Multilayer VLSI Layout for Interconnection Networks

Chi-Hsiang Yeh
Dept. of Electrical & Computer Engineering
Queen's University
Kingston, Ontario, K7L 3N6, Canada
chi-hsiang.yeh@ece.queensu.ca

Emmanouel A. Varvarigos and Behrooz Parhami
Dept. of Electrical and Computer Engineering
University of California,
Santa Barbara, CA 93106-9560, USA
{manos, parhami}@ece.ucsb.edu

Abstract

Current VLSI technology allows more than two wiring layers and the number is expected to rise in future. In this paper, we show that, by designing VLSI layouts directly for an L-layer model, the layout area for a variety of networks can be reduced by a factor of about $(L/2)^2$ compared to the layout area required under a 2-layer model, and the volume and maximum wire length can be reduced by a factor of about $L/2$, leading to considerably lower cost and/or higher performance. The proposed layouts for k-ary n-cubes, hypercubes, butterfly networks, cube-connected cycles (CCC), folded hypercubes, generalized hypercubes, k-ary n-cube cluster-c, hierarchical hypercube networks, reduced hypercubes, hierarchical swap networks, and indirect swap networks, are the best layouts reported for these networks thus far and are optimal within a small constant factor under both the Thompson model and the multilayer grid model. All of our layouts are optimally scalable in that we can allow each network node to occupy the largest possible area (e.g., $o(N/L^2)$ for hypercubes) without increasing the leading constant of the layout area, volume, or maximum wire length.

1. Introduction

Twenty years ago many researchers believed that parallel processing would move to the mainstream of computation due to rapid advance in VLSI technologies. A variety of famous papers, theses, and books considered the VLSI layout of interconnection networks for parallel processing [7, 17, 19, 20, 22, 23, 24, 25, 27]. However, the revolution did not materialize at that time; rather, the increased VLSI density was used to build more complex single processors whose performance has improved by two orders of magnitude since then. As recently pointed out by Dally and Lacy [9], the number of transistors per chip will likely increase by another three orders of magnitude in the next two decades and few efficient alternatives to explicit parallelism exist for exploiting the increased number of transistors and grid points. Therefore, the expected revolution may begin soon and the mainstream computing community may shift from serial computers to parallel and distributed systems. The layout of interconnection networks has important cost and performance implications for single-chip multiprocessors and parallel/distributed systems based on such components. Thus, there is currently renewed interest in finding efficient VLSI layouts for various interconnection networks [3, 8, 10, 12, 13, 21, 28, 30, 31, 32, 35].

VLSI layout of interconnection networks is usually derived under the Thompson model, where two layers of wires are assumed. However, the assumption of two wiring layers cease to be realistic as more and more layers of wires become available in VLSI chips at reasonable cost. When the numbers of wiring layers and active layers (for network nodes) are both increased by a factor of t, the area of a layout designed for the Thompson model can be reduced by a factor of about t by folding the layout, while the volume and maximum wire length remain approximately the same. In this paper, we introduce the *multilayer 2-D grid* and *multilayer 3-D grid models* for VLSI layout of networks. We show that, for a wide variety of networks, including k-ary n-cubes, hypercubes, butterfly networks, cube-connected cycles (CCC) [22, 18], folded hypercubes [1], generalized hypercubes [5, 14], k-ary n-cube cluster-c [4], hierarchical hypercube networks (HHNs) [36], reduced hypercubes [37], hierarchical swap networks (HSNs) [33, 34], indirect swap networks (ISNs) [35], designing layouts under the multilayer 2-D grid model leads to the following advantages:

(1) the area of the layout can be reduced by a factor of approximately t^2 when we use $L = 2t$ layers of wires instead of two layers of wires as in the Thompson model

(2) the volume of the layout can be reduced by a factor of approximately t

(3) the maximum length of wires can be reduced by a factor of approximately t

(4) the maximum total length of wires along the routing path between any source-destination pair can be reduced by a factor of approximately t

For many other networks, including star graphs [2], transposition networks [16, 18], pancake graphs [2], bubble-sort graphs [2], and star-connected cycles (SCC) [15], the preceding arguments are still true, leading to lower cost and/or higher performance for most of the architectures considered thus far for parallel computation. The proposed layouts for butterfly networks, generalized hypercubes, HSNs, and ISNs are optimal within a factor of $1 + o(1)$ under the Thompson model, and are optimal within a factor of $2 + o(1)$ from a trivial lower bound under the multilayer grid model. These layouts and the proposed layouts for hypercubes, CCCs, folded hypercubes, reduced hypercubes, HHNs, and enhanced cubes constitute the best results reported in the literature for these networks, under both the Thompson model and the multilayer grid model.

The organization of the remainder of the paper is as follows. In Section 2, we discuss existing VLSI layout models, introduce the multilayer grid models that we propose, and propose several layout schemes. In Section 3 we present efficient multilayer layout for k-ary n-cubes, product networks, and related networks. In Section 4 we present efficient multilayer layouts for butterfly networks, generalized hypercubes, and related networks. In Section 5 we present efficient multilayer layouts for hypercubes, CCC, folded hypercube, and related networks. In Section 6 we present our conclusions.

2. VLSI layout models and layout schemes

In this section, we describe several models for VLSI layout of interconnection networks.

2.1. The Thompson model

In the Thompson model [23], a network is viewed as a graph whose nodes correspond to processing elements and edges correspond to wires. The graph is then embedded in a 2-D grid, where wires have unit width and a node of degree d occupies a square of side d. The wires can run either horizontally or vertically along grid lines. Two wires can cross each other at a grid point, but cannot overlap or bend at the same grid point, which would form a knock-knee [6].

