The Index-Permutation Graph Model for Hierarchical Interconnection Networks

Chi-Hsiang Yeh and Behrooz Parhami
Department of Electrical and Computer Engineering,
University of California,
Santa Barbara, CA 93106-9560, USA

Abstract

In this paper, we present the index-permutation (IP) graph model, and apply it to the systematic development of efficient hierarchical networks. We derive several classes of interconnection networks based on IP graphs to achieve desired properties; the results compare favorably with popular interconnection networks, as measured by topological (e.g., node degree and diameter) and algorithmic properties, and are particularly efficient in view of their sparse inter-module communication patterns. In particular, the diameters of suitably-constructed super-IP graphs, a subclass of IP graphs, are optimal within a factor of $1 + o(1)$*, given their node degrees. The IP graph model can also be used as a common platform that unifies the architectures and algorithms for a vast variety of interconnection networks.*

1 Introduction

In [4] Akers and Krishnamurthy presented a group-theoretic model called the *Cayley graph model* for designing, analyzing, and improving symmetric interconnection networks. Many subclasses of Cayley graphs are strongly hierarchical and have small diameters and node degrees. In particular, *k*-ary *n*-cubes, cube-connected cycles (CCC) [21, 22], and hypercubes are some well-known examples of Cayley graphs [3, 4]. In [4], Akers and Krishnamurthy showed that Cayley graphs are vertex-symmetric and that most vertex-symmetric graphs can be represented as Cayley graphs; it has also been shown that every vertex-symmetric graph can be represented as a *Cayley coset graph*. Both the Cayley graph model and the Cayley coset graph model have been used to derive a wide variety of interesting networks for parallel processing and have since received considerable attention [4, 5, 9, 14, 17, 19, 20, 29, 30]. In particular, the star graph [3] is a well-known Cayley graph that has a number of desirable properties, such as degree, diameter, and average distance smaller than those of a similar-size hypercube, symmetry, strong embedding capability, and fault tolerance properties.

In this paper, we present the *index-permutation (IP) graph model* for the systematic development of communication-efficient interconnection networks. In contrast to Cayley coset graphs [4, 17], which can represent any vertex-symmetric graph, we show that *any graph* has an IP graph automorphism. We focus on a subclass of IP graphs, called *super-IP graphs*, that use identical copies of a small network as their basic modules. Based on the notion of super-IP graphs, we present several efficient networks that can have node degree and/or diameter smaller than those of a similar-size star graph or hypercube and have strong embedding capability. We use the notion of IP graphs to derive symmetric and regular variants of super-IP graphs, called *symmetric super-IP graphs*. Our designs, based on super-IP graphs and symmetric super-IP graphs, compare favorably with other popular parallel architectures, as measured by topological (e.g., node degree and diameter) and algorithmic properties. In particular, the diameter of a suitably constructed (symmetric) super-IP graph can be optimal within a factor of $1 + o(1)$ from a universal lower bound, given its node degree. Moreover, the required data movements when performing many important algorithms on (symmetric) super-IP graphs are largely confined within basic modules, leading to small network delay when the delay associated with transporting a message through an on-module (e.g., on-chip or on-board) link is small. We define the *inter-cluster degree* as the maximum of the average-per-node off-module (or inter-cluster) links over all modules (or clusters), and the *inter-cluster diameter* as the maximum number of off-module (or inter-cluster) transmissions required for routing between two nodes. We compare the DD-cost (the product of degree and diameter [7]), ID-cost (the product of inter-cluster degree and diameter), and the II-cost (the product of inter-cluster degree and inter-cluster diameter) for super-IP graphs and several popular networks and show that super-IP graphs outperform other networks significantly under these figures of merit.

Due to the impact of inter-processor communication mechanism on the scalability and performance of parallel computers, numerous interconnection topologies have been proposed and intensely studied, forming a "sea of interconnection networks." Among them, certain classes of hierarchical networks, including hierarchical cubic networks (HCN) [15], hierarchical folded-hypercube networks (HFN) [13], hierarchical hypercube networks (HHN) [34], recursively connected complete (RCC) networks [16], and hierarchical shuffle-exchange (HSE) networks [10], have been shown to possess various appealing properties and are gaining considerable attention. Although these networks were proposed and studied independently and their structures may not resemble each other at first glance, we show in this paper and in [28, 33] that these networks and many other interconnection networks, including shuffle-exchange networks [21], hierarchical swap networks (HSN) [24, 25, 26] cyclic-shift networks [27, 32], super-flip networks, and CCC, belong to the class of super-IP graphs or symmetric super-IP graphs and share many properties and algorithms in common. Therefore, the IP graph model, a natural extension of the Cayley graph model [4], not only provide new insight to the design of novel communication-efficient networks, but also serves as a framework that ties together a vast variety of previously proposed interconnection topologies. Suitably constructed super-IP graphs can emulate a corresponding higher-degree network, such as a hypercube, with asymptotically optimal slowdown under various communication models. A variety of important network topologies can also be embedded in super-IP graphs with constant dilation. More details can be found in [28, 33].

The remainder of this paper is organized as follows. In Section 2, we present the index-permutation (IP) graph model. In Section 3, we discuss several efficient subclasses of super-IP graphs and their symmetric variants. In Section 4, we present routing algorithms for super-IP graphs and derive diameters of (symmetric) super-IP graphs. In Section 5, we consider implementation issues and compare the hardware and communication efficiency of several networks. Section 6 concludes the paper.

2 The Index-Permutation Graph Model

In this section, we introduce a mathematical game called the *ball-arrangement game (BAG)*. We then relate the game to the index-permutation graph model in an attempt to provide some intuition and to help in visualizing the model.

