Mesh-Connected Trees: A Bridge Between Grids and Meshes of Trees

Kemal Efe, *Member, IEEE*, and Antonio Fernández

Abstract—The grid and the mesh of trees (or MOT) are among the best-known parallel architectures in the literature. Both of them enjoy efficient VLSI layouts, simplicity of topology, and a large number of parallel algorithms that can efficiently execute on them. One drawback of these architectures is that algorithms that perform best on one of them do not perform very well on the other. Thus there is a gap between the algorithmic capabilities of these two architectures.

We propose a new class of parallel architectures, called the *mesh-connected trees* (or MCT) that can execute grid algorithms as efficiently as the grid, and MOT algorithms as efficiently as the MOT, up to a constant amount of slowdown. In particular, the MCT topology contains the MOT as a subgraph and emulates the grid via embedding with dilation 3 and congestion two. This significant amount of computational versatility offered by the MCT comes at no additional VLSI area cost over these earlier networks. Many topological, routing, and embedding properties analyzed here suggests that the MCT architecture is also a serious competitor for the hypercube. In fact, while the MCT is much simpler and cheaper than the hypercube, for all the algorithms we developed, the running time complexity on the MCT matches those of well known hypercube algorithms.

We also present an interesting variant of the MCT architecture that admits both the MOT and the torus as its subgraphs. While most of the discussion in this paper is focused on the MCT architecture itself, these analyses can be easily extended to the variant of the MCT presented here.

Index Terms—Parallel architectures, interconnection networks, parallel algorithms, product networks, graph embedding, binary tree, grids, hypercubes, mesh of trees.

1 Introduction

GRIDS and meshes of trees (or MOT) are among the best-known interconnection networks devised. Grids are very efficient in computations that require nearest neighbor communication in a lattice space. Meshes of trees perform better with algorithms that require data broadcasting in the rows or columns. Neither one of these networks is very effective in the application domain of the other. In this paper, we propose a new network that bridges the gap that exists between these two classes of networks. We call this new network the *mesh-connected trees* (MCT). The most important feature of the MCT network is that it can perform the grid algorithms as efficiently as the grid, and the MOT algorithms as efficiently as the MOT, up to constant (and small) factors of slowdown, without increasing the asymptotic VLSI area complexity of these networks. In particular, it contains the MOT as a subgraph, and emulates the grid via embedding with dilation three and congestion two. These capabilities make the proposed network extremely useful for large classes of computations, because there is a vast number of algorithms in the literature that have been developed for grids and meshes of trees. Readers not familiar with these algorithms may refer to [14] for a rich collection of such algorithms.

- *K. Efe is with the Center for Advanced Computer Studies, University of Southwestern Louisiana, Lafayette, LA 70504. E-mail: efe@cacs.usl.edu.*
- *A. Fernández is with the Departamento de Arquitectura y Tecnología de Computadores, Universidad Politécnica de Madrid, Spain. E-mail: anto@eui.upm.es.*

Manuscript received Dec. 3, 1993; revised Feb. 20, 1995.
For information on obtaining reprints of this article, please send e-mail to: transpds@computer.org, and reference IEEECS Log Number D95187.

The MCT network is the multidimensional cross product of complete binary trees. Informally, the N^r-node r-dimensional MCT, denoted as $MCT_r(N)$, is obtained from the N^r-node r-dimensional grid by replacing the linear connections along each grid dimension by the connections of an N-node complete binary tree. Fig. 1 shows the 49-node two dimensional mesh-connected trees, $MCT_2(7)$. We use the notation MCT to refer generically to the class of networks that we call mesh-connected trees, while we use $MCT_r(N)$ when we refer specifically to the N^r-node r-dimensional mesh-connected trees.

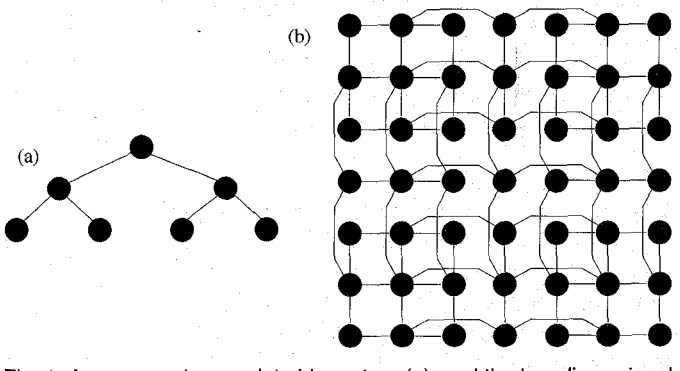


Fig. 1. A seven-node complete binary tree (a), and the two-dimensional mesh-connected trees obtained from it (b), denoted $MCT_2(7)$.

While the MCT network is not more powerful than the hypercube, it can be made as powerful as the hypercube provided that the MCT network is built with a large enough number of dimensions. This fact aside, we do not propose the MCT network to be built with a large number

of dimensions, even though the bandwidth will be higher with the higher number of dimensions. The MCT network, even with a few dimensions, has large enough bandwidth to be considered as a low cost alternative for general-purpose parallel computers. For many algorithms that we considered (see Section 8), the MCT architecture with a few dimensions have the same running time complexity as the hypercube implementations of these algorithms.

We start our study of the MCT by showing that it has logarithmic diameter and a large bisection width. Subsequently, we examine the embedding capabilities of the MCT and show optimal embedding for various networks. We then develop a shortest-path routing algorithm from an arbitrary source to an arbitrary destination, as well as a broadcasting algorithm with optimal running time. We also analyze the number of vertex-disjoint paths between arbitrary pairs of vertices. We then turn our attention to the layout area required by the MCT network. We show that the layout area required by the MCT is the same as the area required by the MOT (within a constant factor), and for more than two dimensions this area is the same as the area required by the grid. Therefore, the significant computational power of the MCT comes at no additional cost over those of the grid or the MOT.

Among the significant contributions of this paper is the introduction of a simple variant of the basic MCT architecture. If we connect the leaves of the complete binary tree by a straight line (see Fig. 7), and then construct the product graph of this extended tree, the resulting graph has all the capabilities of the regular MCT, and it also contains the corresponding-size torus as a subgraph. This is a significant improvement over that of the plain MCT which can only embed the grid with dilation cost three and congestion cost two. Note that the cost of this extension will be negligible, since all the nodes of the MCT architecture can be build identically, and the unused I/O ports at the leaves can be used to realize these connections. All the theoretical analyses given here for the basic MCT architecture can be extended to this variant easily. We therefore focus much of our discussion on the basic MCT graph rather than this extension.

Finally, we briefly address the complexity of several representative algorithms running on the MCT. We find that the asymptotic running time complexities of these algorithms on the MCT architecture are same as those of well known algorithms developed for the hypercube.

1.1 Related Work

The cross product operation has been used as a common framework to study interconnection networks and as a way for defining new interconnection networks with interesting properties [1], [9], [17], [20]. Baumslag and Annexstein [1] obtained some general off-line permutation routing algorithms for product networks. El-Ghazawi and Youssef [9] studied the connectivity of product networks with respect to the connectivity of their factor networks. They obtained lower bounds in the connectivity of product networks and presented an adaptive fault-tolerant routing algorithm for them.

Rosenberg [17] introduced the product of de Bruijn graphs as a potential parallel architecture and analyzed several of its computational properties. He showed that the product of de Bruijn networks contains rings, grids, complete binary trees, and meshes of trees as its subgraph. It can also emulate butterflies, shuffle-exchange and de Bruijn graphs with low dilation and congestion. Youssef [20] defined new product networks by combining the hypercube with various other networks.

Efe and Fernández [5] explored multidimensional "homogeneous" product networks, i.e., product networks whose factor graphs are all isomorphic. General results were derived regarding structural and embedding properties of homogeneous product networks, and these results were applied to three specific cases. In our knowledge the MCT network first appeared there as a potential candidate for a parallel architecture.