The area of a layout is defined as the area of the smallest rectangle that contains all the nodes and wires. (In this paper we only consider upright rectangles for this purpose.) When there are two layers of wires and a node can be laid out in a square of area d^2, it is guaranteed that we can lay out the network within the area of that rectangle. More precisely, we can use one layer of wires to lay out all the horizontal segments of wires and the other layer to lay out all the vertical segments. When a wire makes a turn, its horizontal and vertical parts in different layers are connected by an inter-layer connector known as a via.

Note that some authors have assumed that a node occupies a square of side 1 in the layout model they use. Some such layouts cannot be extended to the Thompson model without a nonnegligible increase in area, while layouts under the Thompson model can usually be extended to the former model using comparable area.

2.2. The multilayer grid model

In the multilayer grid model, a network is viewed as a graph whose nodes correspond to processing elements and edges correspond to wires. The nodes and edges of the graph are then embedded in a 3-D grid, where edges have unit width, can run along grid lines, but cannot cross or overlap with each other (i.e., the paths for embedding these edges must be edge- and node-disjoint). The area A of a layout is defined as the area of the smallest upright rectangle along the x-y directions that contains all the nodes and wires. The volume of a layout is equal to the number L of layers times its area A.

In the multilayer 2-D grid model, the nodes of the graph are embedded in the 2-D grid of the first layer (i.e., $z = 1$). The range of actual node sizes must be specified explicitly in this model, and is usually taken to be between the minimum size required to implement a node (e.g., a square of side d, $d/4$, or $\frac{d}{4L}$ for a degree-d node in some technologies) and the maximum allowable size without affecting the leading constants for area, volume, and maximum wire length. A network with area A under the Thompson model can be laid out with area no larger than A under the multilayer 2-D grid model with $L = 2$ layers, so the former can be viewed as a special case of the latter. Note, however, that we may derive layouts under the two-layer 2-D grid model with area smaller than the Thompson model. In the multilayer 3-D grid model, the nodes of the graph are embedded in L_A layers of the 3-D grid. These L_A layers are called “active layers” and do not need to be consecutive layers. The range of actual node sizes is also required to be specified explicitly, which is usually between the minimum size required to implement a node (e.g., a cuboid with sides at least $d/h \times d/h \times h$, $1 \le h \le L_A \le L$, for a degree-d node in some technologies) and the maximum allowable size without affecting the leading constants for area, volume, and maximum wire length. The multilayer 2-D grid model is a special case of the multilayer 3-D grid model with $L_A = 1$ active layer. Note that a $d/h \times d/h \times h$ cuboid node requires h active layers for its implementation, while a $d \times d \times 1$ cuboid node requires only 1 active layer. The cost of a layout under the multilayer grid model is a function of A, L, and L_A, as well as other parameters.

The motivations for using multilayer layout models include the significant reduction achieved in the layout area, volume, and maximum wire length required, leading to considerable improvements in both hardware cost and performance. When we use L layers, the number of tracks in the x and y directions may both be reduced by a factor of about $L/2$ in many networks, leading to a factor of about $L^2/4$ reduction in its area compared to the layout under the Thompson model, and a factor of about $L/2$ reduction in its volume (since the number of layers is only increased by a factor of $L/2$). Hence, the cost of the resultant layout can be significantly reduced, or the performance can be significantly improved with the same hardware cost. As a point of comparison, if we fold a layout derived for the Thompson model in order to use all the available layers, the area can be reduced by a factor of only $L/2$, while the volume is unaffected; if we extend the collinear layout model to its multilayer counterpart, the volume will not change either since the area can only be reduced by a factor of at most $L/2$ when L layers are used. The maximum wire length in many networks is approximately proportional to the number of tracks in the x or y direction (or to their sum). Therefore, if the numbers of tracks in the x and y directions are both reduced by a factor of about $L/2$, the maximum wire length can also be reduced by a factor of approximately $L/2$, leading to significant improvement in performance. As a point of comparison, the maximum wire length in a collinear layout using L layers, or in a layout obtained by folding the layout derived using the Thompson model, is not significantly affected in most cases. These arguments will become clear after examining the multilayer layouts derived in the following subsections.

We can extend the multilayer grid model to the *multilayer layout model* by allowing nodes and edges to run in other specified directions. Layouts under this model may have smaller area and volume compared with layouts under its multilayer grid model counterpart. Moreover, wires in this model may have different width and cross area, depending on the technology used. For example, wires along the z direction may have larger cross area in PCB. In the remainder of the paper, we focus on the multilayer 2-D grid model. When the number L of layers is equal to 2, the multilayer layouts presented in this paper become layouts under the Thompson model. Note that, in general, a multilayer layout with $L = 2$ is not necessarily a layout under the grid model. Layouts under other models, such as the multilayer 3-D grid model and other multilayer layout models, will be reported in the near future.

2.3. The recursive grid layout scheme

In [28, 32], we have proposed the *recursive grid layout scheme* for simple and efficient 2-D layout of interconnection networks. In this subsection, we extend the scheme to the 3-D layout model and briefly present this generally applicable layout scheme.

To lay out an l-level hierarchical network, we first place nodes belonging to the same level-l cluster within a block, which we call a *level-l block*. We arrange the blocks as a 2-D grid for the 2-D layout model or as a 3-D grid for the 3-D layout model, where neighboring rows (or columns) are separated by a sufficient number of horizontal tracks (or vertical tracks, respectively) (see Fig. 1). We then lay out level-l inter-cluster links (i.e., links connecting nodes in different level-l clusters) outside the blocks. Note that we will eventually connect each of the level-l inter-cluster links incident to a level-l block to a certain node within the block. We can then continue to lay out each level-l cluster, including the M_{l-1} level-$(l-1)$ blocks within it and the links connecting these level-$(l-1)$ blocks, within a level-l block. This process is repeated recursively until each block contains a node or until the number of nodes within a block to be laid out is small. Then we use any viable method to lay out all these small clusters.