In the ball-arrangement game, we are given k balls, each stamped with a number. Different balls may be assigned the same or different numbers. The goal of the game is to rearrange the balls so that the numbers on the balls appear in a desired order. At each step the player can take an arbitrary action from a set of d permissible moves, each being a particular permutation of the balls. The set of permissible moves remains the same throughout the game, independent of the current configuration of the balls. There are $N \leq k!$ possible configurations of the balls (i.e., states) when playing the game, where N depends on the set of permissible moves and how balls are stamped with numbers initially. If we view each of the states as a network node and a permissible move leading from one state to another as a directed link connecting the nodes corresponding to those two states, then a network with N nodes results, where each node has d outgoing links. In other words, the network can be obtained by drawing the state transition graph for the corresponding ball-arrangement game with specified movements. One can then relate playing a ball-arrangement game to routing in the corresponding network, where the initial and final states correspond to the source and destination nodes and the movements performed to solve the game correspond to the links along the routing path. Since the in-/out-degree of the derived network is upper bounded by the number d of permissible movements and the diameter is the maximum number of steps required to solve the game, we generally prefer to select a small number of permissible moves that allow us to solve the game in a small (or optimal) number of steps for any initial and final states.

Recall that a Cayley graph is defined by a set of generators for a finite group, where the vertices correspond to the elements of the group and the edges correspond to the action of the generators [4, 8]. In the model proposed in [4], any element in the group is a permutation of a set of distinct symbols and generators are also permutations. For example, the label of a node in a 6-dimensional star graph (i.e., an element in the finite group) can be represented as $X = x_1x_2x_3x_4x_5x_6 = 123654$, and the generators of the 6-star are

$$\pi_1 = 213456 = (1,2), \;\; \pi_2 = 321456 = (1,3),$$

$$\pi_3 = (1,4), \;\; \pi_4 = (1,5), \;\; \text{and} \;\; \pi_5 = (1,6),$$

where a *cycle representation* (i, j) represents a permutation that interchanges symbols at positions i and j [3, 4]. Then the actions of generators lead to the following neighbors for node X:

$$X\pi_1 = \pi_1(X) = \pi_1(x_1x_2x_3x_4x_5x_6) = x_2x_1x_3x_4x_5x_6 = 213654,$$

$$X\pi_2 = \pi_2(X) = \pi_2(x_1x_2x_3x_4x_5x_6) = x_3x_2x_1x_4x_5x_6 = 321654,$$

$$X\pi_3 = 623154, \;\; X\pi_4 = 523614, \;\;\; \text{and } X\pi_5 = 423651.$$

Applying the same set of generators to these 5 neighbors generates 20 new neighbors. If we continue this process at least 7 times, we will obtain 720 distinct labels (including node X), which form the set of vertices in a 6-star (that is, all the elements in the corresponding finite group). We can see that any given Cayley graph corresponds to a certain ball-arrangement game, where each symbol corresponds to a ball that has a distinct number and the set of generators correspond to the set of permissible moves.

Just as a ball-arrangement game with distinct ball numbers can be used to derive a corresponding Cayley graph, an *arbitrary* ball-arrangement game can be used to derive a corresponding network, called an *index-permutation (IP) graph*, where each ball corresponds to a symbol and the set of permissible movements correspond a set of generators. In the definition of an index-permutation graph, there may be several identical symbols in the label of a node. Therefore, the index-permutation graph model can be viewed as an extension of the Cayley graph model where the restriction of "distinct" symbols for elements in the definition of a Cayley graph has been relaxed. More precisely, an IP graph is defined by a set of generators and a *seed element*, where a generator is a permutation, the edges correspond to the action of the generators, and the vertices correspond to the elements obtained by applying generators on the seed element or a generated element. For example, the label of the *seed node* in a certain IP graph might be represented as $Y = y_1y_2y_3y_4y_5y_6 = 123321$, and the generators of the IP graph can be permutations

$$\pi_1 = (1,2), \;\; \pi_2 = (1,3), \;\;\; \text{and } \pi_6 = 456123.$$

Then the actions of generators lead to the following 3 neighbors for node Y:

$$Y\pi_1 = \pi_1(Y) = \pi_1(y_1y_2y_3y_4y_5y_6) = y_2y_1y_3y_4y_5y_6 = 213321,$$

$Y\pi_2 = \pi_2(Y) = \pi_2(y_1y_2y_3y_4y_5y_6) = y_3y_2y_1y_4y_5y_6 = 321321,$
$Y\pi_6 = \pi_6(Y) = \pi_6(y_1y_2y_3y_4y_5y_6) = y_4y_5y_6y_1y_2y_3 = 321123.$
Repeatedly applying the 3 generators to generated nodes will result in 36 distinct nodes for this IP graph example. We will show that this simple extension is quite powerful and leads to novel and useful classes of parallel architectures.