2 Definitions

In this paper, a network is seen as an undirected graph, whose vertices represent processors and whose edges represent bidirectional links between the processors. We use the terms graph and network interchangeably. Several properties of the MCT are derived from those of the complete binary tree, we therefore list relevant properties of the latter graph.

Definition 1. *The h-level complete binary tree, $T(h)$, is the graph whose vertices comprise the set $\{1, \ldots, 2^h - 1\}$ and whose edges connect each vertex $u < 2^{h-1}$ with the vertices $2u$ and $2u + 1$.*

The number of vertices of $T(h)$ is $2^h - 1$. We will often denote this value as N (i.e., h is $\log(N + 1)$). The vertex labeled one, at level one, is the root of the tree. Vertices of the tree at some level j are numbered from 2^{j-1} to $2^j - 1$, inclusive. The vertices at level h are the leaves of the tree. The parent vertex of a node u can be obtained as $\lfloor \frac{u}{2} \rfloor$.

$T(h)$ has $N - 1$ edges. The diameter of $T(h)$ is $2(h - 1)$. The bisection width is one as the edges incident to the root divide $T(h)$ into an $\lfloor \frac{N}{2} \rfloor$-node and an $\lceil \frac{N}{2} \rceil$-node graph. The minimum vertex degree of $T(h)$ is 1 (for the leaves) and the maximum vertex degree is 3 (for internal nodes).

Finally, $T(h)$ is k-partitionable for $k = 2^i$, $0 \le i \le h - 1$. A graph is said to be k-partitionable if it contains k disjoint isomorphic subgraphs which are consistent with its class definition. Partitionability of a graph is an useful attribute, as it implies a recursive partitioning for subproblems of computations on the architecture.

We denote a vertex of the $MCT_r(N)$ by an r-tuple $x = x_{r-1} \ldots x_1 x_0$, where the index position i refers to dimension i, and x_i indicates a vertex in $T(h)$, the factor graph. A formal definition of the MCT is presented below.

Definition 2. *The N^r-node r-dimensional mesh-connected trees, $MCT_r(N)$, is the graph whose vertices comprise all the r-tuples $x = x_{r-1} \ldots x_1 x_0$, such that, for $0 \le i \le r - 1$, every x_i is a vertex of $T(h)$, and the pair (x, y) defines an edge in $MCT_r(N)$ if and only if x and y differ in exactly one index position i and (x_i, y_i) is an edge in $T(h)$.*

An embedding of a "guest" graph G into a "host" graph

H is a mapping of the vertices of G into the vertices of H, and a mapping of the edges of G into paths in H connecting the images of the vertices incident to the edge. The main cost measures used in embedding efficiency are the following:

- *Load* is the maximum number of vertices of G mapped to any vertex of H.
- *Dilation* is the maximum path length in H representing an edge of G.
- *Congestion* is the maximum number of paths (that correspond to the edges of G) that share any edge of H.

In this paper, we focus on unit-load embeddings and use the dilation and congestion costs to evaluate the embedding efficiency.

3 Structural Properties

It can be easily seen that the $MCT_r(N)$ has N^r vertices. Its number of edges can be computed as follows: $MCT_r(N)$ contains N^{r-1} copies of the complete binary tree $T(h)$ along each dimension (remembering that $h = \log(N + 1)$). There are r dimensions, and $T(h)$ has $N - 1$ edges. Therefore $MCT_r(N)$ has $rN^{r-1}(N - 1)$ edges.

A highly desirable characteristic of a network is a small and fixed vertex degree. A fixed vertex degree allows easy network expansion, while a small vertex degree implies small cost for communication channels built within each node. The $MCT_r(N)$ has a minimum vertex degree of r and a maximum vertex degree of $3r$. In any implementation, r can be kept small if desired without limiting the growth of the network, because arbitrarily large MCT networks can be built by using correspondingly large trees at each dimension. Hence, the number of dimensions of the MCT may be kept small and fixed without sacrificing the expandability.

We begin with discussing the diameter and the bisection width of the MCT.

THEOREM 1. *$MCT_r(N)$ has diameter $2r(h - 1)$, where $h = \log(N + 1)$.*

PROOF. We start by showing that $2r(h - 1)$ is an upper bound on the distance between any two nodes in $MCT_r(N)$. Let x and y be two nodes of $MCT_r(N)$. To reach y from x it is necessary to traverse at most r different $T(h)$ trees, one in each dimension. Since the diameter of $T(h)$ is $2(h - 1)$, the diameter of $MCT_r(N)$ cannot be larger than $2r(h - 1)$.

We now show that this upper bound is tight by presenting two nodes in $MCT_r(N)$ whose distance is exactly $2r(h - 1)$. As the diameter of $T(h)$ is $2(h - 1)$, there are at least two nodes, say u, v, in $T(h)$ that are at distance $2(h - 1)$. Consider a shortest path from the node $x = x_{r-1} \ldots x_1x_0$ to the node $y = y_{r-1} \ldots y_1y_0$, where $x_i = u$, and $y_i = v$ for all $0 \le i \le r - 1$. The problem of finding a path between x and y can be seen as the problem of transforming the label of x into the label of y by only traversing edges of $MCT_r(N)$. From Definition 2 we know that the traversing of an edge of $MCT_r(N)$ affects only one index position of the label. As label changes must be consistent with the adjacencies of $T(h)$, the shortest way to transform the ith index position from x_i to y_i is to follow the shortest path from x_i to y_i along the dimension i tree. Since the shortest path from x to y must traverse r trees with a path of length $2(h - 1)$ on each, the distance from x to y is exactly $2r(h - 1)$.

From this, it follows that the diameter of the MCT is logarithmic in the number nodes. □

THEOREM 2. *The bisection width of $MCT_r(N)$ is at least $\frac{N^{2r}-1}{N^{r-1}(N^2-1)}$.*

PROOF. To prove the theorem, we construct an embedding of the N^r-node directed complete graph into $MCT_r(N)$, such that each edge of the graph is mapped to a path in $MCT_r(N)$. We show that, in that embedding, any edge of $MCT_r(N)$ is contained in at most $N^{r-1}(N^2 - 1)/2$ of these paths. It is known that the bisection width of the N^r-node directed complete graph is $(N^{2r} - 1)/2$, when N is odd. Therefore, the bisection width of $MCT_r(N)$ has to be at least $\frac{N^{2r}-1}{N^{r-1}(N^2-1)}$, because otherwise we could get a bisection of the complete graph by removing less edges than the value of its bisection width.

We construct the above mentioned embedding as follows. We initially map the nodes of the directed complete graph to the nodes of $MCT_r(N)$ one to one. Then, we map the edges of the complete graph to paths in $MCT_r(N)$ as follows: we map the edge from the node mapped to $x = x_{r-1}x_{r-2} \ldots x_0$ to the node mapped to $y = y_{r-1}y_{r-2} \ldots y_0$ through the path $x_{r-1}x_{r-2} \ldots x_0 \to y_{r-1}x_{r-2} \ldots x_0 \to y_{r-1}y_{r-2} \ldots x_0 \to \ldots \to y_{r-1}y_{r-2} \ldots y_0$, such that the shortest (actually the unique) path is used within each tree. We have to show now that, in this embedding, any edge of $MCT_r(N)$ is contained in at most $N^{r-1}(N^2 - 1)/2$ of these paths.

Consider any edge of $MCT_r(N)$ connecting the node $z_{r-1} \ldots z_k \ldots z_0$ to the node $z_{r-1} \ldots \left\lfloor \frac{z_k}{2} \right\rfloor \ldots z_0$, i.e., the connection is from vertex z_k to its parent in a dimension-k tree which is uniquely determined by the z_i values for $0 \le i \le r - 1$, $i \ne k$. Since that edge is the only connection between the subtree rooted at z_k and the rest of that tree, it must be traversed for every path connecting these two parts of the tree. Let s be the number of nodes in the subtree with root z_k, then the number of possible choices for x_k and y_k is $2s(N - s)$.

