JOURNAL OF PARALLEL AND DISTRIBUTED COMPUTING **44**, 97–106 (1997)
ARTICLE NO. PC971356

Embeddings of Star Graphs into Optical Meshes without Bends

Stefan Thomas Obenaus* and Ted H. Szymanski†
*School of Computer Science and †Department of Electrical Engineering, McGill University, Montreal, Quebec H3A 2A7, Canada

We show an embedding of the star graph into a rectangular optical multichannel mesh of d dimensions such that the embedding has no bends; that is, neighbors in the star graph always differ in exactly one coordinate in the mesh, to facilitate one-hop optical communication. To embed an n-star, the mesh can have any number of dimensions d between 1 and $n-1$. The embedding has load 1 and an expansion of at most $n^{d-1}/d!$. The size of the mesh will be at most

$$\underbrace{n \times \cdots \times n}_{d-1} \times (n!/d!).$$

We optimize the size of the host mesh using clique-partitioning to produce embeddings with expansions as low as unity. In two dimensions, for even n, the mesh will be no larger than $n \times n(n-2)!$, and have an expansion of no more than $1\ 1/(n-1)$. Further, we show how we can use a contraction method to efficiently embed the star graph into an optical mesh with near-unity aspect ratios. Contraction on a two-dimensional embedding will yield a mesh of size no larger than $n \times n$ for even n with a load of $(n-2)!$. © 1997 Academic Press

1. INTRODUCTION

1.1. Motivation

Efficient embeddings of star graphs into optical multichannel meshes are proposed. Our motivation comes from the recent development of mesh-like optical networks with multiple channels in each dimension [4, 6, 10, 14, 15]. However, this paper is essentially a theoretical result applicable to any technology that favors rectangular embeddings without bends.

Optics is expected to impact future computing architectures by providing the capability of hundreds and potentially thousands of high bandwidth optical communication channels between processors. One attractive model for a computing architecture places the processors in a two-dimensional grid with tens or hundreds of reconfigurable and partitionable optical channels in each row or column [14, 16]. One advantage of this model is its physical realizability, since all optical channels are parallel and arranged in rows and columns. The optical channels in each row or column can be realized by multiplexing multiple wavelengths onto a single fiber [4, 10, 14]. Alternatively, the optical channels in each row or column can be realized by exploiting hundreds or thousands of microscopic optical beams emanating from a single optoelectronic device such as a *smart pixel array* [15, 16]. Finally, the optical channels can be realized by exploiting the temporal advantages of optics, i.e., the ability to clock optics at much higher rates then electronics, creating the means to support multiple electronic channels in the optical medium [6, 15].

The need for straight line connections, particularly in free-space optics, and the convenience of rectangular structures provided the motivation for finding a way to embed the n-star into a rectangular grid, say in two dimensions, such that neighbors in the graph are always in either the same row or the same column.

Akers *et al.* [1] introduced the star graph as an alternative to the hypercube. The n-star is an $n!$-node regular automorphic graph. Nodes are labeled with different permutations of n symbols. Nodes are neighbors if the label of one can be transformed into the label of the other by swapping the first symbol with one of the other symbols. Due to its small diameter ($\lfloor 3(n-1)/2 \rfloor$) and sublogarithmic degree ($n-1$), the star graph outclasses the hypercube in many aspects. See Day and Tripathi [3] for a comparative study.

A way of embedding meshes into the star graph has been shown by Ranka *et al.* [12]. Thus a star graph can simulate an n-dimensional mesh efficiently. We are proposing the opposite, an embedding of the star graph into a mesh.

Motivated by the need for embeddings of star graphs onto two-dimensional devices such as printed circuit boards, Hoelzman and Bettayeb [7] investigated the genus of a star graph. The genus of a graph determines the number of "bridges" that have to be placed on a two-dimensional surface to avoid edge crossings. They found that the star's genus is lower than a hypercube's and concluded that the layout of a star graph should be more efficient than the layout of a hypercube of similar size.

In contrast to board layouts, edge crossings are not a problem in free-space optics. However, unnecessary bends can be problematic. Up to this point, there has not been a convenient way to embed a star graph into a common rectangular physical device. We propose a way of embedding star graphs into two, three, or more dimensions, such that the

0743-7315/97 $25.00

positions of neighbors differ in one coordinate only. Hence, in a two-dimensional embedding, neighbors always share the same row or column.

1.2. Graph Embeddings

In [9], Leighton defines several terms to describe embeddings. When embedding one graph into another, we say, we embed the *guest graph* into the *host graph*. We call the ratio of the number of host graph nodes over the number of guest graph nodes the *expansion* of the embedding. The *dilation* of an embedding is the maximum path length in the host graph between neighbors in the guest graph. The maximum number of guest graph nodes that are embedded into the same host graph node is called the *load* of an embedding.

The embedding of an $(n-1)$-dimensional mesh into an n-star by Ranka *et al.* exhibits a load and expansion of 1 and a dilation of 3.

Our unoptimized embedding of the star graph into the d-dimensional mesh will have a load of 1, an expansion of at most $n^{d-1}/d!$, and a dilation of at most $n!/d! - 1$. Most importantly, the embedding will have no bends. We say that the embedding has a bend if and only if any of the host graph nodes corresponding to neighbors in the guest graph nodes differ by more than one coordinate in their labels. Finding an embedding of a graph into a mesh without bending edges and having load and expansion one is a known NP-complete problem called *Edge Embedding on a Grid* [5]. Our proposed embedding solves *Edge Embedding on a Grid* for selected even-n n-star embeddings in two-dimensional meshes. When the expansion-one condition is removed, our proposed embedding strategy solves *Edge Embedding on a Grid* for all n-stars in arbitrary dimensional meshes.

We also present two optimization methods, called *group optimization* and *contraction*, which can reduce the expansion and dilation considerably. In two-dimensional mesh embeddings, group optimization guarantees an expansion below 1 $2/(n-1)$ while maintaining unity load. Contraction increases the load without increasing the degree or introducing internal edges within vertices in the contracted node. At the same time, contraction improves the aspect ratio of the host mesh, and thus the dilation of the embedding. In two dimensions, the dilation is no more than $n+1$ while the aspect ratio is no greater than $n+1$: n. As well as improving the aspect ratio, the contraction operation we propose solves the known NP-complete problem *Graph Homomorphism* [5] for n-stars embedded into selected d-dimensional meshes. See [16] for a discussion of NP-complete problems related to embeddings in optical multichannel meshes.