In what follows we represent a few well-known networks using the IP graph model to make the idea clearer. HCN(n,n) [15] without diameter links can be viewed as an IP graph with the generator

$$T_{2,2n} = (2n+1)(2n+2)(2n+3)\cdots(4n)123\cdots(2n)$$

plus the n generators for an n-cube (that is, the generators with cycle representations $(1,2)$, $(3,4)$, $(5,6),\ldots$, $(2n-3,2n-2)$, and $(2n-1,2n)$) applied on the seed element

$$12\ 34\ 56\ \cdots\ (2n-1)(2n)\ \ 12\ 34\ 56\ \cdots\ (2n-1)(2n).$$

Note that, in contrast to distinct symbols in Cayley graphs, both halves of the seed element for the HCN use the same sequence of symbols. Consider HCN(n,n) with $n=2$. By applying the three generators

$$T_{2,4} = 5678\ 1234,\ \ (1,2),\ \text{and}\ \ (3,4)$$

on the seed 12 34 12 34, we first obtain three generated nodes 1234 1234, 21 341234, and 12 43 1234, respectively. Note that, in this example, the first generated node is the seed itself. Also note that the space between symbols is used only to better visualize the action of a generator or the applied action on a label. If we continue applying the generators on the derived nodes for another 4 iterations, we will finally obtain all the 16 nodes in an HCN$(2,2)$ (Fig. 1a). Note that using the label of any of the 16 nodes as the initial seed will eventually generate exactly the same graph. We can also use other types of labels for the seed to obtain a graph with exactly the same connectivity, even though the labels of network nodes will be different. For example, if we use 12 12 12 12 as the seed, we obtain a graph with the same connectivity (see Fig. 1a). For the original description of the construction of HCNs, we refer the reader to [15].

As another example, an n-dimensional de Bruijn graph, one of the densest known graphs, can be defined by generators

$$L_{1,2} = 34\ 56\ 78\cdots(2n-1)(2n)\ 12$$
$$\bar{L}_{1,2} = 34\ 56\ 78\cdots(2n-1)(2n)\ 21$$

applied to the $2n$-symbol seed $12\ 12\ 12\cdots 12$.

In [4], Akers and Krishnamurthy showed that Cayley graphs are vertex-symmetric and that most vertex-symmetric graphs can be represented as Cayley graphs; it was also shown that every vertex-symmetric graph can be represented as a *Cayley coset graph*. The analog of the preceding results for IP graph is given in the following theorem [28, 33]:

Theorem 2.1 *Any graph has an IP graph automorphism.*

In what follows, we will focus on super-IP graphs, IP graphs that use *super-generators*, which are a special class of permutations that interchange two or several sequences of symbols of equal-length. As an example, the generator $T_{2,2n}$ for an HCN(n,n) without diameter links is a super-generator that interchanges 2 sequences of $2n$ symbols.

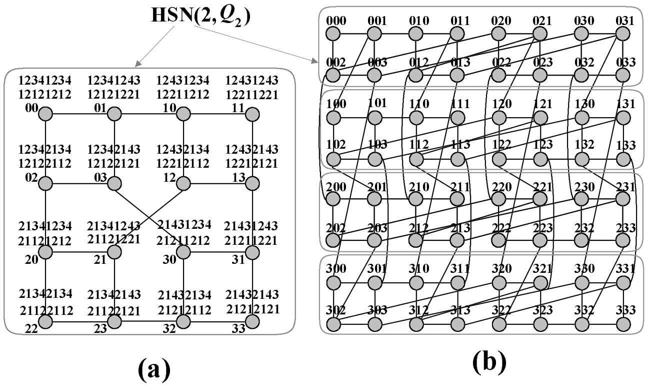


Figure 1. Structures of HSN(l,Q_2), $l=2,3$, represented with radix-4 node labels. (a) Structure and ranking of an HSN$(2,Q_2)$ = HCN$(2,2)$ without diameter links. (b) an HSN$(3,Q_2)$.

3 (Symmetric) Super-IP Graphs

In this section, we show how to derive communication-efficient networks based on super-IP graphs.

3.1 Definition of Super-IP Graphs

In this subsection, we give the definitions of super-IP graphs and introduce some related terminology.

Super-IP graphs are derived from ball-arrangement games with l (initially identical) boxes, each having m balls in it. The permissible moves in the game can be any permutation of the balls within the leftmost box or any permutation of the boxes (without reordering the balls within them). In other words, a super-IP graph is a special class of IP graphs where the seed label consists of l identical groups (boxes) of m symbols (balls) and the generators of the IP graph can either permute the m symbols (the m balls) within the leftmost group (box) or permute the m-symbol groups (m-ball boxes) without changing the order of symbols (balls) within any of the groups (boxes). We call each of the m-symbol groups in the label a *super-symbol*. The generators that permute the symbols within the leftmost super-symbol are called *nucleus generators* and the generators that permute super-symbols are called *super-generators*. For example, with the seed label 123 123, the permutation 321 456, which permutes 123 123 to become 321 123, defines a nucleus generator, whereas the permutation 456 123, permuting 321 123 to 123 321, corresponds to a super-generator. For each of the super-symbols, there must exist a sequence of super-generators that can bring it to the leftmost position. The small IP graph whose seed label is a super-symbol of the seed label of a super-IP graph and whose generator set consists of all the nucleus generators of the super-IP graph is called the *nucleus graph* of the super-IP graph. Since the nucleus determines the nucleus generators and the seed of the super-IP graph, a super-IP graph can be completely specified by its super-generators and its nucleus.

If we place each of the nuclei of a super-IP graph within the same module, then its inter-cluster degree, upper bounded by the number of off-module links per node in the super-IP graph, is no larger than the number of its super-generators, leading to the following theorem.

Theorem 3.1 *The degree of an IP graph is no larger than the number of its generators, and its inter-cluster degree is no larger than the number of its super-generators.*

In this paper, l always signifies the number of super-symbols in the label of a node in a super-IP graph, m always signifies the number of symbols in a super-symbol, and N represents the number of nodes in a network. The size of super-IP graphs is given in the following theorem [28, 33].