2.4. The orthogonal multilayer layout scheme

In this subsection, we propose a special case of the recursive grid layout scheme for multilayer layout of general interconnection networks.

In the *orthogonal multilayer layout scheme* we first partition network nodes into clusters and then arrange these clusters as a 2-D grid for the multilayer 2-D layout model or as a 3-D grid for the multilayer 3-D layout model. The partition and arrangement should be carefully performed so that (most of the) inter-cluster links only connect clusters belonging to the same row or column. Note that we in general prefer to make the clusters small if possible in order to reduce the additional area required to lay out these clusters, and a cluster may consist of a single node. We then lay out the inter-cluster links assuming two layers of wires (e.g., under the Thompson model, with one layer for horizontal tracks and the other for vertical tracks) so that the layout area, volume, and/or other cost/performance criteria (such as maximum wire length) are optimized. We refer to a 2-D layout as an *orthogonal layout* if all of its inter-cluster links connect clusters belonging to the same row or column. As will be shown in what follows, we can always transform an orthogonal layout to an efficient multilayer layout.

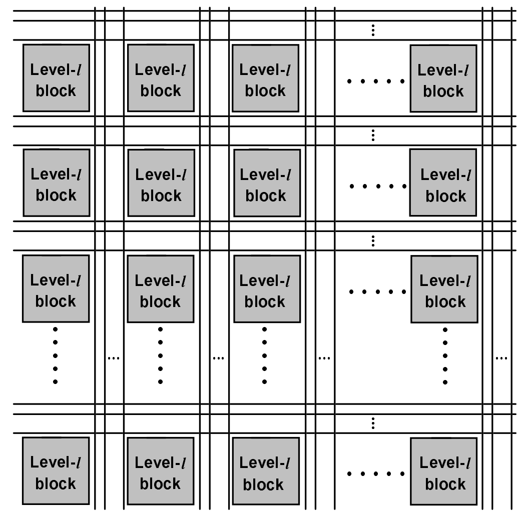

Figure 1. Top-view of a layout based on the recursive grid layout scheme. Level-l blocks are arranged as a 2-D grid.

Assume that the number of horizontal tracks required above row i of clusters is h_i, the number of vertical tracks required to the right of column j of clusters is w_j, and there are L layers of wires available. We partition the h_i horizontal tracks into $\lceil L/2 \rceil$ groups, each having at most $\lceil h_i / \lceil L/2 \rceil \rceil$ tracks, and the w_j vertical tracks into $\lfloor L/2 \rfloor$ groups, each having at most $\lceil w_i / \lfloor L/2 \rfloor \rceil$ tracks. To obtain an L-layer layout, we assign each group to a certain layer. For example, we can assign the groups for horizontal tracks to layers 1, 3, 5, ... , $2\lceil L/2 \rceil - 1$, and the groups for vertical tracks to layers 2, 4, 6, ... , $2\lfloor L/2 \rfloor$. If the cluster is small enough, which is the typical case for the layout of most interconnection networks, the layout area/volume is dominated by these inter-cluster links. The multilayer layout has then been mostly derived since it is easy to lay out small clusters without increasing the leading constant of layout area/volume. Let A be the layout area using two layers of wires. We can see that when L layers of wires are available, the area of the multilayer layout can be reduced by a factor of about $L^2/4$ and the volume can be reduced by a factor of about $L/2$.

If the cluster is very large and the number L of layers is not small, then we may have to lay out these intra-cluster links carefully. A possible method is to lay out these intra-cluster links recursively using the above method. The details are omitted in this paper.

3. Multilayer layout for k-ary n-cubes, product networks, and PN clusters

In this section, we present multilayer layout of k-ary n-cubes as an example to illustrate the multilayer grid model and the associated orthogonal multilayer layout scheme. We then extend the layout method to arbitrary product networks (also called Cartesian product graphs), k-ary n-cube cluster-c, and product network clusters (PN clusters).

3.1. Multilayer layout for k-ary n-cubes

To apply the orthogonal multilayer layout scheme to a k-ary n-cube, we first place node $(i_{n-1}, i_{n-2}, ..., i_0)$ at position (i, j) of a 2-D grid (i.e., each node is viewed as a cluster in the scheme), where

$$i = i_{n-1}k^{\lceil n/2\rceil-1} + i_{n-2}k^{\lceil n/2\rceil-2} + \cdots + i_{\lfloor n/2\rfloor+1}k + i_{\lfloor n/2\rfloor},$$

$$j = i_{\lfloor n/2\rfloor-1}k^{\lfloor n/2\rfloor-1} + i_{\lfloor n/2\rfloor-2}k^{\lfloor n/2\rfloor-2} + \cdots + i_1 k + i_0.$$

Then all links connect nodes belonging to the same row or column. It can be seen that each row is now connected as a k-ary $\lceil n/2\rceil$-cube, and each column is connected as a k-ary $\lfloor n/2\rfloor$-cube, so the 2-D layout problem is reduced to finding collinear layout of k-ary n-cubes, where a collinear layout is a layout derived by first placing all nodes along a line.