1.3. Optical Model

Consider a mesh-based computing architecture, with n^d electronic nodes arranged in a d-dimensional array and with an optical medium interconnecting the n nodes in each row, column, or dimension. The electronic nodes can represent printed circuit boards, multichip modules, or integrated circuits. Hence, each node may contain multiple processing elements. Due to the bandwidth advantage of optics, the optical medium within a row or column can support multiple channels; in a typical configuration, each electronic node may reserve its own contention-free broadcast channel along a row and a column. This optical *multichannel mesh* architecture has been called a *hypermesh* [14].

This optical model assumes that the passing of a packet between any nodes along one dimension requires one *logical hop*. For example, passing a packet from one node to another node along a contention-free optical channel in the same row takes one hop regardless if the nodes are spatially nearest neighbors or at opposite ends of the row. When a guest graph is embedded into this optical mesh model, the number of dimensions traversed in an edge embedding is more relevant than the physical distance, since broadcasts along a row or column require one logical hop regardless of the destination's position within the row. Equivalently, the number of bends of an embedding is more important than its dilation. If an embedding of a guest graph edge into the optical multichannel mesh has k bends, then a packet requires $k+1$ hops to get from one guest node to its neighbor. In this model, the number of bends replaces the dilation as a distance measure of an embedding. As a result, the diameter of this optical mesh-like network can be much smaller than that of the electrical mesh [14]. In the worst case, it requires d logical hops to transmit a packet between the furthest apart nodes in an d-dimensional optical multichannel mesh model. In the conventional mesh model, where packets can only hop along host graph edges between physically nearest neighbors, it requires $d \cdot (n-1)$ logical hops to transmit a packet between the furthest apart nodes in the worst case.

In the remainder of this paper, we will first describe the star graph introduced by Akers *et al.* in [1] and the d-dimensional mesh. Then we will present and prove the embedding strategy. Finally, we show a few examples of embeddings.

2. MESH AND STAR GRAPH

A d-dimensional mesh $H = (V_H, E_H)$ of size $N = n_1 \times n_2 \times \cdots \times n_d$ has N nodes and extends n_i nodes into dimension i. The N mesh nodes in V_H are labeled with d coordinates; that is,

$$v \in V_H \tag{1}$$

$$\Leftrightarrow v = (v_1, v_2, \ldots, v_d) \quad \text{where } 0 \le v_i \le n_i - 1 \text{ and } 1 \le i \le d. \tag{2}$$

Two nodes $u, v \in V_H$ are neighbors, i.e., $\overline{uv} \in E_H$, if their labels only differ by one in exactly one coordinate [13]; that is,

$$\overline{uv} \in E_H \tag{3}$$

$$\Leftrightarrow \exists i \in \{1, 2, \ldots, d\}\ |u_i - v_i| = 1 \wedge \forall j \neq i\ u_j = v_j. \tag{4}$$

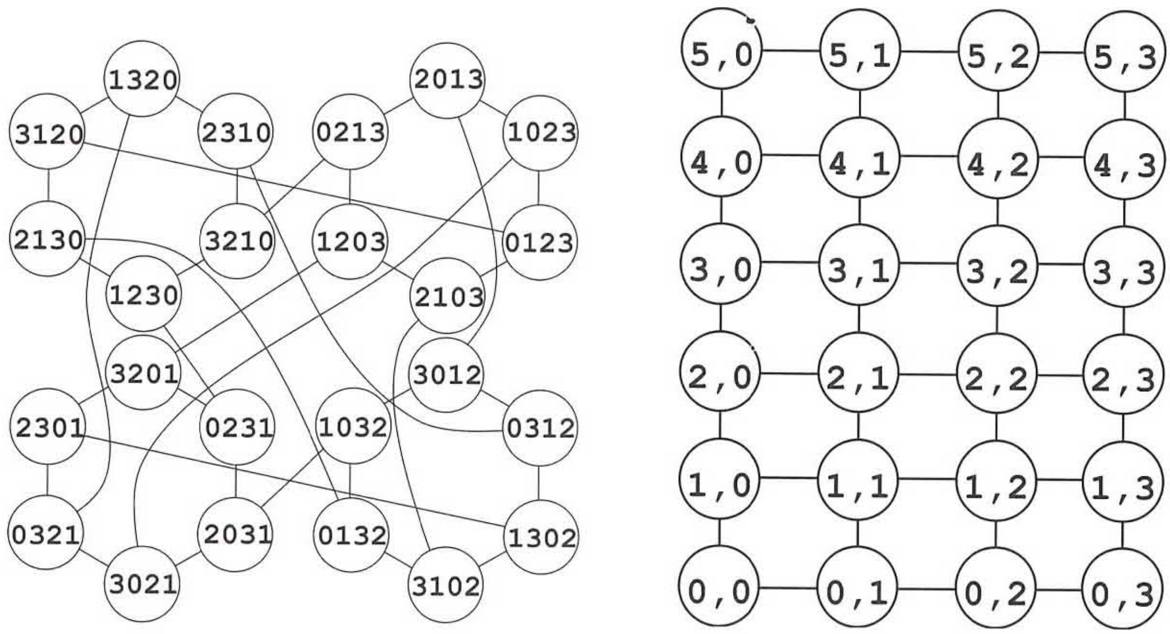

FIG. 1. The 4-star (left) and (4×6)-mesh.

An n symbol star graph $S = (V_S, E_S)$, or n-star as introduced in [1], is a graph with $N = n!$ nodes of degree $n - 1$. The $n!$ nodes are each labeled by a different permutation of n symbols from a set $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$. We will choose these symbols to be the numbers $0 \ldots n-1$. Formally, we say,

$$v \in V_S \tag{5}$$

$$\Leftrightarrow v = (v_1, v_2, \ldots, v_n) \wedge \{v_1, v_2, \ldots, v_n\} = \mathcal{S}. \tag{6}$$

As mentioned in Section 1.1, two nodes u, $v \in V_S$ are neighbors if the label of u can be transformed into the label of v by exchanging the first symbol of u's label with one of the $n - 1$ remaining symbols in its label; that is

$$\overline{uv} \in E_S \tag{7}$$

$$\Leftrightarrow \exists i \neq 1 \; u_1 = v_i \wedge \forall j \notin \{1, i\} \; u_j = v_j. \tag{8}$$

For example, node `23410` of the 5-star has the four neighbors `32410`, `43210`, `13420`, and `03412`. Figure 1 shows a 4-star and a (6×4)-mesh.