Theorem 3.2 *The size of a super-IP graph is $N = M_N^l$, where M_N is the number of nodes in the nucleus graph.*

3.2 Transposition Super-Generators

In this subsection we introduce several communication-efficient networks based on a special class of generators called *transposition super-generators*, each of which exchanges a pair of super symbols. The transposition super-generator that swaps the first and the i^{th} m-symbol groups (super-symbols of length m) will be denoted by $T_{i,m}$.

An l-level *hierarchical swap network* (also called *hierarchical swapped network*) [24, 25, 26], HSN(l,G), is a super-IP graph that has the seed $\underbrace{S_1S_1\cdots S_1}_{l}$, the generators for the nucleus G, and the transposition super-generators $T_{2,m} = (1,2)_m$, $T_{3,m} = (1,3)_m$, $T_{4,m} = (1,4)_m$,..., $T_{l,m} = (1,l)_m$, where S_1 is the label of a (seed) node in the nucleus G and $(1,i)_m$ represents the permutation that interchanges the first super-symbol and the i^{th} super-symbol. The subscript m in $(i,j)_m$ denotes the fact that super-symbols of length m at positions i and j are interchanged. In other words,

$$T_{2,m}(Y) = T_{2,m}(Y_1Y_2Y_3Y_4Y_5Y_6\cdots Y_l) = Y_2Y_1Y_3Y_4Y_5Y_6\cdots Y_l,$$

$$T_{3,m}(Y) = T_{3,m}(Y_1Y_2Y_3Y_4Y_5Y_6\cdots Y_l) = Y_3Y_2Y_1Y_4Y_5Y_6\cdots Y_l,$$

$$T_{4,m}(Y) = T_{4,m}(Y_1Y_2Y_3Y_4Y_5Y_6\cdots Y_l) = Y_4Y_2Y_3Y_1Y_5Y_6\cdots Y_l,$$

where Y_i is a super-symbol, $i = 1,2,3,\ldots,l$. An example of HSN$(3,Q_2)$ is shown in Fig. 1b, where Q_2 is a 2-cube. We can see that an HCN(n,n) without diameter links is equivalent to the special case HSN$(2,Q_n)$. As shown in [26, 33], an HSN can embed corresponding homogeneous product networks such as hypercubes or k-ary n-cubes, with dilation 3. Efficient VLSI layout for HSNs can be found is [31].

3.3 Cyclic-Shift Super-Generators

Cyclic-shift networks (CN) (also called *cyclic networks*) [27, 28] form a special case of super-IP graphs that use super-generators which can perform "cyclic shifts" over the super-symbols. Some subclasses of cyclic-shift networks have fixed node degree and small diameter.

A basic-CN(l,G) (also called ring-CN(l,G)) is defined by nucleus G and super-generators $L_{1,m} = (1 \leftarrow)_m$ and $R_{1,m} = L_{-1,m} = (1 \rightarrow)_m$, where the super-generator $L_{i,m} = (i \leftarrow)_m$ changes the node label $X = X_1X_2\cdots X_l$ into

$$L_{i,m}(X) = X_{i+1}X_{i+2}\cdots X_l\ X_1X_2X_3\cdots X_i$$

and the generator $R_{i,m} = (i \rightarrow)_m$ changes X into

$$R_{i,m}(X) = X_{l-i+1}X_{l-i+2}\cdots X_lX_1X_2X_3\cdots X_{l-i}.$$

3.4 IP Graphs Based on Flip Super-Generators

Another example for the design of super-IP graphs is to use *flip super-generators $F_{i,m}$*, where the super-generator $F_{i,m}$ flips the first i super-symbols, $i = 2,3,...,l$ for an IP graph with l super-symbols in a node label. For example,

$$F_{2,m}(X_1X_2X_3X_4) = X_2X_1X_3X_4,\ F_{3,m}(X_1X_2X_3X_4) = X_3X_2X_1X_4.$$

A super-IP graph with l flip super-generators and nucleus G is called a *super-flip network based on G*. Note that super-flip networks can emulate cyclic-shift networks efficiently since flip super-generators can emulate transposition and cyclic-shift super-generators efficiently, while the latter cannot emulate the former as efficiently. More details can be found in [33].

3.5 Symmetric Super-IP Graph

A *symmetric super-IP graph* is a special type of IP graph whose generator set consists of nucleus generators and super-generators and whose seed label consists of distinct symbols. Since symmetric super-IP graphs form a subclass of Cayley graphs [4], they are vertex-symmetric and regular. In this subsection, we develop a simple and systematic method, based on symmetric super-IP graphs, to obtain symmetric and regular variants of IP graphs.

Recall that an HSN(l,G) can be defined by the transposition super-generators $T_{2,m} = (1,2)_m$, $T_{3,m} = (1,3)_m$, $T_{4,m} = (1,4)_m$,..., $T_{l,m} = (1,l)_m$, the generators for the nucleus G, and the seed $\underbrace{S_1S_1\cdots S_1}_{l}$, where $S_1 = 123\cdots m$ for some nuclei G. If we replace the original seed with a new seed $S_1S_2S_3\cdots S_l$, where $S_i = (i-1)m+1, (i-1)m+2, (i-1)m+3, \ldots, im$, the resultant graph, called a *symmetric HSN(l,G)*, becomes a symmetric super-IP graph which is vertex-symmetric and regular. The seed labels of many HSNs whose nuclei use other type of seed labels can also be transformed into labels that have no repeated symbols, leading to corresponding symmetric HSNs. Since a symmetric HSN(l,G) uses the same generator set and the same nucleus graph as those of an HSN(l,G), we can expect that they share some properties and algorithms. If we assign color i to the super-symbol containing symbol im, then there are $l!$ possible orders of colors for the labels of nodes in a symmetric HSN(l,G). As a result, a symmetric HSN(l,G) has $l!M_N^l$ nodes, $l!$ times more than that of an HSN(l,G), where M_N is the number of nodes in the nucleus G.