If a path from a node $x = x_{r-1}x_{r-2} \ldots x_0$ to a node $y = y_{r-1}y_{r-2} \ldots y_0$ contains that edge, then $y_i = z_i$, for $i > k$, and $x_i = z_i$, for $i < k$, and the edge $\left(z_k, \left\lfloor \frac{z_k}{2} \right\rfloor\right)$ must be contained in the shortest path from x_k to y_k in the dimension-k tree. As $x_{r-1}, \ldots, x_{k+1}, y_{k-1}, \ldots, y_0$ can take N possible values, the number of paths that contain the edge is $2N^{r-1}s(N - s) \le N^{r-1}(N^2 - 1)/2$ (note that N is odd). Hence, at most $N^{r-1}(N^2 - 1)/2$ paths contain any edge of $MCT_r(N)$.

Finally, for the purpose of eventual contradiction, we assume that the bisection width of $MCT_r(N)$, B, is less than $\frac{N^{2r}-1}{N^{r-1}(N^2-1)}$. If so, $MCT_r(N)$ can be divided into two sets of nodes of equal size (within one) connected by B edges. This partition also induces a partition of the nodes of the N^r-node directed complete graph, where every one of the B edges embeds the paths that contain it. Thus, the complete graph has a bisection width of at most $BN^{r-1}(N^2 - 1)/2 < (N^{2r} - 1)/2$. Since the bisection width of the complete graph is $(N^{2r} - 1)/2$, a contradiction is reached and B must be at least $\frac{N^{2r}-1}{N^{r-1}(N^2-1)}$. □

This large bisection width of the MCT, coupled with the logarithmic diameter, make it an attractive architecture.

Another important structural property for any network is the partitionability. A partitionable network adapts better to different problem sizes, or it may be used to simultaneously solve several problems. $MCT_r(N)$ contains N vertex-disjoint copies of $MCT_{r-1}(N)$ as subgraphs. These are obtained by removing all the edges of the trees in any dimension i in $MCT_r(N)$. If this fact and partitionability of $T(h)$ are recursively applied, it is easily observed that $MCT_r(N)$ is k^r-partitionable, for $k = 2^i$, $0 \le i \le h - 1$. This high degree of partitionability is not surprising due to the partitionability of complete binary trees.

4 Embedding Properties

Embedding properties are among the most important properties of a network, because they transfer the computational power of a guest network to a host network.

Among the well known and important networks, the torus, and the grid can be embedded into the MCT with constant dilation and congestion. These results are interesting because several other powerful networks such as the shuffle-exchange, de Bruijn, butterfly, cube-connected cycles, or Benes networks, require logarithmic dilation to embed the grid or torus [2]. Next, we show that the MCT is more powerful than the grid by showing that the MCT cannot be embedded into the grid with constant dilation. We recall from [5], additionally, that the MOT network is a subgraph of the MCT. Thus it can perform every computation that the MOT can without any slowdown. We also improve the embedding of complete binary trees into the MCT originally presented in [5]. We conclude by demonstrating a dilation two congestion one embedding of the MCT into the hypercube.

4.1 Embedding the Grid and the Torus

It was shown in [14] that the *N-node* cycle can be embedded into any N-node tree with dilation cost three and congestion cost two. Here, we first strengthen this result by showing that dilation three and congestion two is the best that can be achieved when embedding the N-node cycle into the *N–node* complete binary tree.

LEMMA 1. *Any embedding of the N–node linear array into the N-node complete binary tree T(h), where $h \ge 5$, requires at least dilation cost three and congestion cost two.*

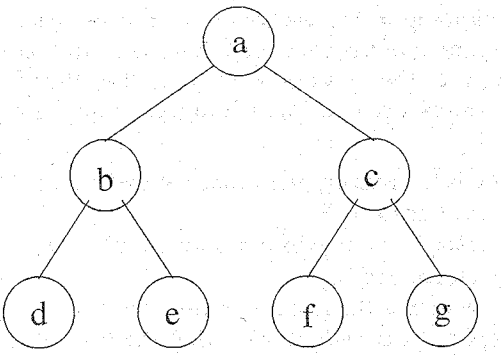


Fig. 2. A subtree of $T(h)$, where $h \ge 5$.

PROOF. First, consider the dilation cost. For the sake of eventual contradiction, suppose that there exists a dilation two embedding of the *N-node* linear array into $T(h)$, where $h \ge 5$. Such an embedding implies a traversal of all the nodes of $T(h)$ with movements of at most distance two from one node to another so that each node of the tree is visited only once. We say that a node is "reachable" from another if the distance between them is at most two.

If we remove the $h - 3$ top levels of a tree $T(h)$, where $h \ge 5$, we obtain 2^{h-3} disjoint subtrees isomorphic to the one shown in Fig. 2. For instance, when $h = 5$, four of these subtrees are obtained by removing the two top levels. Since any traversal of $T(h)$ has only one starting node and only one ending node, at least two of these subtrees are traversed by entering from the top and exiting from the top. Now assuming that the tree of Fig. 2 is such a subtree, we focus on the behavior of the traversal.

It is not possible for the traversal to enter and exit the subtree of Fig. 3 more than once. This is because there are only three external nodes "reachable" from the nodes of this subtree: the parent of a, the sibling of a, and the grandparent of a. If, for example, it were possible to traverse the subtree in Fig. 3 by entering and exiting twice, this would imply that four external nodes are reachable from the nodes of this tree, which is not the case. Therefore every node in the subtree has to be visited the first time when the subtree is being traversed.

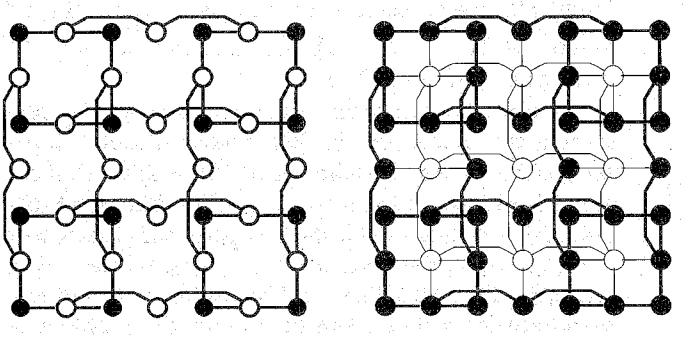


Fig. 3. Embedding the 4×4 MOT into $MCT_2(7)$. The 4×4 MOT graph is shown on the left where the leaves are highlighted as dark nodes. In the MCT graph on the right, the 4×4 MOT subgraph is highlighted as dark nodes and edges.

Since the traversal enters from the top, the first visited node has to be one of a, b, or c. Also, the last visited

node has to be one of a, b, or c. It is easily seen that we cannot have one of b or c as the first visited node and the other as the last. This is because there is only one external node reachable from b or c (it is the parent of a). So the only case that needs to be examined is entering the subtree at a and exiting at one of b or c.

If a is the first visited node, the next node is either in the subtree rooted at b or in the subtree rooted at c. In any case, to go from one subtree to the other the traversal needs to visit both of b and c in order to limit the dilation at two. Since at least one of these must be saved as the exit node, a contradiction has been reached and there is no way to traverse $T(h)$ with dilation cost two, and the dilation of any embedding must be at least three.

To prove that the congestion cost must be at least two, we observe that the N-node binary tree has $\frac{N+1}{2}$ nodes (leaves) with degree one, while the linear array has only two nodes with degree one. Then, several nodes of the linear array must be mapped to the leaves of the tree. When such a node of the linear array is assigned to a leaf, its two neighbors must be mapped in a way that shares the edge connecting the leaf to its parent. Therefore the congestion must be at least two. □

This result shows that the method of embedding the linear array into the complete binary tree in [14] is optimal. We can use this result to construct an embedding for the N^r-node r-dimensional torus into $MCT_r(N)$ as follows: We first embed the N-node cycle into the N-node complete binary tree with dilation cost three and congestion cost two. Then, by using Definition 2, we construct the $MCT_r(N)$. This construction automatically induces an embedding for the N^r-node r-dimensional torus into $MCT_r(N)$ with dilation cost three and congestion cost two. The next result addresses the optimality of these dilation and congestion costs for the embedding of the grid into the MCT graph. The same result also applies for torus since the grid is a subgraph of the torus.