To describe the 2-layer collinear layout for a k-ary n-cube, we use a bottom-up approach, starting with a k-node ring (i.e., a k-ary 1-cube), and inductively moving to k-ary n-cubes of higher dimensions n. A collinear layout of a ring can be obtained by placing the k nodes along a row, connecting neighboring nodes through wires in the first track, and then connecting node 0 with node $k-1$, through a wire in the second track. Clearly, this layout requires 2 tracks. Assume that we have a collinear layout for a k-ary n-cube that requires $f_k(n)$ tracks. To obtain the collinear layout of a k-ary $(n+1)$-cube, we start with k copies of the layout of a k-ary n-cube. By increasing the horizontal space by a factor of k, we can place the i^{th} node of the j^{th} copy adjacent (from the right) to the i^{th} node of the $(j-1)^{th}$ copy, $i, j = 0, 1, ..., k-1$. We also increase the number of tracks (i.e., vertical space) to accommodate the $kf_k(n)$ tracks of the k collinear layout copies. Moreover, to connect the k copies of the k-ary n-cube into a k-ary $(n+1)$-cube, we need two extra tracks, one containing links between adjacent nodes (i.e., the i^{th} nodes of the k copies) and the other containing a wire connecting the ending nodes of the ring (i.e., the i^{th} nodes of the 0^{th} and $(k-1)^{th}$ copies). Figure 2 illustrates a resultant collinear layout for a 3-ary 2-cube. Therefore, the number of tracks required for the collinear layout of the k-ary $(n+1)$-cube is $f_k(n+1) = kf_k(n) + 2$. Since $f_k(1) = 2$, we have

$$f_k(n) = kf_k(n-1) + 2k^0 = k^2 f_k(n-2) + 2k^1 + 2k^0 = \cdots$$

$$= 2(k^{n-1} + k^{n-2} + \cdots + k^1 + k^0) = \frac{2(k^n-1)}{k-1} = \frac{2(N-1)}{k-1}.$$

If we connect nodes belonging to the same row (or column) as a 2-layer collinear layout of a k-ary n_1-cube (or k-ary n_2-cube, respectively), we obtain a 2-layer 2-D layout of a k-ary (n_1+n_2)-cube.

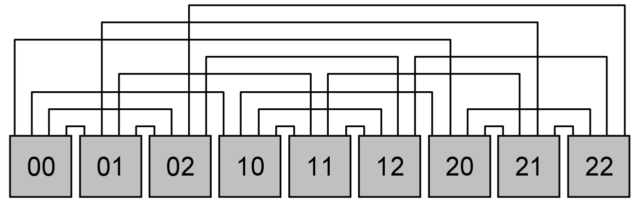

Figure 2. Collinear layout for a 3-ary 2-cube.

We then use the approach described for the orthogonal multilayer layout scheme to transform the 2-layer orthogonal layout to obtain an L-layer layout. When L is even, the number of tracks per layer above a row is $\lceil \frac{4(k^{\lfloor n/2\rfloor}-1)}{L(k-1)} \rceil$ and there are $k^{\lceil n/2\rceil}$ rows; the number of tracks per layer to the right of a column is $\lceil \frac{4(k^{\lceil n/2\rceil}-1)}{L(k-1)} \rceil$ and there are $k^{\lfloor n/2\rfloor}$ columns. Therefore, the area of the L-layer k-ary n-cube layout becomes

$$\frac{16N^2}{L^2k^2} + o\left(\frac{N^2}{L^2k^2}\right),$$

and the volume becomes

$$\frac{16N^2}{Lk^2} + o\left(\frac{N^2}{Lk^2}\right),$$

assuming that k is not a constant. To reduce the maximum wire length, we fold each row and column and the resultant maximum wire length becomes

$$O(\frac{N}{Lk^2}).$$

The area for an L-layer k-ary n-cube layout with odd L is

$$\frac{16N^2}{(L^2-1)k^2} + o\left(\frac{N^2}{L^2k^2}\right)$$

and the volume is

$$\frac{16N^2L}{(L^2-1)k^2} + o\left(\frac{N^2}{Lk^2}\right).$$

3.2. Multilayer layout for product networks, PN clusters, and k-ary n-cube cluster-c

The preceding multilayer layout method can be easily extended to general meshes and tori, and can also be further generalized to all product networks [11]. More precisely, for a product network $G = A \times B$, we can use the collinear layouts for the factor graphs A and B to lay out G. To do so, we simply arrange network nodes as a 2-D grid, and connect nodes belonging to the same row as a collinear layout of the factor graph A and nodes belonging to the same column as a collinear layout of the factor graph B. We then obtain a 2-layer orthogonal layout of the product network G, which can then be transformed to an L-layer layout using the techniques described for the orthogonal multilayer layout scheme. Clearly, this layout method is applicable to binary hypercubes and generalized hypercubes, special cases of product networks, as will be demonstrated in the following sections.

A network obtained by replacing each node of a product network with a cluster is referred to as a *product network cluster (PN cluster)*. In other words, the quotient graph obtained by shrinking each cluster of a PN cluster into a supernode will become a product network. In what follows we further extend the layout method to PN clusters. We can lay out such networks by first deriving an L-layer layout for the quotient graph, and then using the recursive grid layout method (Subsection 2.3) to lay out the clusters. More precisely, we expand each node (which corresponds to a supernode of the PN cluster) in the layout of the quotient graph into a rectangular block and arrange these blocks as a 2-D grid, where neighboring rows (or columns) are separated by a sufficient number of horizontal (or vertical, resp.) tracks (see Fig. 1). We then lay out the cluster within each of the blocks, and connect incident inter-cluster links from outside a block to network nodes within the block in the way specified by the topology. If the area increase due to the expansion of nodes in the quotient graph into rectangles (to lay out the clusters) does not dominate the area of the resultant layout, then the area of the PN cluster remains asymptotically the same as that of the quotient PN layout. Since the clusters and the nodes within the clusters are arranged as 2-D grids, a network node can occupy $o(\frac{\text{Layout Area}}{N})$ area without increasing the leading constants of the layout area, volume, and maximum wire length. For example, a hypercube node can occupy an area as large as $o(N)$ and a k-ary n-cube node can occupy an area $o(N/k^2)$ when L is a constant, instead of areas $\log_2^2 N$ and $4n^2 = 4\log_k^2 N$, respectively, as assumed in most previous papers. Such layouts (including all the layouts proposed in this paper) are optimally scalable in terms of node size since the leading constant of the layout area must become larger when network nodes are larger (i.e., with area $\Omega(\frac{\text{Layout Area}}{N})$).