For comparison, respectively, $n!$-node star graphs and $n!$-node square meshes have degrees $n - 1$ and 4, diameters $\lfloor 3(n-1)/2 \rfloor$ and $2\sqrt{n!} - 2$, and average distances $n + 2/n - 4 + \sum_{i=1}^{n} 1/i$ and $2(\sqrt{n!} - 1/\sqrt{n!})/3$.

3. EMBEDDING

Until now, it has been unknown if the star graph could be embedded into an orthogonal structure such that all neighbors are in either the same row or column. Fortunately, we were able to determine such embeddings for arbitrary size star graphs and arbitrary dimensional host-meshes. Hereafter, we will assume that the mesh which is to host an n-star is of n or fewer dimensions. The mesh may have higher dimensions, but we will not utilize more than n dimensions.

Let us introduce and prove the embedding strategy in Section 3.1. In Section 3.2, we shall show a way of minimizing the expansion of the embedding through a method we call *group optimization*, and will show some example embeddings. Finally, in Section 3.3, we use a *contraction* method to improve the aspect ratio of the embedding to near unity.

3.1. Embedding Strategy

To construct an embedding, we will look at every star node and insert it into two appropriate sets. We call these sets *clusters* and *groups*. The label of a node will determine its group and cluster, which in turn will determine the node's position in the mesh. First, we describe how such an embedding is found. Then, we shall formally prove that such an embedding can always be found.

Assume we are dealing with an $n!$-node n-star, and assume we are trying to embed it into a rectangular mesh of d dimensions. In this case, we will put each node into one of $n!/d!$ groups, each containing $d!$ nodes, and into one of $n!/(n - d + 1)!$ clusters, each containing $(n - d + 1)!$ nodes.

3.1.1. Clusters

A cluster is a $(n - d + 1)$-substar within the n-star. For each node, its cluster is determined by the last $d - 1$ symbols in the label. For example, if we want to find the correct cluster for node v = `30421` given that we want to embed it into a three-dimensional mesh, then the last $d - 1$ symbols in label of v are `21`, and hence v belongs in cluster C_{21}.

More formally,

$$v \in C_{c_1 c_2 \ldots c_{d-1}}$$
$$\Leftrightarrow (v_{n-d+2}, v_{n-d+3}, \ldots, v_n) = (c_1, c_2, \ldots, c_{d-1}). \quad (9)$$

In order to specify a node's position in the d-dimensional mesh, we need to specify d coordinates. The first $d-1$ coordinates are determined from the cluster, and the last coordinate follows from the node's group.

We use the $d-1$ symbols that mark a cluster as the first $d-1$ mesh coordinates of the nodes in that cluster. Nodes in our example-cluster C_{21} will have 2 and 1 as their first two coordinates when embedded into the mesh host-graph. It should be noted that not all mesh positions will be associated with a cluster. Host graph nodes whose labels have duplicates in their first $d-1$ coordinates cannot be associated with any star graph nodes since it would imply that such a star graph node had duplicate symbols in its label. For example, host graph node `2,2,3,1` will always remain unoccupied by guest nodes because the label of a guest node would have to end with `223`. However, no permutation of n different symbols has any duplicates.

3.1.2. Groups

To find the last coordinate of a node, we look at its group. A node's group is determined by the $n-d$ symbols in positions 2 through $n-d+1$ in its label. For instance, $v =$ `30421` would be a member of group G_{04} because symbols 2 through $n-d+1$ in v's label are 0 and 4. The other members of G_{04} are `10423`, `10432`, `20413`, `20431`, and `30412`. Formally, we say

$$v \in G_{g_1 g_2 \ldots g_{n-d}}$$
$$\Leftrightarrow (v_2, v_3, \ldots, v_{n-d+1}) = (g_1, g_2, \ldots, g_{n-d}). \quad (10)$$

Since $|G| = d!$, we have $n!/d!$ groups. We arbitrarily assign an ordering to the groups and number them accordingly from 0 to $n!/d! - 1$. In the proof we will call this ordering mapping O. The group's index in the ordering will serve as the dth coordinate for all of the group's nodes. Thus, the group and cluster of a node determine all d coordinates of the node's embedding in the host graph. See Fig. 2 for a visualization of the mesh coordinate assignment procedure.

Now, we are ready to state the embedding as a theorem.

3.1.3. Proof

THEOREM 1. *Assume an n-star $S = (V_S, E_S)$ where all nodes are labeled using symbols from set $\mathcal{S} = \{0, \ldots, n-1\}$ and a d-dimensional mesh host graph $H = (V_H, E_H)$ of dimensions*

$$\underbrace{n \times \cdots \times n}_{d-1} \times (n!/d!).$$

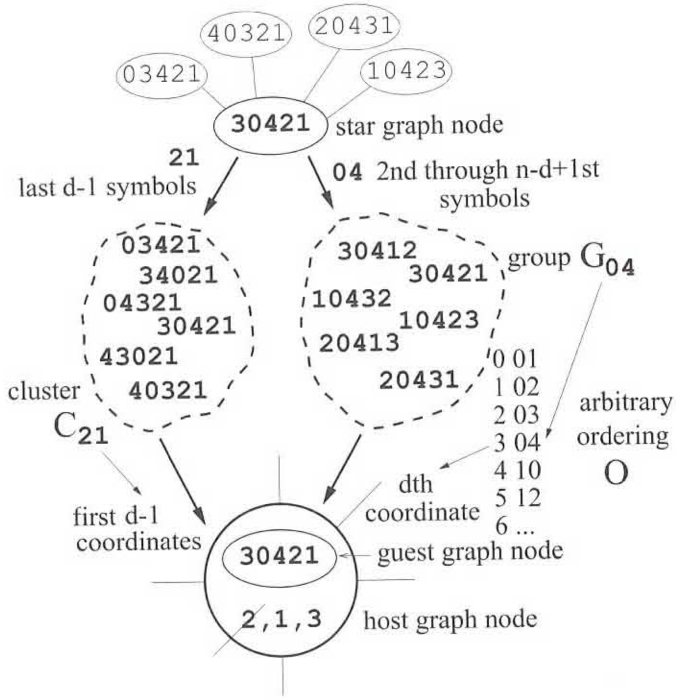

FIG. 2. Finding the coordinates in the host graph.