A similar strategy can be applied to other super-IP graphs. If we can replace the seed node of a CN(l,G) with the seed $S_1S_2S_3\cdots S_l$, the resultant graph becomes a symmetric super-IP graph, called a symmetric $\mathrm{CN}(l,G)$, which is vertex-symmetric and regular. A symmetric $\mathrm{CN}(l,G)$ has lM_N^l nodes since there are l different orders for the colors of super-symbols. Note that these properties are common to all cyclic-shift networks, including ring-CNs, complete-CNs [27, 28], and their intermediate variants.

Similarly, this strategy can be applied to virtually any IP graph, such as an HCN, HFN, RCC, shuffle-exchange network, de Bruijn graph, HSE, or super-flip network, to obtain its symmetric and regular Cayley-graph counterpart. Note that even though the derived Cayley graphs have size and connectivity that are different from the original networks, they still share some properties and algorithms in common due to the similarity in their generator sets.

4 Routing and Network Diameter

Recall that the routing algorithms on Cayley graphs and, in particular, star graphs, can be viewed as "sorting" the symbols in the label associated with the source node until that of the destination node is obtained. Routing within any IP graph can also be performed in a similar manner.

Theorem 4.1 *Let t be the minimum number of applications of the super-generators of a super-IP graph in order for each super-symbol to appear at the leftmost position at least once. Then the diameter of the super-IP graph is $lD_G + t$, where l is the number of super-symbols in a node label and D_G is the diameter of the nucleus of the super-IP graph.*

Proof: In what follows we present a routing algorithm for the super-IP graph by sorting the label of the source node to the label of the destination node. Choose a particular t-step rearrangement of the l super-symbols such that each super-symbol is brought to the leftmost position at least once. Let d_i be the final position of the super-symbol initially in position i. We first use the nucleus generators of the super-IP graph to sort the leftmost super-symbol to that of the d_1-th super-symbol of the destination node address. We then use the above t steps to bring each of the super-symbols of a node label to the leftmost position at least once. When the super-symbol initially in position i is brought to the leftmost position for the first time, we use the nucleus generators of the super-IP graph to sort the current leftmost super-symbol to that of the d_i-th super-symbol of the destination node address.

Since the diameter of the nucleus graph is D_G, the time required for the nucleus generators to sort a super-symbol is no more than D_G. Therefore, routing in the super-IP graph can be performed in no more than $lD_G + t$ time.

Let A' and B' be the addresses of two nodes that have distance D_G within the same nucleus G. Routing from node $A = \underbrace{A'A'\cdots A'}_{l}$ to node $B = \underbrace{B'B'\cdots B'}_{l}$ requires at least $lD_G +$ t steps since each of the l super-symbols with value A' has to be sorted to the state with value B', and the super-symbols have to be brought to the leftmost position so that they can be sorted. This completes the proof. □

It can be seen that the parameter t in Theorem 4.1 is at least $l-1$ for any super-IP graph and is equal to $l-1$ for all the super-IP graphs introduced in Section 3. This, combined with Theorems 4.1 and 3.2, leads to the following corollary.

Corollary 4.2 *The diameter of an N-node HSN, RHSN, RCC, cyclic-shift network, directed cyclic-shift network, or super-flip network is*

$$(D_G + 1)\log_{M_N} N - 1,$$

where M_N is the number of nodes in the nucleus graph and D_G is the diameter of the nucleus graph.

The proofs and examples for the following theorems can be found in [33].

Theorem 4.3 *Let t_S be the minimum number of applications of the super-generators of a symmetric super-IP graph in order for each super-symbol to appear at the leftmost position at least once and for the super-symbols to be eventually arranged to any possible order. Then the diameter of the symmetric super-IP graph is $lD_G + t_S$, where l is the number of super-symbols in a node label and D_G is the diameter of the nucleus of the super-IP graph.*

Theorem 4.4 *The diameter of an HSN, cyclic-shift network, directed cyclic-shift network, super-flip network, RCC, RHSN, or any of their symmetric super-IP graph variants is asymptotically optimal within a factor of $1 + o(1)$ from its lower bound (given its node degree) if the diameter of the nucleus graph is asymptotically optimal within a factor of $1 + o(1)$ from its lower bound and $d_S = d_N^{1+o(1)}$, where d_S is the number of super-generators, d_N is the number of nucleus-generators, and D_G and d_N are not constant. The diameter of any of these networks is asymptotically optimal within a constant from its lower bound if the diameter of the nucleus graph is asymptotically optimal within a constant from its lower bound and $\log_2 d_S = O(\log d_N)$.*

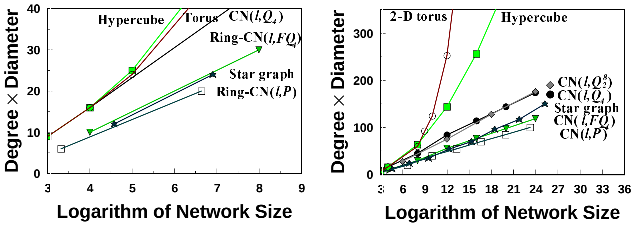
To obtain (symmetric) super-IP graphs with optimal diameters, we can use networks such as a generalized hypercube [7] of proper size and dimension as the nucleus.

5 Comparison of Several Networks

In this section, we look into several implementation and performance issues, including the assignment of processors to chips (or modules), pin limitations, bandwidth of on-chip and off-chip links, and the number of off-module transmissions required for routing.