THEOREM 3. *Any embedding of the N^r-node r-dimensional grid into $MCT_r(N)$ requires at least dilation cost two and congestion cost two.*

PROOF. Consider the number of nodes with vertex degree exactly r. In $MCT_r(N)$ there are $\left(\frac{N+1}{2}\right)^r$ nodes with degree exactly r (these correspond to the leaves of the factor binary tree). In the grid there are only 2^r nodes with degree exactly r (these correspond to the end-points of the factor linear array). All the remaining $N^r - 2^r$ nodes of the grid have larger vertex degrees. Therefore, for $N > 3$, several grid nodes must be mapped to MCT nodes with smaller degree. Once such a grid node is mapped to an MCT node with lesser degree, at least one neighbor of the grid node must be mapped with dilation two and congestion two. □

This theorem shows that the congestion of the simple embedding method described above is optimal. Dilation of this embedding differs from the proven lower bound by one unit. After considering the problem in some detail, we conjecture that the dilation of three may also be optimal, although the theorem above only bounds the dilation as being at least two.

Later, we will show in Section 7 that simple extensions made in the basic topology of the MCT allow reducing these constants to unity. Even without such extensions, the above embedding is much better than any embedding of the n-node grid into the de Bruijn or the shuffle-exchange graphs that require $\Omega(\log \log n)$ dilation [2], or into the butterfly, cube-connected cycles, or Beneš networks that require $\Omega(\log n)$ dilation [2], [14]. This high cost in embedding a graph as important as the grid reduces the practical value of these networks, and favors the MCT network.

The question of whether the torus or the grid can be embedded into the MOT with constant dilation and constant congestion is currently open. It seems plausible, therefore, that such an embedding will probably be complex, should it exist. The simple embeddings that we have obtained for the MCT network also favors it over the MOT.

The next theorem establishes that the grid is not very good at emulating the MCT. Since the MCT graph can emulate the grid with constant slowdown, it follows that the MCT is computationally more powerful than the grid.

THEOREM 4. *Any embedding of $MCT_r(N)$ into the N^r-node r-dimensional grid requires at least dilation cost $\left\lceil \frac{N-1}{2h-2} \right\rceil$.*

PROOF. The claim follows from a comparison of the diameters of the two graphs. Diameter of the N^r-r-dimensional grid is $r(N-1)$. Diameter of the $MCT_r(N)$ is $r(2h-2)$. Therefore, there is at least one edge of $MCT_r(N)$ that must be mapped to a path of length $\left\lceil \frac{r(N-1)}{r(2h-2)} \right\rceil = \left\lceil \frac{N-1}{2h-2} \right\rceil$ in the grid. □

4.2 Embedding the MOT

The following theorem (Theorem 6 in [5]) shows that the MOT is a subgraph of the MCT.

THEOREM 5. *$MCT_r(N)$ has the r-dimensional mesh of $\frac{N+1}{2^i}$-leaf trees as subgraph, for $1 \le i \le h-1$.*

Theorem 5 allows us to view the MCT as a hierarchy of nested interconnected meshes of trees of different sizes, so that it may be adapted to solve multiple problems, or tailored to a particular size. Fig. 3 shows the two-dimensional MOT networks contained in $MCT_2(7)$. In this figure, there are two MOT networks contained, one with four leaves for each tree (shown in dark nodes) and one with two leaves for each tree (shown in empty nodes, ignoring the center node). The largest MOT that can be embedded into $MCT_r(N)$ has $\Theta(N^r)$ nodes. This fact will be used in Theorem 10 to obtain a lower bound on the layout area required by the MCT.

4.3 Embedding the complete binary tree

In [5], it is shown that the $(r(h-1)+1)$-level complete binary tree is a subgraph of $MCT_r(N)$, where $h = \log(N+1)$. For easy reference, this result is included here.

THEOREM 6. *$T(r(h-1)+1)$ is a subgraph of $MCT_r(N)$, where $h = \log(N+1)$.*

As an example, Fig. 4 shows the embedding method for $r = 2$. The complete binary tree subgraph of the MCT obtained in this theorem is the largest possible when $r = 2$, but for $r > 2$, larger trees can be embedded with constant dilation and constant congestion. We use Theorem 6 here to obtain an embedding of a larger complete binary tree than that embedded by Theorem 6 itself when $r > 2$. In particular, we show a method that embeds the largest possible tree when $r \leq 3$ and very close to the largest for small values of r. For instance, $MCT_9(7)$ has enough nodes to contain a 25-level complete binary tree and our method embeds a 23-level tree.

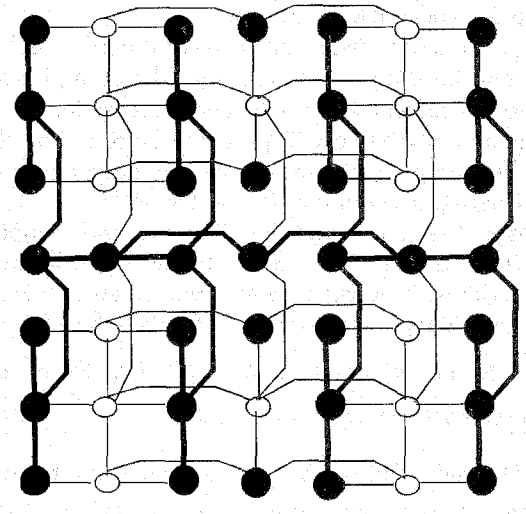


Fig. 4. Embedding the complete binary tree into the MCT by Theorem 6. The complete binary tree subgraph is highlighted by heavy dark lines.

In order to simplify the proofs we will first distinguish special sets of nodes in any MCT network as follows.

DEFINITION 3. *A node $x = x_{r-1} \ldots x_1 x_0$ is a leaf of $MCT_r(N)$ if and only if x_i is a leaf of $T(h)$, for all $0 \leq i \leq r-1$.*

DEFINITION 4. *The node $x = x_{r-1} \ldots x_1 x_0$ is the root of $MCT_r(N)$ if and only if $x_i = 1$ (i.e., x_i is the root of $T(h)$), for all $0 \leq i \leq r-1$.*

We now define a new class of graphs that is going to be useful in this section. We do not give a special name to these graphs, we instead use a short notation to identify them.

DEFINITION 5. *$TT(l, r, N)$ is the graph obtained by connecting the roots of N^r copies of $T(l)$ in the $MCT_r(N)$ pattern, i.e., $TT(l, r, N)$ is obtained by "hanging" a complete binary tree, $T(l)$, at each node of $MCT_r(N)$.*

First we show that $T(l + 5)$ can be embedded into $TT(l, 2, 7)$ with constant dilation and congestion and that this embedding has particular properties. These properties are used to obtain the subsequent results which show how to embed $T\left(3r - \left\lfloor \frac{r}{2} \right\rfloor\right)$ in $MCT_r(7)$ by iteratively using this first embedding. Finally, by combining this result and Theorem 6 the general result is obtained in Theorem 7.

LEMMA 2. *$T(l + 5)$ can be embedded into $TT(l, 2, 7)$, where $l \geq 2$, with dilation three and congestion three. In this embedding the root of the embedded tree coincides with the root of the $MCT_2(7)$ subgraph of $TT(l, 2, 7)$ and the edges incident to the root are embedded with dilation one and congestion one.*

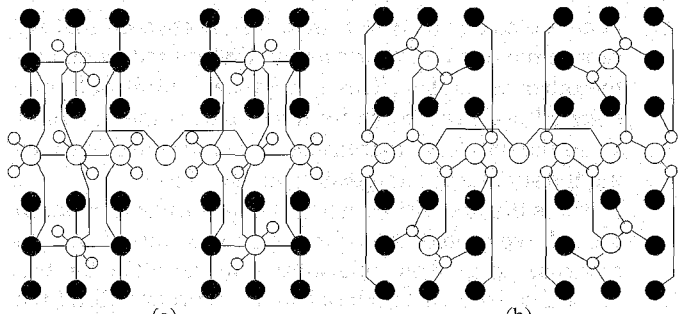


Fig. 5. Embedding the $(l + 5)$-level complete binary tree into a subgraph of $TT(l, 2, 7)$.