Further, consider a one-to-one mapping O: {all permutations of $n-d$ symbols from $\mathcal{S}$} $\rightarrow \{0, 1, 2, \ldots, n!/d! - 1\}$. Then S can be embedded into H with no bends and load 1. Such an embedding is achieved by embedding every node $v \in V_S$ into a node $u \in V_H$ such that

$$(u_1, u_2, \ldots, u_{d-2}, u_{d-1}, u_d) = (v_{n-d+2}, v_{n-d+3}, \ldots, v_{n-1}, v_n, O((v_2, v_3, \ldots, v_{n-d+1}))). \quad (11)$$

Proof. We call S the guest graph and H the host graph.

No two different nodes can belong to the same group and the same cluster since nodes with the same group symbols (symbols $2 \ldots n-d+1$) have different permutations of cluster symbols (symbols $n-d+2 \ldots n$), and vice versa. Hence, every guest graph node is embedded into a unique host graph node.

To show that there are no bends between guest graph neighbors in the host graph, we need to show that the labels of host graph nodes of neighbors in the guest graph differ by one coordinate only.

All members of one cluster differ in only one coordinate in their host graph labels since the cluster determines $d-1$ of d coordinates, which are common. Further, recall that the labels of neighbors of a star graph node are found by swapping the first symbol in a node's label with one of the $n-1$ other symbols. It follows that a node in a cluster C must have $n-d$ of its $n-1$ neighbors in C since $d-1$ neighbors can only be reached by changing one of the last $d-1$ symbols. But $n-d$ neighbors leave the last $d-1$ symbols unchanged and those neighbors are thus within the same cluster.

The remaining $d-1$ neighbors of a node v (which are found by swapping the first symbol with one of the $d-1$ last ones of node v) are in the same group as v since the labels of those neighbors have the same symbols 2 through $n-d+1$. v and its neighbors in the same group share the same dth coordinate in the host graph label. Moreover, these $d-1$ neighbors of v in v's group differ with v in only one of the last $d-1$ symbols, and since the last $d-1$ symbols are the first $d-1$ coordinates in the host graph, the labels of these guest graph neighbors differ by only one coordinate in the host graph. ■

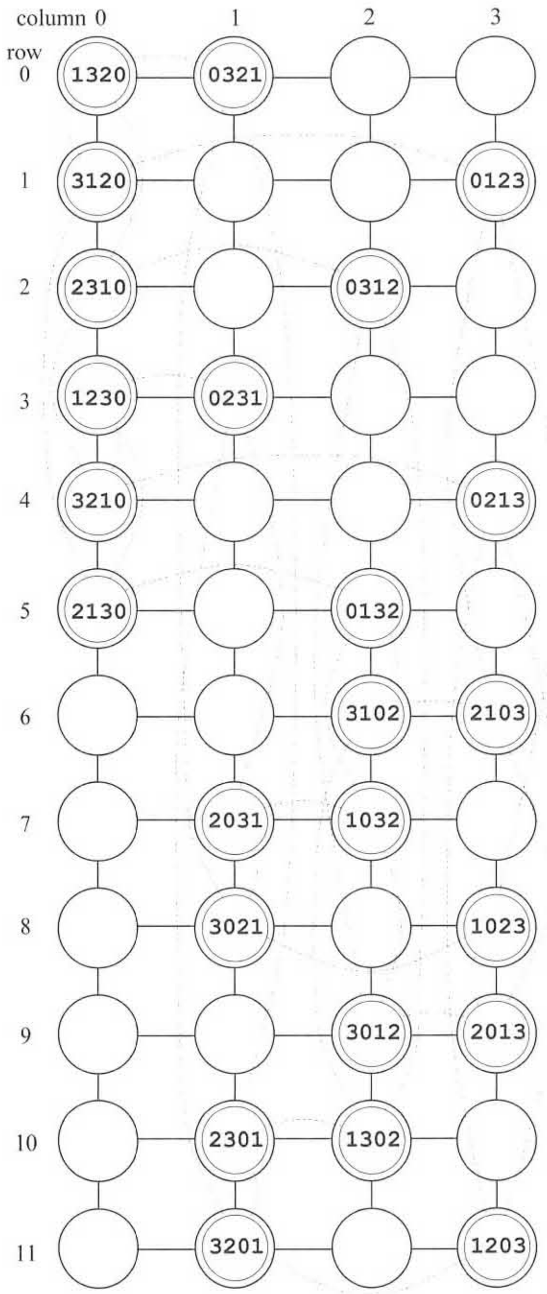

FIG. 3. Unoptimized embedding of the 4-star into a (4×12)-mesh as yielded by Theorem 1.

3.2. Group Optimization

According to Theorem 1 we embed the 4-star of Fig. 1 into a 4×12-mesh as shown in Fig. 3. In Fig. 3, every group occupies one row and some nodes are left "idle." We can improve the efficiency of the embedding by mapping two groups into every row in order to reduce the number of idle nodes, thus reducing the expansion of the embedding. Ideally, two groups will occupy distinct nodes and fully occupy all the nodes in a row, resulting in an embedding with unity load and unity expansion. To achieve this improvement, we need to find a perfect matching between pairs of groups. Figure 4 illustrates an embedding of the 4-star into a (4×6)-mesh (from the original embedding into a (4×12)-mesh shown in Fig. 3) obtained using a perfect group matching.

For general d-dimensional embeddings, we shall refer to a group's dth coordinate as its row, just as in the 2-dimensional case. We can put two groups with all their nodes onto the same row if none of their nodes claim the same mesh node, as determined by the nodes' clusters. As we will show in Theorem 2, it turns out that we can put those groups that differ by two or more symbols in their labels onto the same row.