5.1 Comparison of DD-Cost

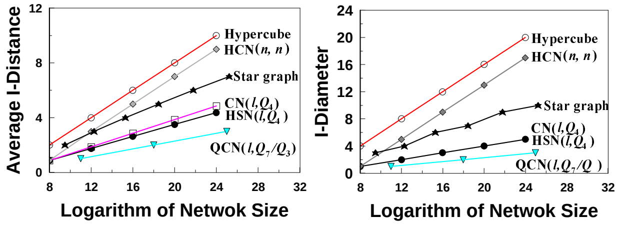
Although diameter and average distance may be less important for networks using wormhole routing under light traffic, they are crucial for network performance under heavy load. The maximum possible throughput of a network is inversely proportional to these parameters for any switching technique. Figure 2 shows a rough comparison of some of the interconnection networks discussed so far and certain other popular networks on the basis of the product of node degree and network diameter (which is regarded as



(a)

(b)

Figure 2. Comparison of DD-cost, the product of node degree and network diameter, for several interconnection networks, where P is the Petersen graph, and FQ_4 is a 4-dimensional folded hypercube.



(a)

(b)

Figure 3. Comparison of the (a) average inter-cluster distance and (b) inter-cluster diameter for several networks, assuming that at most 24 processors are available per cluster (module). Quotient-CN$(l, Q_7/Q_3)$, abbreviated QCN$(l, Q_7/Q_3)$ here, is obtained by merging each 3-cube in CN(l, Q_7) into a node. HCN(n, n) in the above figures are HSN$(2, Q_n)$ = HCN(n, n) without diameter links.

a suitable composite figure of merit [7] and is called *DD-cost* in this paper). When the sum of the capacities of all the links of a node is fixed (i.e., unit node capacity) and packet-switching is used (or wormhole/cut-through routing is used but messages are very short), the latency of a network with light traffic is approximately proportional to its DD-cost. From Fig. 2, we can see that cyclic-shift networks have DD-cost that is comparable to that of the star graph, and outperform other popular topologies significantly under this criterion, especially when the network size is large. In [2, 11], it has been shown that low-dimensional k-ary n-cubes perform better than high-dimensional ones under the constraint of constant bisection bandwidth. In [1], Abraham and Padmanabhan examined network performance under pin-out constraints and showed that higher-dimensional networks performed better. Generally speaking, low-dimensional k-ary n-cubes outperform super-IP graphs under the constant bisection-bandwidth constraint; while super-IP graphs outperform k-ary n-cubes and hypercubes under constant pin-out constraint. Detailed comparisons based on such considerations are outside the scope of this paper.

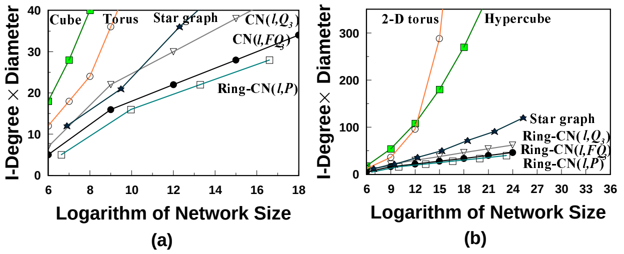
5.2 I-Diameter and Average I-Distance

In present computing environments, processors are expensive and memory is relatively cheap. Therefore, an optimization question is: “how large should the memory be to utilize the processor(s) efficiently?” The utilization of processors in parallel computers is not as efficient as that in a single-processor computer for general-purpose applications, so that the latter achieves better performance per dollar. In future computing environments, however, the roles might be reversed, so that memory is expensive and processors are relatively cheap. Therefore, the question might become: “how many processors are appropriate to utilize the memory efficiently?” As pointed out by Dally [12] and researchers working on processor in memory (PIM) or computing in RAM [18, 23], multiple processors per chip, integrated with memory banks, can increase memory-processor bandwidth considerably and improve the utilization of memory significantly. Moreover, with the rapid advances in VLSI technologies, the number of transistors and the number of processors that can be put onto a chip are expected to continue their exponential growths. Therefore, single-chip multiprocessors are expected to achieve better performance per dollar even for general-purpose applications and may become a mainstream in the future computing market. Another trend in the synthesis of multicomputers is to use off-the-shelf PC or workstation boards (or processor chips) as building modules.

In either case, a parallel computer is built from several chips on a board and multiple such boards in a card-cage. Modules at each level of the packaging hierarchy have their respective characteristics in terms of the number of pins, maximum capacity, bisection bandwidth, maximum wire length, and channel bandwidth [6]. In what follows, we consider the case where several nodes (processors, routers, and associated memory banks) of a network are implemented on a single chip, or more generally, a single module (e.g., chip, board, wafer, or multi-chip module (MCM)). Since transmissions over off-chip (or off-module) links are more expensive than transmissions over on-chip (or on-module) links, it is generally preferred to reduce the number of off-chip (or off-module) transmissions for a routing task. Figure 3a compares the *average inter-cluster distance (average I-distance)* (also called *average inter-module distance*), the average number of inter-cluster or off-module (e.g., off-chip or off-board) transmissions required for routing between two nodes, in several interconnection networks, assuming that at most 24 nodes can be placed within a module, if the parallel system is to execute a random routing problem with uniformly distributed sources and destinations. Figure 3b compares the *inter-cluster diameter (I-diameter)* (also called *inter-module diameter*), the maximum number of inter-cluster (off-module) transmissions for routing between two nodes, in several interconnection networks. It can be shown that the maximum throughput of a network is inversely proportional to its average inter-cluster distance when the off-module links are uniformly utilized and the off-module bandwidth is the communication bottleneck.

