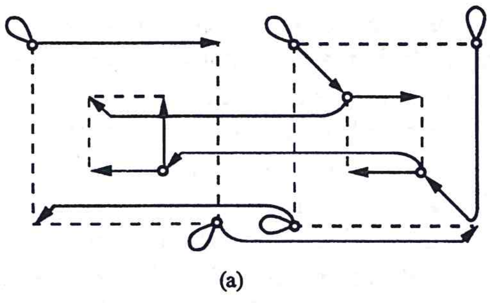


Figure 3.



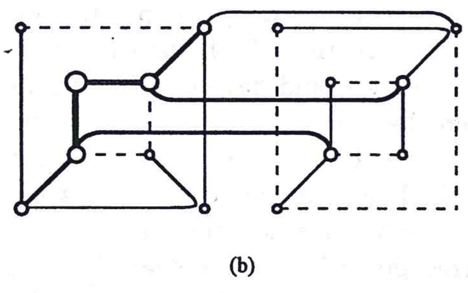


Figure 4.

the edge transitivity of the hypercube and given that each of the horizontal hashed paths of length 3 has three edges of different dimension. Figure 3(b) shows the embedding of the DRCB tree with n leaves and unit dilation in the constructed hypercube with n nodes. The labelling of nodes in this figure makes clear its derivation from figure 3(a). It is straightforward to see that the properties of the theorem statement are satisfied. The theorem is thus proved for $n > 16$.

For $n = 16$, an embedding satisfying the theorem is shown in figure 4. Again, we have used the convenience of illustrating the embedding of tree edges attached to leaves in one diagram (figure 4(a)) and in another diagram we show the embeddings of all other edges (figure 4(b)). Note that for $n = 16$ we do not show an embedding of the DRCB tree with unit dilation but an embedding of the complete binary tree for which three edges have dilation 2. $\square$

Remark 1: If we disregard figure 2(a) and take figure 2(b) as a vertex disjoint embedding of the DRCB tree with 16 leaves in the hypercube with 32 nodes then the inductive construction of figure 3 provides vertex disjoint embeddings of the n leaf DRCB tree with unit dilation in the $2n$ node hypercube. Note that [BI] also provides vertex disjoint embeddings by a similar inductive construction but the embedding of figure 2(b) cannot be obtained from [BI]. This is essentially because [BI] starts the induction at a smaller value of n. In fact, the embeddings of the n leaf tree in a $2n$ node hypercube obtainable by [BI] do not admit the addition of another level of the tree (such as figure 2(a) provides).

Remark 2: Using the inductive construction of theorem 1, it is not possible to take the base of the induction for n smaller than 32. This is because (as is trivially proved) for $n = 16$ the n leaf DRCB tree cannot be embedded in the n node hypercube with unit dilation so as to satisfy the conditions of theorem 1. In addition and by a trivial proof, for $n = 4$ and $n = 8$ it is not possible to embed the n leaf complete binary tree in the n node hypercube so as to satisfy the same conditions.

Remark 3: The maximum leaf to root distance of $\log_2 n + 1$ provided by theorem 1 is optimal. If an embedding existed satisfying the conditions of theorem 1 except that this distance be $\log_2 n$, then this would imply that a unit dilation embedding of the complete binary tree (perhaps with some leaf to father edges mapped to null paths) was possible in the hypercube. Consider the mapping of subtree consisting of all edges other than leaf to father edges. An embedding of this subtree would have to be vertex disjoint with every edge being mapped to a hypercube edge, this is not possible because this graph is not a subgraph of the hypercube. It is easy to see that it is not such a subgraph because both this subtree and the hypercube are bipartite graphs. In the case of the hypercube both halves of the bipartition contain the same number of nodes, this is not the case for the subtree and (with each subtree edge mapped precisely to an edge of the hypercube) this would force more than one node of the subtree to be embedded in a single node of the host.

References

[BI] S. N. Bhatt and I. C. F. Ipsen, *How to Embed Trees in Hypercubes*, Report Yale/DCS/RR-443, Dept. of Computer Science, Yale University (1985).

[GP] A. M. Gibbons and M. S. Paterson, *Dense Edge-Disjoint Embedding of Binary Trees in the Mesh*, Proceedings of the 4th Annual ACM Symposium on Parallel Algorithms and Architectures, San Diego, California (1992), 257-263.

[GR] A. M. Gibbons and W. Rytter, *Efficient Parallel Algorithms*, Cambridge University Press (1988).

[Le] F. T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays·Trees·Hypercubes*, Morgan Kaufmann Publishers, San Mateo, California (1992).

[Va] L. G. Valiant, *A Bridging Model for Parallel Computation*, Communications of the ACM (August, 1990) Vol.33, No.8, 103-111.