In order to determine which groups can be placed on the same row in a group-optimized embedding, we shall place all groups with the same symbols into a *meta group* Γ such that

$$G_{g_1 g_2 \dots g_{n-d}} \in \Gamma_{\{\gamma_1, \gamma_2, \dots, \gamma_{n-d}\}} \Leftrightarrow (g_1, g_2, \dots, g_{n-d}) \text{ is a permutation of } \{\gamma_1, \gamma_2, \dots, \gamma_{n-d}\}. \quad (12)$$

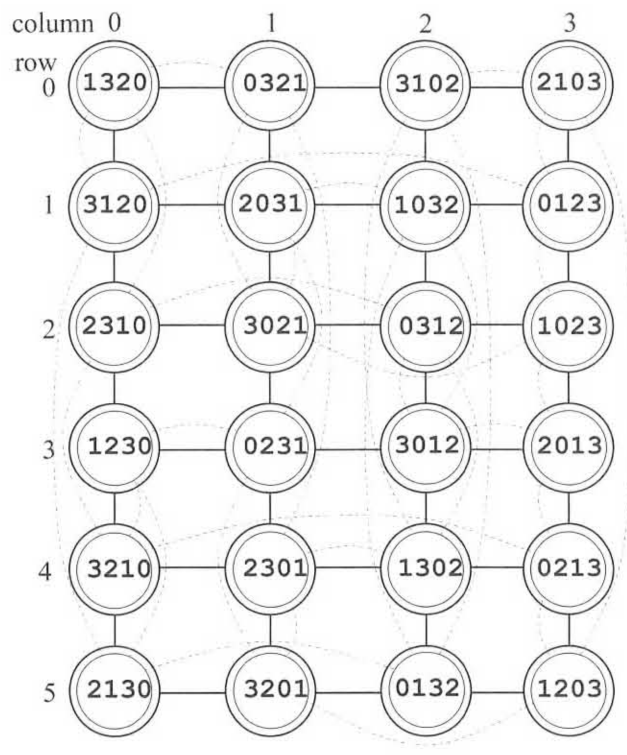

FIG. 4. Group-optimized embedding of the 4-star in a 2-d mesh.

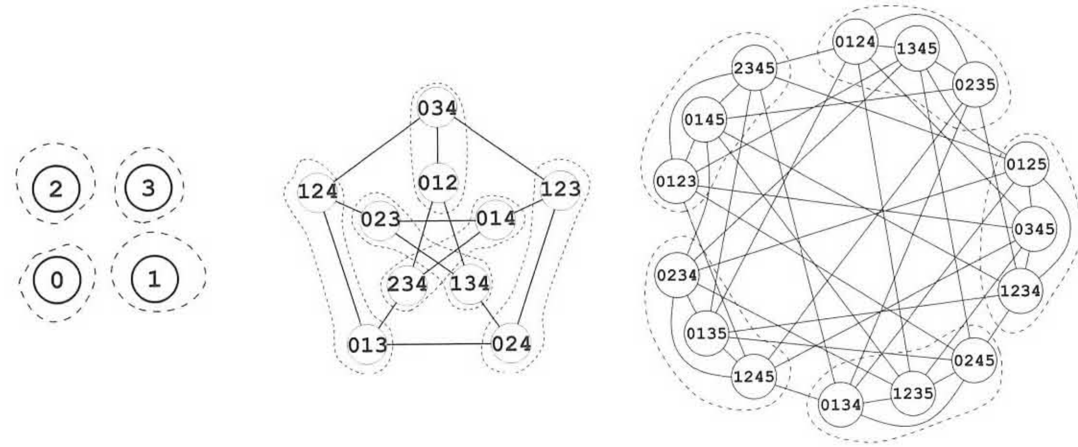

FIG. 5. Clique partitionings of $M_{4,3}$, $M_{5,2}$, and $M_{6,2}$.

Each meta group Γ holds $|\Gamma| = (n-d)!$ groups.

Groups that differ in two or more symbols can share a row, cf. proof of Theorem 2. Thus, we construct a *meta group graph* $M_{n,d} = (V_M, E_M)$ in which nodes represent meta groups and an edge is present whenever two meta groups differ in at least two symbols:

$$V_M = \{v : v \text{ is a set of } n-d \text{ symbols from } \mathcal{S}\} \qquad (13)$$

and

$$E_M = \{\overline{uv}\colon |u \cap v| \leq n-d-2\}. \qquad (14)$$

Now, we show how we can use a method called *clique-partitioning* and a meta group graph to group-optimize the embedding of a star graph. Note, that a meta group graph $M_{n,d} = (V_M, E_M)$ can always be partitioned into c cliques for some $c \leq |V_M|$.

THEOREM 2. *Given a clique-partitioning of a meta group graph $M_{n,d}$ into c cliques, we can embed an n-star into a d-dimensional mesh of size*

$$\underbrace{n \times \cdots \times n}_{d-1} \times c \cdot (n-d)!$$

with load one and no bends.

Proof. A clique partitioning of an $M_{n,d}$ is a partitioning of the nodes of $M_{n,d}$ into cliques, i.e., complete subgraphs, such that every node is a member of exactly one of these cliques.

Take any two groups G_1 and G_2 whose meta groups are neighbors in $M_{n,d}$. G_1 and G_2's symbols differ in at least two. Consequently, their respective clusters differ in at least one symbol (recall that

$$s_1 \underbrace{s_2 \ldots s_{n-d+1}}_{\text{group}} \underbrace{s_{n-d+2} \ldots s_n}_{\text{cluster}})$$

and so members of G_1 and G_2 can never occupy the same mesh node even if placed in the same row.

In a clique, all meta groups are neighbors, and hence we can embed one group from each meta group in a clique into the same row. There are c cliques and $(n-d)!$ groups per meta group. Thus, we can embed the n-star in a d-dimensional mesh of size

$$\underbrace{n \times \cdots \times n}_{d-1} \times c \cdot (n-d)!$$

with load one. ■

We observe that the expansion e is

$$e = \frac{n^{d-1} \cdot c(n-d)!}{n!}. \qquad (15)$$

It is desirable to partition $M_{n,d}$ into as few cliques as possible in order to minimize the expansion. However, clique partitioning is a well known and hard problem. Algorithms for clique partitioning exist, and a survey can be found in [11]. Optimal partitionings of $M_{4,3}$, $M_{5,2}$, and $M_{6,2}$ are shown in Fig. 5.

In two dimensions, we can determine an upper bound on the expansion by exploiting symmetry in the meta group graph $M_{n,2}$. First, we show how to partition an $M_{n,2}$.

LEMMA 3. *An $M_{n,2}$ can be partitioned into n cliques if n is even, and $n+1$ cliques if n is odd.*

Proof. The labels of $M_{n,2}$ consist of $n-2$ symbols. For simplicity, we may alternatively identify an $M_{n,2}$ node by the

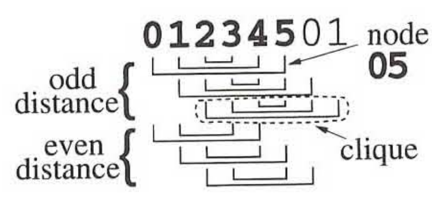

FIG. 6. A clique-partitioning of $M_{6,4}/M_{6,2}$.