PROOF. Fig. 5a shows a subgraph of $TT(l, 2, 7)$. In this figure, dark nodes represent $T(l)$ trees collapsed into supernodes for the purpose of a suitable abstraction for the discussion below. Large empty nodes represent roots of other $T(l)$ trees, and small empty nodes represent their immediate children in their $T(l)$ trees. The subtrees rooted at small empty nodes are ignored. Fig. 5b shows the tree that can be embedded into this subgraph. The edges shown correspond to the edges of the complete binary tree embedded. Only dark nodes in Fig. 5b have $T(l)$ subtrees.

It can be easily checked that any edge in Fig. 5b corresponds to a path of length not more than three in Fig. 5a. Dilation three edges are those that connect the large dark nodes to small empty nodes in Fig. 5b. It can be also easily seen that the maximum congestion of three is found in some of the edges connecting large empty nodes with small empty nodes in Fig. 5a (the reader can trace the connections sharing the edge from the large empty node to the small empty node at center-right of Fig. 5a).

Since the tree of Fig. 5.(b) has six levels and each dark node represents a collapsed l-level tree, we have obtained an embedding of $T(l + 5)$ into $TT(l, 2, 7)$, where the dilation and the congestion costs are three. From the figure it is easily verified that the root of the embedded tree coincides with the root of the $MCT_2(7)$ subgraph and that the edges incident to the root of the tree are mapped to the edges of $TT(l, 2, 7)$ with unit dilation and congestion. □

The properties of the embedding highlighted in the statement of the lemma are needed in order to iteratively apply the embedding (in the next lemma) without increasing the congestion of the global embedding.

LEMMA 3. *$T\left(3r - \frac{r-1}{2}\right)$ can be embedded into $MCT_r(7)$, where r is odd, with dilation three and congestion three. In this embedding the root of the embedded tree is the root of $MCT_r(7)$ and the edges incident to the root of the embedded tree have dilation one and congestion one.*

PROOF. We prove the claim by induction in the number of dimensions, r. The initial condition, $r = 1$, is trivially verified, since $MCT_1(7)$ is isomorphic to $T(3)$. In the induction step we have to show that, given an embedding of $T\left(3k - \frac{k-1}{2}\right)$ into $MCT_k(7)$ as specified, it is possible to embed $T\left(3(k+2) - \frac{(k+2)-1}{2}\right)$ into $MCT_{k+2}(7)$.

By removing all the edges along dimensions k and $k+1$ from $MCT_{k+2}(7)$ we obtain 49 disjoint copies of $MCT_k(7)$. From the induction hypothesis, we can embed a disjoint copy of $T\left(3k-\frac{k-1}{2}\right)$ into each of these copies. The root of the embedded tree has label $x = x_{k-1} \ldots x_1x_0$, where $x_i = 1$ for $0 \le i \le k-1$, and the edges incident to the root are edges of $MCT_k(7)$.

Now consider only the roots of the embedded trees and reconnect them along dimensions k and $k+1$. The graph so obtained contains the nodes of $MCT_{k+2}(7)$ of the form $x = x_{k+1}x_kx_{k-1} \ldots x_1x_0$, where $x_i = 1$ for $0 \le i \le k-1$, and is isomorphic to $MCT_2(7)$.

Each node in the above graph is the root of an embedded complete binary tree. Then, considering again the whole graph, we have obtained an embedding of $\left(3k-\frac{k-1}{2}+5\right)$ into $MCT_{k+2}(7)$, where the $MCT_2(7)$ subgraph and the first two levels of the trees are embedded with dilation one and congestion one. Since the embedding method of Lemma 2 only changes this part of the TT graph, it can be applied here to obtain an embedding of the $\left(3k-\frac{k-1}{2}+5\right)=\left(3(k+2)-\frac{(k+2)-1}{2}\right)$-level complete binary tree into $MCT_{k+2}(7)$ with dilation three and congestion three.

From Lemma 2, the root of the embedded tree is the root of the $MCT_2(7)$ subgraph, that is of the form $x = x_{k+1}x_kx_{k-1} \ldots x_1x_0$, where $x_i = 1$ for $0 \le i \le k+1$, and the edges incident to this root have dilation one and congestion one. □

LEMMA 4 *$T\left(3r-\left\lfloor\frac{r}{2}\right\rfloor\right)$ can be embedded into $MCT_r(7)$ with dilation three and congestion three. In this embedding the root of the embedded tree is the root of $MCT_r(7)$.*

PROOF. If r is odd the above lemma can be trivially applied and the claim follows. The case of even r requires a little more elaboration.

By removing all the dimension-$(r-1)$ edges we obtain 7 disjoint copies of $MCT_{r-1}(7)$. Since r is even $r-1$ is odd and the above lemma can be applied to each copy. Then $T\left(3(r-1)-\frac{r-2}{2}\right)$ can be embedded into each copy with its root in the node $x = x_{r-2} \ldots x_1x_0$, with $x_i = 1$ for $0 \le i \le r-2$.

We can now connect the roots of the embedded trees with a dimension-$(r-1)$ tree. This tree has three levels and each of its leaves is the root of a $\left(3(r-1)-\frac{r-2}{2}\right)$-level tree, and hence we have found an embedding of $T\left(3r-\frac{r}{2}\right)$ into $MCT_r(7)$. Since the dimension-$(r-1)$ tree connects the roots of the seven copies and the root of the complete binary tree embedded is the root of this tree, the root of the embedded tree is the node $x = x_{r-1} \ldots x_1x_0$, with $x_i = 1$ for $0 \le i \le r-1$, root of $MCT_r(7)$. □

THEOREM 7. *$T\left(rh-\left\lfloor\frac{r}{2}\right\rfloor\right)$ can be embedded into $MCT_r(N)$, where $N > 3$, with dilation three and congestion three.*

PROOF. If we remove the two lowest levels from every tree along each dimension in $MCT_r(N)$ we obtain a graph isomorphic to $MCT_r(2^{h-2}-1)$. Similarly, if we remove the $h-3$ top levels from every tree along each dimension we obtain a disconnected graph formed by $2^{r(h-3)}$ disjoint copies of $MCT_r(7)$. In the rest of this proof we will pay special attention to the two subgraphs of $MCT_r(N)$ so identified. In particular, we first show how to embed a $(r(h-3)+1)$-level tree into the first subgraph whose leaves are the leaves of $MCT_r(2^{h-2}-1)$. Then, we show how to embed a $\left(3r-\frac{r}{2}\right)$-level tree into each copy of $MCT_r(7)$ so that the root of the tree is the root of the copy. The combination of both embeddings into $MCT_r(N)$ yields the desired embedding.

We first recall that Theorem 6 shows that $MCT_r(N)$ has a subgraph isomorphic to $T(r(h-1)+1)$. By construction, the leaves of this tree are also leaves of $MCT_r(N)$. The direct application of Theorem 6 to $MCT_r(2^{h-2}-1)$ allows to obtain a subgraph of this graph isomorphic to $T(r(h-3)+1)$ and whose leaves are the leaves of $MCT_r(2^{h-2}-1)$.