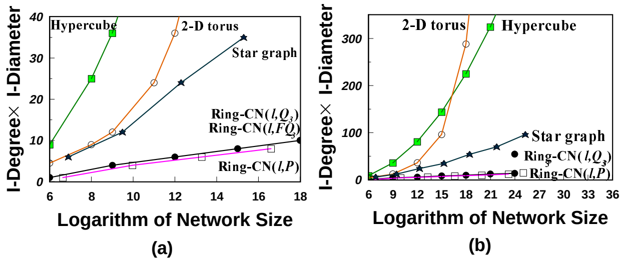
5.3 Comparison of Inter-cluster Degree

We define the *inter-cluster degree (I-degree)* (or *inter-module degree*) as the maximum of the average-per-node inter-cluster (off-module) links over all clusters (modules). Since the number of available off-module pins per node is one of the major constraints limiting the performance and



(a) (b)

Figure 4. Comparison of ID-cost, the product of inter-cluster degree and diameter, for several interconnection networks, assuming that no more than 10 nodes can be placed in a cluster (module).



(a) (b)

Figure 5. Comparison of II-cost, the product of inter-cluster degree and inter-cluster diameter, for several interconnection networks, assuming that no more than 10 nodes can be placed in a cluster (module).

the number of processors that can be put on the module, it is desirable to minimize the *inter-cluster degree* when assigning network nodes to basic building modules. Also, to eliminate the "number of parts" problem, it is preferred that the building chips be identical or of only a few types. To satisfy the above criteria, we can place each of the nuclei of a super-IP graph within the same module when partitioning the network. Then the maximum number of off-module links per node in an l-level ring-cyclic network is equal to 1 when $l = 2$ and 2 when $l \geq 3$; the corresponding numbers for an l-level HSN, complete-CN [27, 28], or super-flip network are 1,2,3,4, respectively, when $l = 2,3,4,5$. As a comparison, the maximum numbers of off-module links per node in an n-dimensional hypercube and star graph are $n-3$ (or $n-4$) and $n-2$ (or $n-3$), when a 3(or 4)-cube or a 3(or 4)-star is placed within the same module, where $n = \log_2 N$ for an N-node hypercube and $n = O(\log N / \log\log N)$ for an N-node star graph. For example, a node in a 17-cube has 14 (or 13) off-module links and a node in a 8-star has 6 (or 5) off-module links.

If we assume *unit node off-module capacity*, where the average off-module bandwidth per node (i.e., the sum of the bandwidth of all off-module links per node) is the same for parallel architectures based on different networks, then an off-module link of a super-IP graph has bandwidth considerably larger than that of a hypercube or star graph. The maximum number of off-module links per node in a de Bruijn graph is equal to 4 when assigning nodes with the same most significant bits into the same module, so the bandwidth of an off-module link of a ring-cyclic network or an HSN(l,Q_4) (or complete-CN(l,Q_4)) of practical size is also better than that of a de Bruijn graph using such a partitioning. Note that when wormhole or cut-through routing is used and messages are long, the delay of a network with light traffic is approximately proportional to its inter-cluster degree.

5.4 Comparison of ID-Cost and II-Cost

We define the *ID-cost* of a network as the product of its inter-cluster degree and its diameter. The inter-cluster degree is usually (approximately) equal to the number of off-module links per node, which is the case for the networks considered in Figs. 2,3, and 4, except for 2-D tori. When the sum of the capacities of all the off-module links of an M-node module is cM, where c is a constant (i.e., the sum of the capacities of all the off-module links of a node is fixed for networks considered in Fig. 4, except for 2-D tori), the delay of a packet-switched network with light traffic is proportional to its ID-cost. The delay of a network using wormhole or cut-through routing is also approximately proportional to its ID-cost when the traffic is light and the messages are short. From Fig. 4, it can be seen that cyclic-shift networks have ID-cost that is considerably smaller than those of other popular topologies, for small- to large-scale networks.

In the preceding arguments, we have assumed that the speeds of all links, including on-module and off-module links, are the same and the traffic is approximately balanced over all network links. However, on-chip links are significantly shorter than off-chip links and do not need extra delay to drive off-chip pins, they can be driven at a considerably higher clock rate. Moreover, since the cost for an on-chip connection is much smaller than that of an off-chip connection, the channel width of an on-chip link can be increased, if required, without significantly increasing the hardware cost. When transmissions over on-module links are considerably faster than over off-module links, the delay of a packet-switched network with light traffic is approximately proportional to its *II-cost*, defined as the product of its inter-cluster degree and inter-cluster diameter. Moreover, when the traffic is heavy and the utilization of off-module links is higher than that of on-module links, the delay of a packet-switched network is also approximately proportional to its II-cost even when all links in the network have the same speed, since the average waiting time required for a packet to be transmitted over an off-module link is considerably larger than that required for an on-module link. From Fig. 5, we can see that cyclic-shift networks have II-cost that is considerably smaller than those of other popular topologies, for small- to large-scale networks, even when module size is limited to 8 or 10 nodes. When the module size is larger than 10 nodes, the superiority of super-IP graphs over other network topologies is even more pronounced.