Let us now consider a k-ary n-cube cluster-c [4] as an example of PN clusters. Assume that the clusters in the k-ary n-cube cluster-c are c-node hypercubes. Then a block with area $O(c^2/L^2)$ is sufficient to accommodate the c-node cluster and its inter-cluster links (see Section 5or [31]). Since these blocks are arranged as a $k^{n/2} \times k^{n/2}$ grid, the increase in area is negligible as long as the number c of nodes in a cluster is not very large; that is, $c = o(k^{n/2-1})$ so that $\frac{k^{n/2}c}{L} = o(\frac{k^n}{L(k-1)})$ (or $o(\frac{k^{n-1}}{L})$ when k is not a constant), which is the case except when c is large and/or n is small. Clearly, this conclusion applies to any k-ary n-cube cluster-c whose cluster is at most as dense as hyercubes. Similarly, we can show that even if the clusters are complete graphs, a k-ary n-cube cluster-c still has asymptotically the same area (witin a factor of $1+o(1)$) as a k-ary n-cube as long as $c = o(k^{n/4-1})$.

4. Multilayer layout for generalized hypercubes, butterflies, and related networks

In this section we present efficient multilayer layouts for generalized hypercubes, butterfly networks, HSNs, HHNs, and ISNs.

4.1. Multilayer layout for generalized hypercubes

A generalized hypercube [5] is the Cartesian product of two smaller generalized hypercubes. Therefore, we can use the layout method for product networks introduced in Subsection 3.2 to lay out generalized hypercubes.

To describe the 2-layer collinear layout for an n-dimensional radix-$(r_{n-1}, r_{n-2}, ..., r_1, r_0)$ generalized hypercube, we use a bottom-up approach similar to that for laying out k-ary n-cubes. We start with an r_0-node complete graph (i.e., a 1-dimensional radix-r_0 generalized hypercube), and inductively moving to generalized hypercubes of higher dimensions n. We have proposed a strictly optimal collinear layout for N-node complete graphs that requires $\lfloor N^2/4 \rfloor$ tracks (see Fig. 3 for an example) [30, 35]. Assume that we have a collinear layout for an n-dimensional generalized hypercube that requires $f_r(n)$ tracks. To obtain the collinear layout of an $(n+1)$-dimensional generalized hypercube, we start with r_n copies of the layout for an n-dimensional generalized hypercube. By increasing the horizontal space by a factor of r_n, we can place the i^{th} node of the j^{th} copy adjacent (from the right) to the i^{th} node of the $(j-1)^{th}$ copy, $i, j = 0, 1, ..., r_n - 1$. We also increase the number of tracks (i.e., vertical space) to accommodate the $r_n f_k(n)$ tracks of the r_n collinear layout copies. Moreover, to connect the r_n copies of the n-dimensional generalized hypercube into an $(n+1)$-dimensional generalized hypercube, we need $\lfloor r_n^2/4 \rfloor$ extra tracks to connect r_n adjacent nodes (i.e., the i^{th} nodes of the r_n copies) as a complete graph. Therefore, the number of tracks required for the collinear layout of the $(n+1)$-dimensional generalized hypercube is $f_r(n+1) = r_n f_r(n) + \lfloor r_n^2/4 \rfloor$, where $f_r(1) = \lfloor r_0^2/4 \rfloor$. It is easy to solve the recursive function $f_r(n)$ once the mixed-radix $(r_{n-1}, r_{n-2}, ..., r_1, r_0)$ is knowdni. For a radix-r generalized hypercube (i.e., $r_{n-1} = r_{n-2} = \cdots = r_0 = r$), we have

$$f_r(n) = rf_r(n-1) + r^0\lfloor r^2/4 \rfloor = r^2 f_r(n-2) + (r^1 + r^0)\lfloor r^2/4 \rfloor$$

$$= \cdots = (r^{n-1} + r^{n-2} + \cdots + r^1 + r^0)\lfloor r^2/4 \rfloor$$

$$= \frac{(r^n-1)\lfloor r^2/4 \rfloor}{r-1} = \frac{(N-1)\lfloor r^2/4 \rfloor}{r-1}$$

By connecting each row as an m-dimensional radix-$(r_{m-1}, r_{m-2}, ..., r_0)$ generalized hypercube and each column as an $(n-m)$-dimensional radix-$(r_{n-1}, r_{n-2}, \ldots, r_m)$ generalized hypercube using the preceding collinear layouts, we obtain a 2-layer orthogonal layout for an n-dimensional radix-$(r_{n-1}, r_{n-2}, ..., r_0)$ generalized hypercube. We then use the approach described for the orthogonal multilayer layout scheme to transform the 2-layer orthogonal layout to obtain an L-layer layout. For an n-dimensional radix-r generalized hypercube with an even number L of wiring layers, the number of tracks per layer above a row is $\lceil \frac{2\lfloor r^2/4 \rfloor (r^{\lfloor n/2 \rfloor}-1)}{L(r-1)} \rceil$ and

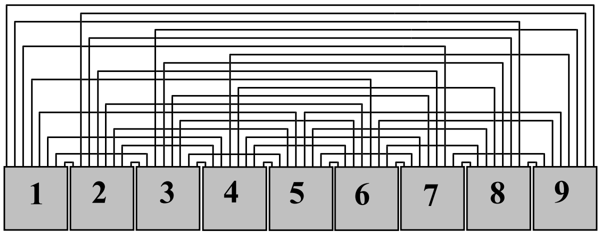

Figure 3. Collinear layout for a 9-node complete graph.

there are $r^{\lceil n/2 \rceil}$ rows; the number of tracks per layer to the right of a column is $\lceil \frac{2\lfloor r^2/4 \rfloor (r^{\lceil n/2 \rceil}-1)}{L(r-1)} \rceil$ and there are $r^{\lfloor n/2 \rfloor}$ columns. Therefore, the area of the L-layer generalized-hypercube layout becomes

$$\frac{r^2N^2}{4L^2} + o\left(\frac{r^2N^2}{L^2}\right).$$

The volume of the layout is

$$\frac{r^2N^2}{4L} + o\left(\frac{r^2N^2}{L^2}\right).$$

The maximum wire length is

$$\frac{rN}{2L} + o\left(\frac{rN}{L}\right) \textit{ and}$$

the maximum total length of wires along a shortest routing path is

$$\frac{rN}{L} + o\left(\frac{rN}{L}\right).$$

When L is odd, the area of the resultant generalized-hypercube layout is

$$\frac{r^2N^2}{4(L^2-1)} + o\left(\frac{r^2N^2}{L^2}\right),$$

assuming that r is not a constant.