TABLE I
Star Graph Embeddings in Two Dimensions

Star		2-dimensional mesh					
		Unoptimized		Group optimized		Contracted	
n	Nodes $n!$	size	expansion	size	expansion	size	load
4	24	4×12	2	4×6	1	4×3	2
5	120	5×60	$2\frac{1}{2}$	5×30	$1\frac{1}{4}$	5×5	6
6	720	6×360	3	6×120	1	6×5	24
7	5040	7×2520	$3\frac{1}{2}$	7×840	$1\frac{1}{6}$	7×7	120
8	40320	8×20160	4	8×5040	1	8×7	720
9	362880	9×181440	$4\frac{1}{2}$	9×45360	$1\frac{1}{8}$	9×9	5040

two symbols that are missing from its label. For example, node `0145` in $M_{6,2}$ can be uniquely identified by its missing symbols `23`. Using these new labels, $M_{n,2}$ becomes $M_{n,n-2}$. In fact, this symmetry holds for all d and clique-partitioning an $M_{n,d}$ is equivalent to clique-partitioning an $M_{n,n-d}$.

In an $M_{n,n-2}$, neighbors have no symbols in common. We select cliques by choosing pairs of symbols, corresponding to $M_{n,n-2}$ node labels, that do not overlap. For instance, symbol pairs `01`, `23`, and `45` form a clique. Let us define the distance between two symbols s_1 and s_2 as the smallest integer Δ such that $s_2 = (s_1 + \Delta) \bmod n$. Every symbol needs to pair up with $\lceil (n-1)/2 \rceil$ odd, and $\lfloor (n-1)/2 \rfloor$ even distance symbols in order to form the $\binom{n}{2}$ meta group graph node labels.

When n is even, we pair up symbols at odd distances $n-1, n-3, \ldots, 1$ to form $n/2$ cliques that cover all the odd distance symbol pairings. Similarly, we create $n/2$ cliques to cover all the even symbols pairings. See Fig. 6 for an example. In this way, we partition $M_{n,n-2}$ and consequently $M_{n,2}$ with n cliques.

When n is odd, we introduce a *ghost* symbol $n+1$ to make the number of symbols even and then proceed as in the even case. ■

Now, we use Lemma 3 to get an upper bound on the expansion.

THEOREM 4. *A group-optimized unity-load embedding with no bends of an n-star into a two dimensional mesh has an expansion of no more than* $1\ 1/(n-1)$ *if n is even, and* $1\ 2/(n-1)$ *if n is odd.*

Proof. The proof follows from Lemma 3 and the formula for expansion (15) where the number of cliques c is n or $n+1$ when n is even or odd, respectively. ■

Theorem 4 guarantees us that we can always find a near unity expansion embedding in two dimensions. In Tables I and II, we have compiled parameters of actual embeddings. Table I obeys the expansion bound of Theorem 4 and even suggests a tighter actual bound of unity if n is even, and $1\ 1/(n-1)$ if n is odd. Table II suggests that the expansion also approaches unity as the star graph size increases. Figures 7 and 8 show example embeddings of the 5-star in two dimensions and the 4-star in three dimensions.

In the next section we will show how the high aspect ratio of the larger embeddings can be reduced to yield aspect ratios of approximately unity.

TABLE II
Star Graph Embeddings in Three Dimensions

Star		3-dimensional mesh					
		Unoptimized		Group optimized		Contracted	
n	Nodes $n!$	size	expansion	size	expansion	size	load
4	24	$4 \times 4 \times 4$	$2\frac{2}{3}$	$4 \times 4 \times 4$	$2\frac{2}{3}$	$4 \times 4 \times 4$	1
5	120	$5 \times 5 \times 20$	$4\frac{1}{6}$	$5 \times 5 \times 10$	$2\frac{1}{12}$	$5 \times 5 \times 5$	2
6	720	$6 \times 6 \times 120$	6	$6 \times 6 \times 36$	$1\frac{4}{5}$	$6 \times 6 \times 6$	6
7	5040	$7 \times 7 \times 840$	$8\frac{1}{6}$	$7 \times 7 \times 168$	$1\frac{19}{30}$	$7 \times 7 \times 7$	24
8	40320	$8 \times 8 \times 6720$	$10\frac{2}{3}$	$8 \times 8 \times 840$	$1\frac{1}{3}$	$8 \times 8 \times 7$	120
9	362880	$9 \times 9 \times 60480$	$13\frac{1}{2}$	$9 \times 9 \times 5040$	$1\frac{1}{8}$	$9 \times 9 \times 7$	720

	14203	03412	34201	23410
	14023	03142	34021	23140
	12403	04312	32401	24310
	12043	04132	32041	24130
	10423	01342	30421	21340
	10243	01432	30241	21430
23104		43102	03241	13240
23014		43012	03421	13420
21304		41302	02341	12340
21034		41032	02431	12430
20314		40312	04321	14320
20134		40132	04231	14230
13204	04213		43201	34210
13024	04123		43021	34120
12304	02413		42301	32410
12034	02143		42031	32140
10324	01423		40321	31420
10234	01243		40231	31240
03214	24103	34102		43210
03124	24013	34012		43120
02314	21403	31402		42310
02134	21043	31042		42130
01324	20413	30412		41320
01234	20143	30142		41230
32104	42103	13402	23401	
32014	42013	13042	23041	
31204	41203	14302	24301	
31024	41023	14032	24031	
30214	40213	10342	20341	
30124	40123	10432	20431	

FIG. 7. Embedding of the 5-star into a 2-dimensional mesh.

3.3. Contraction

For large star graphs, the high aspect ratio of the mesh, e.g., 8×5040 for the 8-star in 2 dimensions, may complicate physical implementation. In this section, we will show how we can use contraction to efficiently improve the aspect ratio, and thus the dilation of the embedding, by reducing the number of rows in the embedding to be equal to the number of cliques in a clique-partitioned meta group graph. In two-dimensional embeddings, we will have no more than $n + 1$ rows, implying a dilation of at most n.

We can reduce the aspect ratio by contracting several nodes into one, thus increasing the load. Two caveats are required, though. First, contraction should not increase the degree of a node. Otherwise, the aspect ratio problem would merely be shifted from a large-size problem to a large-degree problem [2]. Second, contracted nodes should not have internal edges. If internal edges were present, potentially slow electrical intranode connections might have to replace otherwise fast optical internode links.