6 Conclusion

In this paper, we have presented an extension of Cayley graphs, called the IP graph model, for the development of communication-efficient interconnection networks. We presented several interconnection networks based on super-IP and symmetric super-IP graphs that have certain desirable properties. The diameters and inter-cluster diameters of suitably constructed (symmetric) super-IP graphs are asymptotically optimal within a small constant factor from their respective lower bounds. IP graphs provide flexibility in the design of parallel architectures in view of the possibility of selecting several parameters, nuclei, super-generators, seed labels, and/or the nodes to be merged, an appropriate combination of which can mitigate performance bottlenecks and balance system resources. In particular, a dense nucleus graph reduces the diameter and average distance, a strong set of super-generators enhances the embedding capability, a seed label consisting of distinct symbols generates a symmetric and regular network, a quotient variant [28, 33] minimizes the required off-module data transmissions, and their combined effect determines the algorithmic properties of the resulting network.

References

[1] Abraham, S. and K. Padmanabhan, "Performance of multicomputer networks under pin-out constraints," *J. Parallel Distrib. Comput.,* Vol. 12, no. 3, Jul. 1991, pp. 237-248.

[2] Agarwal, A., "Limits on interconnection network performance," *IEEE Trans. Parallel Distrib. Sys.,* Vol. 2, no. 4, Oct. 1991, pp. 398-412.

[3] Akers, S.B., D. Harel, and B. Krishnamurthy, "The star graph: an attractive alternative to the n-cube," *Proc. Int'l Conf. Parallel Processing,* 1987, pp. 393-400.

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

[5] Akl, S.G., *Parallel Computation: Models and Methods,* Prentice Hall, Englewood Cliffs, NJ, 1997.

[6] 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.

[7] 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.

[8] Biggs, N., *Algebraic Graph Theory,* 2nd edition, Cambridge, Cambridge University Press, 1993.

[9] Corbett, P.F., "Rotator graphs: an efficient topology for point-to-point multiprocessor networks," *IEEE Trans. Parallel Distrib. Sys.,* vol. 3, no. 5, pp. 622-626, Sep. 1992.

[10] Cypher, R. and J.L.C. Sanz, "Hierarchical shuffle-exchange and de Bruijn networks," *Proc. IEEE Symp. Parallel and Distributed Processing,* 1992, pp. 491-496.

[11] Dally, W.J., "Performance analysis of k-ary n-cube interconnection networks," *IEEE Trans. Comput.,* Vol. 39, no. 6, Jun. 1990, pp. 775-785.

[12] Dally, W.J. and S. Lacy, "VLSI architecture: past, present, and future," *Proc. Advanced Research in VLSI Conf.,* 1999, to appear.

[13] Duh, D., G. Chen, and J. Fang, "Algorithms and properties of a new two-level network with folded hypercubes as basic modules," *IEEE Trans. Parallel Distrib. Sys.,* vol. 6, no. 7, Jul. 1995, pp. 714-723.

[14] Fragopoulou, P. and S.G. Akl, "Edge-disjoint spanning trees on the star network with applications to fault tolerance," *IEEE Trans. Computers,* vol. 45, no. 2, Feb. 1996, pp. 174-185.

[15] Ghose, K. and R. Desai, "Hierarchical cubic networks," *IEEE Trans. Parallel Distrib. Sys.,* vol. 6, no. 4, Apr. 1995, pp. 427-435.

[16] Hamdi, M., "A class of recursive interconnection networks: architectural characteristics and hardware cost," *IEEE Trans. Circuits and Sys.–I: Fundamental Theory and Applications,* vol. 41, no. 12, Dec. 1994, pp. 805-816.

[17] Huang, J.-P., S. Lakshmivarahan, and S.K. Dhall, "Analysis of interconnection networks based on simple Cayley coset graphs," *Proc. IEEE Symp. Parallel and Distributed Processing,* 1993, pp. 150-157.

[18] Kogge, P.M., "EXECUBE – a new architecture for scalable MPPs," *Proc. Int'l Conf. Parallel Processing,* vol. I, 1994, pp. 77-84.

[19] Lakshmivarahan, S., J.-S. Jwo, and S.K. Dhall, "Symmetry in interconnection networks based on Cayley graphs of permutation groups: a survey," *Parallel Computing,* Vol. 19, no. 4, Apr. 1993, pp. 361-407.

[20] Latifi, S., M.M. 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.

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

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

[23] Sterling, T., P. Messina, and P. Smith, *Enabling Technologies for Petaflops Computing, MIT Press.,* 1995.

[24] Yeh, C.-H. and B. Parhami, "Parallel algorithms on three-level hierarchical cubic networks," *Proc. High Performance Computing Symp.,* Mar. 1996, pp. 226-231.

[25] Yeh, C.-H. and B. Parhami, "Hierarchical swapped networks: efficient low-degree alternatives to hypercubes and generalized hypercubes," *Proc. Int'l Symp. Parallel Architectures, Algorithms, and Networks,* 1996, pp. 90-96.

[26] Yeh, C.-H. and B. Parhami, "Recursive hierarchical swapped networks: versatile interconnection architectures for highly parallel systems," *Proc. IEEE Symp. Parallel and Distributed Processing,* Oct. 1996, pp. 453-460.

[27] Yeh, C.-H. and B. Parhami, "Cyclic networks – a family of versatile fixed-degree interconnection architectures," *Proc. Int'l Parallel Processing Symp.,* Apr. 1997, 739-743.

[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 E.A. Varvarigos, "Parallel algorithms on the rotation-exchange network – a trivalent variant of the star graph," *Proc. Symp. Frontiers of Massively Parallel Computation,* Feb. 1999, pp. 302-309.

[31] 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.

[32] Yeh, C.-H. and B. Parhami, "Routing and embeddings in cyclic Petersen networks: an efficient extension of the Petersen graph," *Proc. Int'l Conf. Parallel Processing,* Sep. 1999, to appear.

[33] 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.

[34] 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.