Lemma 4 shows how to embed $T\left(3r-\left\lfloor\frac{r}{2}\right\rfloor\right)$ with congestion three and dilation three into each copy of $MCT_r(7)$ so that the root of the tree is the root of the copy. Combining this result with the previous one we have obtained an embedding of the $\left(3r-\left\lfloor\frac{r}{2}\right\rfloor+r(h-3)\right)=\left(rh-\left\lfloor\frac{r}{2}\right\rfloor\right)$-level complete binary tree into $MCT_r(N)$ with congestion three and dilation three. □

4.4 Embedding the MCT into the Hypercube

We close this section by showing how to embed the $MCT_r(N)$ graph into the (hr)-dimensional hypercube, where $h = \log(N+1)$. Since the MCT architecture is topologically much simpler than the hypercube, the embedding method presented here may be used as a suitable abstraction for developing parallel algorithms for the hypercube architecture.

THEOREM 8. *$MCT_r(N)$ can be embedded into the (hr)-dimensional hypercube with dilation cost two and congestion cost one, where $h = \log(N+1)$.*

PROOF. In [10] it is proven that the $(N+1)$-node double-rooted complete binary tree is a subgraph of the $(N+1)$-node h-dimensional hypercube. This mapping implies an embedding of $T(h)$ into the $(N+1)$-node hypercube with dilation two and congestion one.

From [5] we know that if G can be embedded into H with dilation d and congestion c, then the r-dimensional product of G can be embedded into the r-dimensional product of H with dilation d and congestion c. Since the rh-dimensional hypercube can be considered as the r-dimensional product of 2^h-node hypercubes, the claim follows. □

5 Routing Algorithms and Parallel Paths

Here we consider shortest-path routing from a single node to a single node, and broadcasting from a single node to all the nodes in MCT. For permutation routing, algorithms developed in [1] may be easily adapted for the MCT architecture.

5.1 Shortest Path Routing

We recall the labeling of the vertices of $T(h)$ as defined in Section 2. The root of $T(h)$ is labeled 1. For any internal node labeled u, its left child is labeled $2u$ and its right child is labeled $2u + 1$. Based on these labels, a simple algorithm to find the first edge of the shortest path from a vertex u to a vertex v, $u \neq v$, can be derived [16]. Let u_b and v_b be the binary representations of the labels assigned to u and v, respectively, where leading zeroes have been removed. If u_b is not a prefix of v_b, the first edge of the path from u to v connects u with its parent vertex. If u_b is a prefix of v_b, then remove these bits from v_b and consider the leftmost remaining bit. If this bit is 0 the first edge of the path connects u and its left child. If the bit is 1 the first edge connects u and its right child. By applying this process in a greedy fashion the path between any pair of vertices in $T(h)$ can be found. For instance, Fig. 6 shows the path between the vertices $u_b = 100$ and $v_b = 1101$ in $T(4)$.

The algorithm to find the shortest path in $MCT_r(N)$ is a simple extension of the shortest path routing algorithm for $T(h)$. The shortest path from any node x to any node y in $MCT_r(N)$ is obtained by simply applying the shortest path routing algorithm described for the tree $T(h)$ along each dimension where the labels of x and y differ. By an argument similar to that used in the proof of Theorem 1, the reader can easily show that the path generated is a shortest path. In fact, several shortest paths may be found if we apply the above algorithm to the dimensions in different orders.

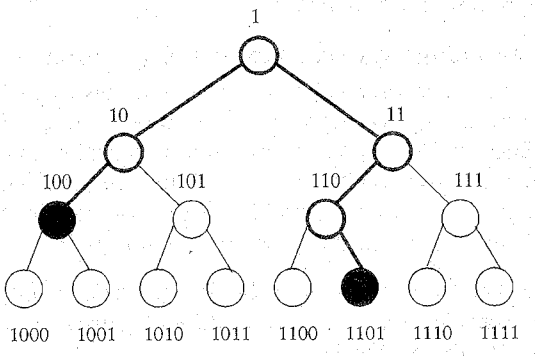


Fig. 6. Path in $T(4)$ between 100 and 1101. Leading zeros have been removed from the labels for clarity.

5.2 Broadcasting

The broadcasting of a message is the process of sending a message from a given node to every node in the network. Initially, the source node sends a copy of the message to the root node in its dimension-0 tree. The root then broadcasts the message in its dimension-0 tree. After at most $2(h - 1)$ steps each of the N nodes in the tree has a copy of the message. The same process is then repeated for dimension one trees. This second process takes $2(h - 1)$ steps and at the end of it N^2 nodes have a copy of the message. The process continues dimension by dimension and, after using all the dimensions, every node in the network has a copy of the message. The process takes at most $2r(h - 1)$ steps and the algorithm can be implemented centralized or distributed. If the "root" of the MCT (i.e., the node with label 1 ... 1) is the source of the broadcast operation, then the algorithm can be completed in half the time.

5.3 Parallel Paths

The number of disjoint paths between any two nodes is an indicator of fault-tolerance capabilities of a network. We show that there are m disjoint paths between any two nodes in the $MCT_r(N)$, where m is the lesser of the degrees of the two nodes. This result improves the result obtained by the direct application of Lemma 4 in [5] or Theorem 2 in [9].

THEOREM 9. *Every pair u, v of vertices in $MCT_r(N)$, where $r > 1$, is connected by exactly m vertex-disjoint paths, where m is the lesser of the degrees of the vertices u, v.*

PROOF. (Sketch) In the interest of brevity, we give a sketch of the basic idea of proof. A detailed proof that identifies these paths is available in [6]. We proceed by induction on the number of dimensions r. For $r = 2$, the claim can be verified by inspecting Fig. 1b. Suppose that the claim is true for $r - 1$ dimensions. The case for r dimensions follows by showing that each additional dimension increases the vertex degrees by at least one and at most three. More specifically, there is a new vertex-disjoint path for each increment of the vertex degree of the node with the lesser degree. □

Several facts follow immediately from this result.

1) Every pair of vertices in $MCT_r(N)$ is connected by at least r and at most $3r$ vertex-disjoint paths.
2) If $MCT_r(N)$ contains less than r faulty vertices, it is possible to find a path between any two fault-free vertices. (This result could have been obtained by direct application of Theorem 2 in [9]. The adaptive routing algorithm presented there may be used for the MCT graph with simple changes.)
3) Every pair of vertices in $MCT_r(N)$ is connected by at least r and at most $3r$ edge-disjoint paths.
4) $MCT_r(N)$ is $(r - 1)$-fault tolerant to faults in edges.

6 VLSI Layout Area

We show in this section that the VLSI layout area required for the MCT network is asymptotically same as the optimal area given for the MOT in [19], [13]. The layout of the MCT also meets the area required for grids when $r > 2$. As in the earlier papers, our analyses assume that the MCT network has a bounded vertex degree (i.e., r is fixed). This assumption is reasonable since a concrete implementation has a fixed number of dimensions, and this does not limit the modular growth of the network.

THEOREM 10. *$MCT_r(N)$ can be laid out in an area of $\Theta(N^{2(r-1)})$ for $r > 2$ and $\Theta(N^2 \log^2 N)$ for $r = 2$.*

PROOF. We start by proving the upper bound. It is enough to show that $MCT_r(N)$ is strongly $O\left(x^{\frac{r-1}{r}}\right)$-separable

and then apply Theorem 3.5 in [19] to obtain the results. This theorem states that an *N–node* strongly $O(x^{\alpha})$-separable graph can be laid out in a square of side $O(n^{\alpha})$ for $\alpha > 1/2$ and $O\left(n^{1/2} \log n\right)$ for $\alpha = 1/2$.

An *N–node* graph is said to be strongly $f(x)$-separable either if it has only one node or if by removing at most $f(n)$ edges it is divided into two graphs with the same number of nodes (within one), both strongly $f(x)$-separable. $MCT_r(N)$ can be separated into two graphs as above by removing the edges incident to any dimension-0 root. This isolates the roots of dimension-0, which are distributed evenly between the two subgraphs. The number of edges removed is $2N^{r-1} + (r-1)(N-1)N^{r-2} = O(N^{r-1})$. Given that the number of nodes is N^r, strong separability is satisfied in this step.