A solution to the second caveat in the contraction problem can once again be found in the meta group graph introduced in Section 3.2. Every star graph node is part of a group and meta group, which in turn make up the meta group graph, and within a meta group, all star graph nodes of a given cluster are isolated.

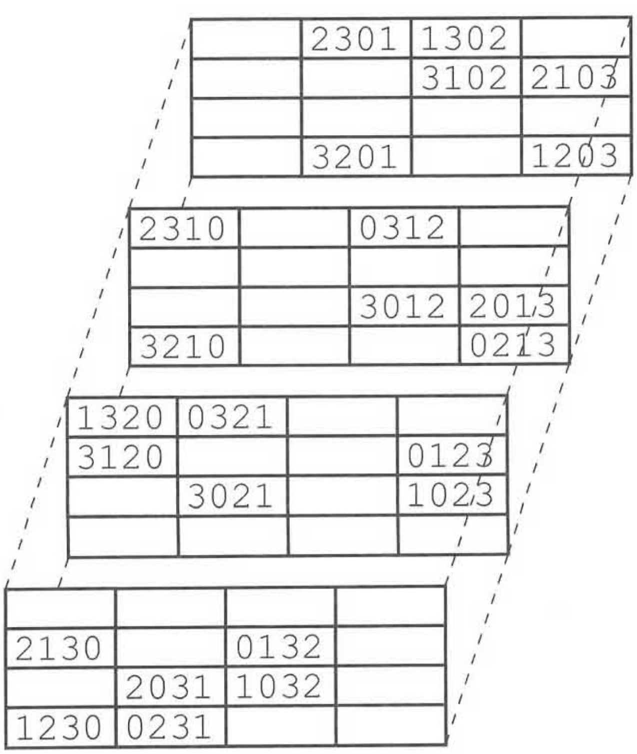

FIG. 8. 3-d embedding of the 4-star.

THEOREM 5. *Star graph nodes of the same cluster have no neighbors in their meta group.*

Proof. Let nodes $u = (u_1, \ldots, u_n)$ and $v = (v_1, \ldots, v_n)$ be any two distinct members of the same cluster and meta group. Then $(u_2, \ldots, u_n)$ is a permutation of $(v_2, \ldots, v_n)$. Consequently, $u_1 = v_1$, and u and v cannot be neighbors in the star graph. ■

By Theorem 5, we can contract all $(n - d)!$ star graph nodes of the same cluster and meta group into one mesh node without introducing any potentially slow electrical intranode connections. Further, we can show that although contraction may increase the load of the embedding substantially, the degree of the host node will remain at $n - 1$, i.e., the degree of the star graph.

THEOREM 6. *The degree of a contracted node containing all nodes of a meta group for a given cluster is equal to the degree of the individual nodes.*

Proof. For each $i = 1, \ldots, (n-d)!$, let u^i denote a subset of the nodes in a meta group $\Gamma_{\{\gamma_1, \ldots, \gamma_{n-d}\}}$ which share the same cluster. All nodes u^i are embedded in the contracted node u.

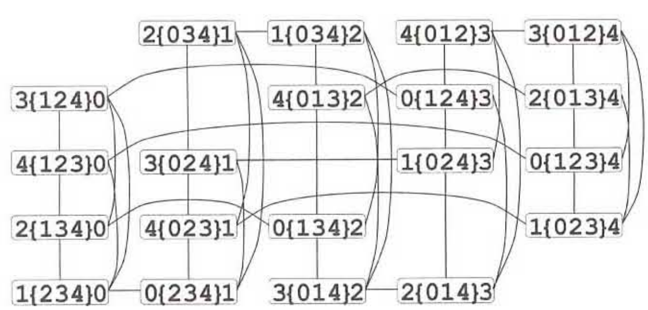

FIG. 9. Contracted embedding of a 5-star in two dimensions.

```
input: n,d          dimensions of star and mesh
      contraction   'true' if contraction is desired
Embed(n,d,contraction)
  Q=clique-partitioning of M_n,d
  row=0
  foreach clique C in Q
    foreach meta group Γ in C
      thisRow=row
      foreach group G in Γ
        foreach permutation (p_1,...,p_d) of {0,...,n-1}-{γ_1,...,γ_n-d}
          guestnode of (p_2,...,p_d ,thisRow)=(p_1,g_1,...,g_n-d ,p_2,...,p_d)
        if contraction=false then
          thisRow = thisRow+1
    if contraction=false then
      row=row+(n-d)!
    else
      row=row+1
```

FIG. 10. Group-optimized embedding algorithm with optional contraction.

For example, in Fig. 9, contracted node $u = (3, 0, 1, 2, 4)$ in the upper right corner holds nodes $u^1 = (3, 0, 1, 2, 4)$, $u^2 = (3, 0, 2, 1, 4)$, $u^3 = (3, 1, 0, 2, 4)$, $u^4 = (3, 1, 2, 0, 4)$, $u^5 = (3, 2, 1, 0, 4)$, and $u^6 = (3, 2, 0, 1, 4)$. These u^i are all part of cluster C_4 and members of meta group $\Gamma_{\{0,1,2\}}$ We label u by $(u_1, \gamma_1, \ldots, \gamma_{n-d}, u_{n-d+2}, \ldots, u_n)$. We observe that for all $i = 1, \ldots, (n-d)!$, $u_1^i = u_1$, and $(u_{n-d+2}^i, \ldots, u_n^i) = (u_{n-d+2}, \ldots, u_n)$. The cluster of all nodes in u is $C_{u_{n-d+2}, \ldots, u_n}$. Within $C_{u_{n-d+2}, \ldots, u_n}$, every u^i connects to a node with the same label only with symbols u_1^i and γ_k, where $k = 1, \ldots, n-d$, exchanged. Since $u_1^i = u_1^j = u_1$ for all nodes in $\Gamma_{\{\gamma_1, \ldots, \gamma_{n-d}\}}$ and in cluster $C_{u_{n-d+2}, \ldots, u_n}$, all γ_k-symbol edges of the nodes in u will connect to contracted node v of meta group $\Gamma_{\{u_1, \gamma_1, \ldots, \gamma_{n-d}\}-\{\gamma_k\}}$ in cluster $C_{u_{n-d+2}, \ldots, u_n}$. Intragroup edges will also connect to the same contracted node in a different cluster since the neighbors' meta group remains $\Gamma_{\{\gamma_1, \ldots, \gamma_{n-d}\}}$. ■

Since we contract all nodes of the same cluster in one meta group into one contracted node, we only require as many rows for our embeddings as there are cliques in a clique partitioned meta group graph. Thus, in two dimensions, by Lemma 3, we require no more than $n + 1$ rows.