4.2. Multilayer layout for butterfly networks

In [35], we have shown that by appropriately partitioning a butterfly network into clusters, these clusters can be connected as a generalized hypercube with multiple links. It is interesting, therefore, that butterfly network can be viewed as a PN cluster (i.e., a generalized hypercube cluster) and laid out using the approach introduced in Subsection 3.2. More precisely, we can partition an $R \times R$ butterfly network into $r(\log_2 R + 1)$-node clusters so that these clusters are connected as an $\frac{R}{r\log_2 R}$-node generalized hypercube where each pair of neighboring clusters are connected by 4 links, where $N = R\log_2 R$. Since a cluster only contains several copies of small butterfly networks, the layout area is dominated by inter-cluster links and is 16 times that of the area of an $\frac{R}{r\log_2 R}$-node generalized hypercube. Therefore, when L is even, the area of the resultant L-layer butterfly layout is

$$16 \times \frac{r^2(\frac{N}{r\log_2 N})^2}{4L^2} + o\left(\frac{r^2(\frac{N}{r\log N})^2}{L^2}\right)$$

$$= \frac{4N^2}{L^2\log_2^2 N} + o\left(\frac{N^2}{L^2\log^2 N}\right),$$

the volume of the L-layer layout is

$$\frac{4N^2}{L\log_2^2 N} + o\left(\frac{N^2}{L\log^2 N}\right),$$

and the maximum wire length is

$$\frac{2N}{L\log_2 N} + o\left(\frac{N}{L\log N}\right).$$

When the number L of wiring layers is odd, an N-node butterfly network can be laid out using area

$$\frac{4N^2}{(L^2-1)\log_2^2 N} + o\left(\frac{N^2}{L^2\log^2 N}\right).$$

More details can be found in [35].

4.3. Multilayer layout for hierarchical swap networks and related networks

An l-level hierarchical swap network (HSN) based on r-node nucleus graphs [33, 34] can be derived by replacing each node of an $(l-1)$-dimensional radix-r generalized hypercube with an r-node nucleus graph. Therefore, we can derive multilayer layout for HSNs using our layout for generalized hypercubes. When L is even, the area of the resultant L-layer layout for an N-node HSN is

$$\frac{r^2(N/r)^2}{4L^2} + o\left(\frac{r^2(N/r)^2}{L^2}\right) = \frac{N^2}{4L^2} + o\left(\frac{N^2}{L^2}\right),$$

the volume is

$$\frac{N^2}{4L} + o\left(\frac{N^2}{L}\right),$$

the maximum wire length is

$$\frac{N}{2L} + o\left(\frac{N}{L}\right),$$

and the maximum total length of wires along a shortest routing path is

$$\frac{N}{L} + o\left(\frac{N}{L}\right),$$

assuming that r is not a constant. When L is odd, the area of the resultant HSN layout is

$$\frac{N^2}{4(L^2-1)} + o\left(\frac{N^2}{L^2}\right).$$

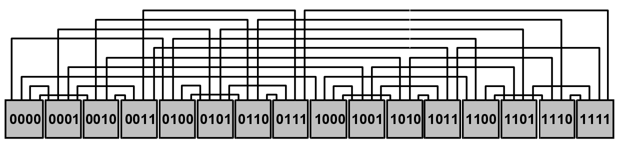

Figure 4. Collinear layout for a 4-cube.

The HSN layout can be easily generalized to general swap networks. Since hierarchical hypercube networks (HHNs) [36] are a special case of HSNs where the basic modules are hypercubes, they can be laid out in asymptotically the same area, volume, and maximum wire length (within a factor of $1+o(1)$).

Similar to butterfly networks, we can partition an $R \times R$ indirect swap network (ISN) [35] into $r(\log_2 R + o(\log R))$-node clusters so that these clusters are connected as an $(\frac{N}{r\log_2 N} + o(\frac{N}{r\log n}))$-node generalized hypercube with two links connecting each pair of neighboring clusters. Therefore, we can show that the multilayer layout for an ISN has area and volume smaller than those of a similar-size butterfly network by a factor of approximately 4, and the maximum wire length and the maximum total length of wires along a shortest routing path are smaller than those of a similar-size butterfly network by a factor of approximately 2.

We can use similar strategies to obtain efficient multilayer layouts for star graphs and other Cayley graphs [2], such as transposition networks [16, 18], pancake graphs [2], bubble-sort graphs [2], macro stars [29], and star-connected cycles (SCC) [15]. The details will be reported in the near future.

5. Multilayer layout for hypercubes, CCCs, and related networks

In this section, we present multilayer layouts for hypercubes, folded hypercubes, CCC, and reduced hypercubes.