We can apply the same procedure to divide each subgraph in halves. Any edge incident to a dimension-1 root is removed and these roots are distributed between the two subgraphs. Again $O(N^{r-1})$ edges are removed from each of the $\frac{N^r}{2} = \Theta\left(N^r\right)$-node graph. This process can be repeated for each dimension. For dimension $r-1$, $O(N^{r-1})$ edges are removed from $\frac{N^r}{2^{r-1}} = \Theta\left(N^r\right)$-node graphs, while continuing to satisfy conditions of strong separability. After applying this process to the dimension $r-1$, we obtain 2^r subgraphs, each composed by an $MCT_r\left(\frac{N-1}{2}\right)$ graph and several isolated nodes. Then, the same process can be recursively applied until all the nodes are isolated. Therefore, $MCT_r(N)$ is shown to be strongly $O\left(x^{\frac{r-1}{r}}\right)$-separable and the upper bound of the claim follows.

We now prove the lower bound. For $r = 2$ it is shown in [19] that the $\frac{N+1}{2}$-leaf MOT requires an area of $\Omega\left(N^2 \log^2 N\right)$. Since such an MOT is a subgraph of $MCT_r(N)$ (recall Theorem 5), the later also requires an area of $\Omega\left(N^2 \log^2 N\right)$. The lower bound for $r > 2$ can be obtained by simply applying Theorem 5-1 in [13], first obtained in [18], which says that the area required by a network with bisection width B is $\Omega(B^2)$. □

If we denote the number of nodes of $MCT_r(N)$ as n, the above bounds can be rewritten as $\Theta(n^{2(r-1)/r})$, for $r > 2$, and $\Theta(n \log^2 n)$, for $r = 2$. These bounds are the same as those obtained for the MOT with bounded number of dimensions [19], [13] and for the grid with more than two dimensions [15]. These facts show that there is no additional asymptotical complexity cost incurred for the increased power of the MCT.

7 An Extension of the Basic Network

Consider connecting the leaves of the complete binary tree as shown in Fig. 7. We denote the resulting graph as $XT(h)$, where again h denotes the height of the tree. Justification for this extension is quite easy: In a modular implementation, all the nodes could be designed with the same number of I/O channels, and the unused channels at the leaves could be used to connect the leaves in this fashion. The next result shows that if we build the MCT graph with these trees as the factor graph, the resulting extended MCT network, denoted MCXT, contains the torus as a subgraph. This is a much better result than the embedding obtained in Section 4.1.

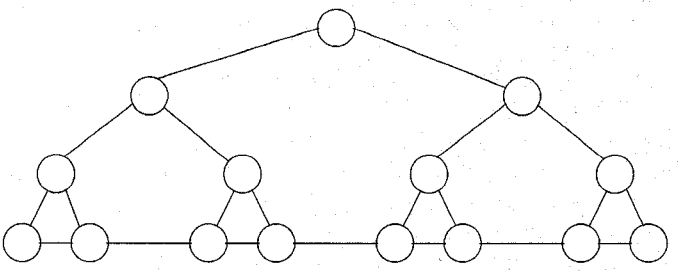


Fig. 7. Extending the complete binary tree by connecting the leaves.

THEOREM 11. *$MCXT_r(N)$ contains the N^r-node r-dimensional torus as a subgraph.*

PROOF. We show in the next lemma that $XT(h)$ contains a Hamiltonian cycle. The claim then follows from this result and Theorem 3 in [5]. □

LEMMA 5. *$XT(h)$ contains a Hamiltonian cycle.*

PROOF. We first show that $XT(h)$ contains the following Hamiltonian paths:

- Type-LL: Starts at one of the leftmost or the rightmost leaf and ends at the other.
- Type-LR: Starts at the leftmost leaf or the rightmost leaf and ends at the root.
- Type-RL: Starts at the root and ends at the leftmost leaf or the rightmost leaf.

If there is a Type-RL path that exits at the leftmost leaf, it can be converted into one which exits at the rightmost leaf. Once we show that these paths exist, we construct the claimed Hamiltonian cycle by recursively combining them together.

We proceed by induction on the height of the tree. For $h = 2$, $XT(2)$ is just a triangle and all three types of paths above are contained in it. Therefore assume that these paths exist in $XT(h-1)$, where $h > 1$. To study the case for $XT(h)$, note that the root of $XT(h)$ connects two such $XT(h-1)$ graphs. The Type-LL path for $XT(h)$ is obtained as: the Type-LR path in the left subtree of the root (entering from the left), followed by the root, followed by the Type-RL path (exiting from the right) in the right subtree. The Type-LR path is obtained as: the Type-LL path in the left subtree of the root, followed by the Type-LR path in the right subtree, followed by the root. The Type-RL path is obtained by reverse listing the Type-LR path.

This completes the proof that all three types of Hamiltonian paths exist in $XT(h)$ for any h. The Hamiltonian cycle for $XT(h)$ is obtained as: Type-LR path in the left subtree of the root (entering from the right), followed by the root, followed by the Type-RL path in the right sub-

tree (exiting from the left), and finally connecting this exit node to the entry node in the left subtree. □

8 Parallel Algorithms for the MCT Network

Although the MCT network is conceptually simple, it is able to execute a variety of algorithms very efficiently. First of all, there is a large number of algorithms developed for grids, and the ability of MCT to emulate grids with dilation cost three and congestion cost two implies that it can execute these grid algorithms with constant slowdown. Moreover, we know from Theorem 11 that when the extended tree structure of Fig. 7 is used as the factor graph, the corresponding product graph will contain the same-size grid as a subgraph. In this case, no slowdown is needed and all the grid algorithms in the literature can be directly executed on the proposed architecture. Since efficient algorithms that utilize the grid architecture are too numerous, we will not attempt to cite them here.

Besides grid algorithms, the class of computations that benefit from the proposed architecture are those that use frequent broadcasting capability from one node to the rest of the nodes in a dimension. Typical computations with this structure are graph algorithms such as minimum weight spanning trees, shortest paths, connected components, etc. On the N^r-node r-dimensional grid architecture, these computations require at least $\Omega(rN)$ running time due to the $\Theta(rN)$ diameter of the grid. The MOT is more efficient for these computations since it reduces the running time to $O(r \log N)$, matching the running time of hypercubes for similar computations (see [14] for a survey of these results). For the MCT architecture, one of the coauthors of this paper has investigated several algorithms in detail [7]. Due to space limitations, we summarize the results of [7] only briefly here.

Fig. 8 summarizes the running time for some of the algorithms considered in [7]. The result shown for graph algorithms applies for the computation of minimum-weight spanning trees, connected components, transitive closure, and all-pairs shortest paths. In all the cases in Fig. 8, the number of data points is equal to the number of processors, which is N^r.

Algorithm	Complexity
Summation of Numbers	$O(r \log N)$
Matrix Multiplication	$O(r \log N)$
Sorting	$O(r^2 N)$
Graph Algorithms	$O(r \log N)$

Fig. 8. Running times of various algorithms for the MCT network.

Summation of numbers is representative of a large class of computations requiring binary associative operators, such as AND, XOR, Min, Max, etc. Without increasing the asymptotic running time, the algorithm can be converted to prefix computations which has several other applications [12]. The given running time of $O(r \log N)$ is optimal since, due to the well known fan-in theorem, N^r numbers cannot be summed in less time than this.

Matrix-matrix multiplication algorithm has running time $O(r \log N)$, which is optimal and the same as the time required by other networks such as the MOT and the hypercube. It is also much more processor efficient than the MOT which requires $(r + 1) N^r - r N^{r-1}$ processors as opposed to N^r processors required by the MCT.

The MCT network is surprisingly efficient for sorting although it does not appear to have a structure suitable for sorting in the first instance. If r is fixed and N varies, the given running time reduces to $O(N)$ to sort N^r keys which is optimal for the MCT because the bisection width is $O(N^{r-1})$, and in the worst case $O(N^r)$ values may need to cross the bisection of the network. This running time matches that of the best known algorithms given for N^{r-1}-node r-dimensional grids when r is fixed. If N is fixed and r varies (e.g., in the hypercube $N = 2$ fixed but r varies), then the running time reduces to $O(r^2)$ which matches the asymptotic complexity of well known deterministic algorithms running on the hypercube. A more detailed discussion of sorting in the MCT as well as other product networks is given in our recent paper [8].