A contracted embedding of the 5-star in two dimensions is shown in Fig. 9. The aspect ratio of the embedding is now reduced from 30 : 5 to 5 : 5 with a load of $(n-d)! = 6$. The degree of the contracted nodes is still $n - 1 = 4$.

Tables I and II show parameters of contracted embeddings for star graphs of up to 9! nodes. Figure 10 shows the algorithm for group-optimized embeddings with optional contraction.

4. CONCLUSION

Star graphs can be embedded in rectangular meshes in up to $n - 1$ dimensions without bends, thus allowing for one-hop optical communication between star graph neighbors in a rectangular implementation.

These embeddings exhibit load 1 and expansion of at most $n^{d-1}/d!$. Using clique partitioning, we were able to reduce the size of the host graph and produce embeddings with expansions as low as unity for two-dimensional embeddings and close to unity in three-dimensional embeddings. In general, the expansion of a group-optimized two-dimensional embedding is guaranteed not to exceed $1\ 2/(n-1)$.

By increasing the load in a contraction process, we were able to reduce the aspect ratio to values near unity without requiring potentially slow electrical intranode edges. Further, we managed to keep the degree of a contracted node at $n - 1$, the degree of the embedded n-star.

After Latifi and Bagherzadeh [8] overcame the scalability problem on the $n!$-node star graph, our embedding procedure eliminates one more obstacle that has hindered the practical use of star graphs.

ACKNOWLEDGMENTS

We are thankful for the support of the Natural Sciences and Engineering Research Council (NSERC) Canada Grant OGP0121601, and the Canadian Institute for Telecommunications Research (CITR) Grant 96-3-4.

REFERENCES

1. Akers, S. B., Harel, D., and Krishnamurthy, B. The star graph: An attractive alternative to the *n*-cube. *Proceedings of the International Conference on Parallel Processing*. 1987, pp. 393–400.
2. Barak, A., and Schenfeld, E. Embedding classical communication topologies in the scalable OPAM architecture. *IEEE Trans. Parallel Distrib. Systems* **7,** 9 (Sep. 1996), 962–978.
3. Day, K., and Tripathi, A. A comparative study of topological properties of hypercubes and star graphs. *IEEE Trans. Parallel Distrib. Systems* **5,** 1 (Jan. 1994), 31–38.
4. Dowd, P. Wavelength division multiple access channel hypercube processor interconnection. *IEEE Trans. Comput.* **42,** 10 (Oct. 1992), 1223–1241.
5. Garey, M. R., and Johnson, D. S. *Computers and Intractability.* Freeman, New York, 1979.
6. Guo, Z., Melhem, R., Hall, R., Chiarulli, D., and Levitan, S. Pipelined communications in optically interconnected arrays. *J. Parallel Distrib. Comput.* **12** (Oct. 1991), 269–282.
7. Hoelzeman, D. A., and Bettayeb, S. On the genus of star graphs. *IEEE Trans. Comput.* **43,** 6 (June 1994), 755–759.
8. Latifi, S., and Bagherzadeh, N. Incomplete star: An incrementally scalable network based on the star graph. *IEEE Trans. Parallel Distrib. Systems* **5,** 1 (Jan. 1994), 97–102.
9. Leighton, F. T. *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes.* Morgan Kaufman, San Mateo, CA, 1992.
10. Li, Y., Rao, S., Redmond, I., Wang, T., and Lohmann, A. Free space WDMA optical interconnects using mesh connected bus topology. *IEEE International Conference on Optical Computing,* Vol. WA2. 1994, pp. 191–192.
11. Pullman, N. J. Clique coverings of graphs—A survey. *Combinatorial Mathematics X—Proceedings of the Conference.* 1982, pp. 72–85.

12. Ranka, S., Wang, J.-C., and Yeh, N. Embedding meshes on the star graph. *J. Parallel Distrib. Comput.* **19,** 2 (Oct. 1993), 131–135.
13. Scherson, I. Orthogonal graphs for a class of interconnection networks. *IEEE Trans. Parallel Distrib. Systems* **2,** 1 (Jan. 1991), 3–19.
14. Szymanski, T. H. "Hypermeshes": Optical interconnection networks for parallel computing. *J. Parallel Distrib. Comput.* **26,** 1 (Apr. 1995), 1–23.
15. Szymanski, T. H., and Hinton, H. S. Reconfigurable intelligent optical backplane for parallel computing and communications. *Appl. Opt.* **35,** 8 (Mar. 1996), 1253–1268.
16. Szymanski, T. H., and Obenaus, S. T. Embedding properties of reconfigurable partitionable optical networks. In Berthome, P., and Ferreira, A., (Eds.). *Optical Interconnections and Parallel Processing: The Interface.* Kluwer Academic, 1997, to appear.

STEFAN THOMAS OBENAUS received his B.Sc. in physics and computer science from the University of British Columbia in 1993 and his M.Sc. in computer science from McGill University in 1995. Currently he is working on his Ph.D. in computer science at McGill University. His research interests include novel network topologies, network embedding methods, and their applications to reconfigurable intelligent optical backplanes and similar multichannel architectures.

TED H. SZYMANSKI received the B.Sc. in engineering science and the M.A.Sc. and Ph.D. in electrical engineering from the University of Toronto. From 1987 to 1991, he was an assistant professor at Columbia University and a principal investigator at the NSF Center for Telecommunications Research. He is currently an associate professor at McGill University, Director of the Microelectronics and Computer Systems Laboratory, and a project leader in the Canadian Institute for Telecommunications Research. An "Intelligent Optical Backplane" architecture developed by this project will be demonstrated in Canada in 1998. He is active professionally, and has been on the Technical Program Committees for the 1998 and 1997 Workshops on Optics in Computer Science, the 1998 and 1997 International Conferences on Massively Parallel Processing using Optical Interconnects, the 1998 International Conference on Optical Computing, the 1997 Innovative Systems on Silicon Conference, the 1995 Workshop on High-Speed Network Computing, and the 1998, 1995, and 1994 Canadian Conferences on Programmable Logic Devices. His research interests include ATM switching, gigabit networks, intelligent optical backplanes, and performance modeling.

Received September 26, 1995; revised June 9, 1997; accepted June 15, 1997