5.1. Multilayer layout for hypercubes

An n-dimensional hypercube is the Cartesian product of two smaller hypercubes of dimensions $\lceil n/2 \rceil$ and $\lfloor n/2 \rfloor$. Therefore, we can use the layout method for product networks introduced in Subsection 3.2 to lay out hypercubes. We have proposed an efficient collinear layout that requires $\lfloor 2N/3 \rfloor$ tracks, derived by a bottom-up approach as that for k-ary n-cubes (Subsection 3.1) and generalized hypercubes (Subsection 4.1). The layout is based on a 2-track collinear layout for 2-cubes (rather than the 1-track collinear layout for 1-cubes) as the basic building block (see Fig. 4) [28, 31]. Therefore, we can show that the area of the resultant L-layer hypercube layout is

$$\frac{16N^2}{9L^2} + o\left(\frac{N^2}{L^2}\right);$$

the volume is

$$\frac{16N^2}{9L^2} + o\left(\frac{N^2}{L^2}\right);$$

and the maximum wire length is

$$\frac{2N}{3L} + o\left(\frac{N}{L}\right).$$

More details can be found in [31].

5.2. Multilayer layout for CCC and reduced hypercubes

An n-dimensional cube-connected cycles (CCC) graph is obtained by replacing each node in an n-cube with an n-node cycle [22]. A reduced hypercube, $\mathrm{RH}(\log_2 n, \log_2 n)$ [37], can be obtained by replacing each n-node cycle in a CCC with a $\log_2 n$-dimensional hypercube. Clearly, both of them can be viewed as PN clusters (i.e., hypercube clusters) and laid out using the method presented in Subsection 3.2.

To lay out a CCC, we first lay out an n-cube using the 2-D layout introduced in Subsection 5.1, and then lay out the n-node cycles within each of the hypercube nodes using the recursive grid layout scheme. Since the size of an n-dimensional CCC is $N = n2^n$ and its area is dominated by its hypercube links, which require $\frac{2^{n+4}}{9L^2} + o(2^n/L^2)$ area, an N-node CCC can be laid out in

$$\frac{16N^2}{9L^2\log_2^2 N} + o\left(\frac{N^2}{L^2\log^2 N}\right)$$

area when L is even. The layout area is better than that of a recently proposed CCC layout [8]. Using the same layout method, the reduced hypercube can be laid out in asymptotically the same area.

5.3. Multilayer layout for folded hypercubes and related networks

An enhanced-cube is a hypercube with one additional outgoing link per node leading to a random node [26]. A folded hypercube [1] is a hypercube with one additional link per node, where each node S has a link connecting it to the node whose label is the bitwise complement of S.

By adding additional links to our hypercube layout we can lay out folded hypercubes efficiently. More precisely, we first lay out an N-node hypercube in a square of side $\frac{2N}{3L} + o(N/L)$. To lay out an additional link, we need at most one additional vertical track and one additional horizontal track, besides the two ending segments connecting the link to two nodes. Since there are $N/2$ additional links in a folded hypercube, we need at most $N/2$ extra vertical and horizontal tracks to accommodate all the diameter links. These links can be partitioned into $L/2$ groups easily and laid out using L layers. Therefore, the area for the layout of a folded hypercube is

$$\left(\frac{7N}{3L} + o(\frac{N}{L})\right) \times \left(\frac{7N}{3L} + o(\frac{N}{L})\right) = \frac{49N^2}{9L^2} + o(N^2/L^2),$$

when L is even. Since there are N additional links in an enhanced-cube, we need at most N vertical and horizontal tracks to accommodate all the additional links. Therefore, the area for the layout of an enhanced-cube is $\frac{100N^2}{9L^2} + o(N^2/L^2)$. Some of these additional links may be placed in the same tracks so that the layout areas may be reduced.

6. Conclusion

In this paper, we introduced the multilayer grid model and showed that, for a variety of networks, the area can be reduced by a factor of $L^2/4$, and the volume and the maximum wire length can be reduced by a factor of $L/2$, relative to layouts using two layers of wires. The proposed layouts are the best layouts reported for these networks thus far and are optimal within a small constant factor under the multilayer grid model. The techniques introduced in this paper can also be applied to a variety of other networks.

References

[1] Adams, G.B. and H.G. Siegel, "The extra stage cube: a fault-tolerant interconnection network for supersystems," *IEEE Trans. Comput.,* vol. 31, no. 5, May. 1982, pp. 443-454.

[2] Akers, S.B. and B. Krishnamurthy, "A group-theoretic model for symmetric interconnection networks," *IEEE Trans. Comput.,* Vol. 38, Apr. 1989, pp. 555-565.

[3] Avior, A., T. Calamoneri, S. Even, A. Litman, and A. Rosenberg, "A tight layout of the butterfly network," *Theory Comput. Sys.,* vol. 31, no. 4, 1998, pp. 475-488.

[4] Basak, D. and D.K. Panda, "Designing clustered multiprocessor systems under packaging and technological advancements," *IEEE Trans. Parallel Distrib. Sys.,* vol. 7, no. 9, Sep. 1996, pp. 962-978.

[5] Bhuyan, L.N. and D.P. Agrawal, "Generalized hypercube and hyperbus structures for a computer network," *IEEE Trans. Comput.,* vol. 33, no. 4, Apr. 1984, pp. 323-333.

[6] Brady, M.L. and M. Sarrafzadeh, "Stretching a knock-knee layout for multilayer wiring," *IEEE Trans. Computers,* vol. 39, no. 1, Jan. 1990, pp. 148-151.

[7] Brebner, G., "Relating routing graphs and two-dimensional grids," *VLSI: Algorithms and Architectures,* 1985, pp. 221-231.

[8] Chen, G. and F.C.M. Lau, "Tighter layouts of cube-connected cycles," *IEEE Trans. Parallel Distrib. Sys, IEEE Trans. Parallel Distrib. Sys.,* vol. 11, no. 2, Feb. 2000, pp. 182-191.

[9] Dally, W.J. and S. Lacy, "VLSI architecture: past, present, and future," *Proc. Advanced Research in VLSI Conf.,* 1999, pp. 232-241.