Graph algorithms cited above represent another important class of computations. It is shown in [7] that these problems can be solved in $O(r^2 \log^2 N)$ time in $MCT_r(N)$, with the adjacency matrix (or the weight matrix as the case may be) stored one entry per processor. The given running time matches those of the best known algorithms for the grid if r is fixed, and for the hypercube if N is fixed.

9 Conclusions

We have shown that the mesh-connected trees network is capable of performing algorithms developed for the torus, the grid, and the mesh of trees. This has considerable value because of the large number of algorithms that have been developed for these networks [14]. This capability also favors the mesh-connected trees in comparison to many of the other well known networks like the shuffle-exchange, de Bruijn, butterfly, cube-connected cycles, and the Benes networks which cannot host the grid with constant dilation [2], [14].

Although it is not more powerful than the hypercube, for all the SIMD algorithms we considered, the running time on the MCT matches the running time of well known hypercube algorithms. Its easier implementation and significant computational power make it more desirable when the full bandwidth of the hypercube is not required. For several computations, only a small number of hypercube links are used at a time. In such cases, the proposed MCT architecture matches the power of the hypercube, and the lower implementation cost makes it more desirable.

Finally, these advantages are obtained without any increase in the asymptotical complexity of the layout area over the grid (with at least three dimensions) or the MOT (with any number of dimensions), and with significantly less area complexity than the shuffle-exchange and de Bruijn graphs.

Acknowledgments

The authors wish to thank the anonymous referees for their detailed comments which improved the presentation of this paper significantly.

REFERENCES

[1] M. Baumslag and F. Annexstein, "A Unified Framework for Off-Line Permutation Routing in Parallel Networks," *Math. Systems Theory*, vol. 24, no. 4, pp. 233–251, 1991.

[2] S.N. Bhatt, F.R.K. Chung, J.-W. Hong, F.T. Leighton, B. Obrenic', A.L. Rosenberg, and E.J. Schwabe, "Optimal Emulations by Butterfly-Like Networks," *J. ACM*, 1993.

[3] S.N. Bhatt and I.C.F. Ipsen, "How to Embed Trees in Hypercubes," Technical Report RR-443, Department of Computer Science, Yale Univ., New Haven, Conn., 1985.

[4] K. Efe, "Embedding Mesh of Trees in the Hypercube," *J. Parallel and Distributed Computing*, vol. 11, pp. 222–230, Mar. 1991.

[5] K. Efe and A. Fernández, "Products of Networks with Logarithmic Diameter and Fixed Degree," *IEEE Trans. Parallel and Distributed Systems*, vol. 6, no. 9, pp. 963-975, Sept. 1995.

[6] K. Efe and A. Fernández, "Mesh-connected Trees: A Bridge Between Grids and Meshes of Trees," Technical Report 94-8-7, Center for Advanced Computer Studies, University of Southwestern Louisiana (also available from the first author).

[7] A. Fernández, "Homogeneous Product Networks for Processor Interconnection," PhD Thesis, Center for Advanced Computer Studies, Univ. of Southwestern Louisiana, Dec. 1994.

[8] A. Fernández, N.A. Eleser, and K. Efe, "Generalized Algorithm for Parallel Sorting on Product Networks," *Proc. 1995 Int'l Conf. Parallel Processing*, vol. III, pp. 155 – 159, Aug. 14-18 1995.

[9] T. El-Ghazawi and A. Youssef, "A General Framework for Developing Adaptive Fault-Tolerant Routing Algorithms," *IEEE Trans. Reliability*, vol. 42, pp. 250–258, June 1993.

[10] I. Havel and P. Liebl, "Embedding the Polytomic Tree into the *N*–cube," *Časpios pro Pěstován í Matematiky*, vol. 98, pp. 307–314, 1973.

[11] R. Heckmann, R. Klasing, B. Monien, and W. Unger, "Optimal Embeddings of Complete Binary trees into Lines and Grids," *Proc. 17th Int'l Workshop, WG'91, Graph-Theoretic Concepts in Computer Science*, G. Schmidt and R. Berghammer, eds., vol. 570 of *Lecture Notes in Computer Science*, pp. 25–35. Fischbachau, Germany: Springer Verlag, June 1991.

[12] R. Ladner and M. Fisher, "Parallel Prefix Computation," *J. ACM*, vol 27, no. 4, pp. 831-838, Oct. 1980.

[13] F.T. Leighton, *Complexity Issues in VLSI*. Cambridge, Mass.: MIT Press, 1983.

[14] F.T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, and Hypercubes*. San Mateo, Calif.: Morgan Kaufmann, 1992.

[15] C.E. Leiserson, "Area-Efficient graph Layout (for VLSI)," *Proc. 21st Annual Symp. Foundations of Computer Science*, pp. 270–281, Oct. 1980.

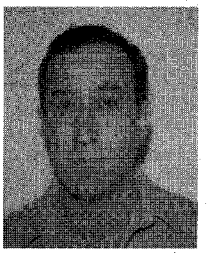
[16] *Fault-Tolerant Computing: Theory and Techniques*, D.K. Pradhan, ed., vol. 2, c. 7. Englewood Cliffs, N.J.: Prentice Hall, 1986.

[17] A.L. Rosenberg, "Product-Shuffle Networks: Toward Reconciling Shuffles and Butterflies," *Discrete Applied Mathematics*, vol. 37/38, pp. 465–488, July 1992.

[18] C.D. Thompson, "A Complexity Theory for VLSI," PhD thesis, Carnegie-Mellon Univ., Aug. 1980.

[19] J.D. Ullman, *Computational Aspects of VLSI*. Rockville, Md.: Computer Science Press, 1984.

[20] A. Youssef, "Product Networks: A Unified Theory of Fixed Interconnection Networks," Technical Report GWU-IIST-90-38, Inst. Information Science and Technology, Dept. Electrical Engineering and Computer Science, School of Eng. and Applied Science, George Washington Univ., Washington, D.C. 20052, Dec. 1990.

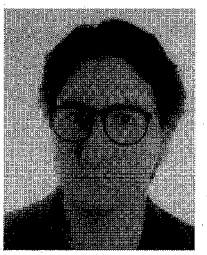


Kemal Efe received the BSc degree in electronic engineering from Istanbul Technical University, the MS degree in computer science from UCLA, and the PhD degree in computer science from the University of Leeds.

He is currently an associate professor of computer science in the Center for Advanced Computer Studies, the University of Southwestern Louisiana. Previously, he was on the faculty of the Computer Science Department, University of Missouri-Columbia.

Dr. Efe's research interests are in parallel and distributed computing in which he made many significant contributions. His expertise includes parallel algorithms and architectures, interconnection networks, distributed operating systems, distributed algorithms, performance evaluation, and VLSI complexity models. Dr. Efe served on technical committees of many international conferences and gave invited talks in the U.S., Europe, and Japan.

Dr. Efe received the "Certificate of Recognition" from NASA for his research contributions in 1995. He is a member of the ACM and IEEE.



Antonio Fernández received the degree of Diplomado en Informática in March 1988 and the degree of Licenciado en Informática in July 1991 from the Universidad Politécnica de Madrid. He received the MS degree in computer science in the fall of 1992 and the PhD degree in computer science in the fall of 1994 from the the University of Southwestern Louisiana, supported by a Fullbright Scholarship.

He is an associate professor in the Departamento de Arquitectura y Tecnologia de Computadores at the Universidad Politécnica de Madrid, where he has served on the faculty since 1988. He is currently on leave as a post-doctoral researcher at the Laboratory for Computer Sciences at MIT.

Dr. Fernández's research interests include parallel architectures and algorithms, interconnection networks, distributed systems, and data communication.