[10] Dinitz, Y., S. Even, R. Kupershtok, and M. Zapolotsky, "Some compact layouts of the butterfly," *Proc. ACM Symp. Parallel Algorithms and Architectures,* Jun. 1999, pp. 54-63.

[11] Efe, K. and A. Fernandez, "Products of networks with logarithmic diameter and fixed degree," *IEEE Trans. Parallel Distrib. Sys.,* vol. 6, no. 9, Sep. 1995, pp. 963-975.

[12] Even, S., S. Muthukrishnan, M.S. Paterson, and S. Cenk Sahinalp, "Layout of the Batcher bitonic sorter," *Proc. ACM Symp. Parallel Algorithms and Architectures,* 1998, pp. 172-181.

[13] Fernández, A. and K. Efe, "Efficient VLSI layouts for homogeneous product networks," *IEEE Trans. Computer,* vol. 46, no. 10, Oct. 1997, pp. 1070-1082.

[14] Lakshmivarahan, S. and S.K. Dhall, "A new hierarchy of hypercube interconnection schemes for parallel computers," *J. Supercomputing,* vol. 2, 1988, pp. 81-108.

[15] Latifi, S., M.M. de Azevedo, and N. Bagherzadeh, "The star connected cycles: a fixed-degree network for parallel processing," *Proc. Int'l Conf. Parallel Processing,* Vol. I, 1993, pp. 91-95.

[16] Latifi, S. and P.K. Srimani, "Transposition networks as a class of fault-tolerant robust networks," *IEEE Trans. Parallel Distrib. Sys.,* Vol. 45, no. 2, Feb. 1996, pp. 230-238.

[17] Leighton, F.T., *Complexity Issues in VLSI : Optimal Layouts for the Shuffle-exchange Graph and Other Networks* Cambridge, Mass., MIT Press, 1983.

[18] Leighton, F.T., *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes,*Morgan-Kaufman, San Mateo, CA, 1992.

[19] Leiserson, C.E., *Area-Efficient VLSI Computation,* Cambridge, MA, MIT Press, 1983.

[20] Leiserson, C.E., "Fat-trees: universal networks for hardware-efficient supercomputing," *IEEE Trans. Computers*, vol. C-34, no. 10, Oct. 1985, pp. 892-901.

[21] Muthukrishnan, S., M.S. Paterson, S. Cenk Sahinalp, and T. Suel, "Compact grid layouts of some multi-level networks," *Proc. ACM Symp. Theory of Computing,* 1999, to appear.

[22] Preparata, F.P. and J.E. Vuillemin, "The cube-connected cycles: a versatile network for parallel computation," *Commun. ACM,* vol. 24, No. 5, pp. 300-309, May 1981.

[23] Thompson, C.D., "Area-time complexity for VLSI," *Proc. ACM Symp. Theory of Computing*, 1979, pp. 81-88.

[24] Thompson, C.D., "A complexity theory for VLSI," Ph.D. dissertation, Dept. of Computer Science, Carnegie-Mellon Univ., Pittsburgh, PA, 1980.

[25] Ullman, J.D., *Computational Aspects of VLSI,* Rockville, MD., Computer Science Press, 1984.

[26] Varvarigos, E.A., "Static and dynamic communication in parallel computing," Ph.D. dissertation, Dept. Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1992.

[27] Wise, D.S., "Compact layouts of banyan/FFT networks," *VLSI Systems and Computations,* Computer Science Press, 1981, pp. 186-195.

[28] Yeh, C.-H., "Efficient low-degree interconnection networks for parallel processing: topologies, algorithms, VLSI layouts, and fault tolerance," Ph.D. dissertation, Dept. Electrical & Computer Engineering, Univ. of California, Santa Barbara, Mar. 1998.

[29] Yeh, C.-H. and E.A. Varvarigos, "Macro-star networks: efficient low-degree alternatives to star graphs," *IEEE Trans. Parallel Distrib. Sys.,* Vol. 9, no. 10, Oct. 1998, pp. 987-1003.

[30] Yeh, C.-H. and B. Parhami, "VLSI layouts of complete graphs and star graphs," *Information Processing Letters,* Vol. 68, Oct. 1998, pp. 39-45.

[31] Yeh, C.-H., E.A. Varvarigos, and B. Parhami, "Efficient VLSI layouts of hypercubic networks," *Proc. Symp. Frontiers of Massively Parallel Computation*, Feb. 1999, pp. 98-105.

[32] Yeh, C.-H., B. Parhami, and E.A. Varvarigos, "The recursive grid layout scheme for VLSI layout of hierarchical networks," *Proc. Merged Int'l Parallel Processing Symp. & Symp. Parallel and Distributed Processing*, Apr. 1999, pp. 441-445.

[33] Yeh, C.-H. and B. Parhami, "The index-permutation graph model for hierarchical interconnection networks," *Proc. Int'l Conf. Parallel Processing*, Sep. 1999, pp. 48-55.

[34] Yeh, C.-H. and B. Parhami, "A unified model for hierarchical networks based on an extension of Cayley graphs," *IEEE Trans. Parallel Distrib. Sys.*, to appear.

[35] Yeh, C.-H., B. Parhami, E.A. Varvarigos, and H. Lee, "VLSI layout and packaging of butterfly networks," *Proc. ACM Symp. Parallel Algorithms and Architectures,* 2000, to appear.

[36] Yun S.-K. and K.H. Park, "Hierarchical hypercube networks (HHN) for massively parallel computers," *J. Parallel Distrib. Comput.,* vol. 37, no. 2, Sep. 1996, pp. 194-199.

[37] Ziavras, S.G., "RH: a versatile family of reduced hypercube interconnection networks," *IEEE Trans. Parallel Distrib. Sys.,* vol. 5, no. 11, Nov. 1994, pp. 1210-1220.