VLSI Design
1995, Vol. 2, No. 4, pp. 365–374
Reprints available directly from the publisher
Photocopying permitted by license only

Trade-Off Considerations in Designing Efficient VLSI Feasible Interconnection Networks

S. Q. ZHENG
Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803

B. CONG
Department of Computer Science, South Dakota State University, Brookings, SD 57007

S. BETTAYEB
Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803

It is well known that the hypercube has a rich set of good properties, and consequently it has been recognized an ideal structure for parallel computation. Nevertheless, according to the current VLSI technology, the implementation feasibility of the hypercube remains questionable when the size of the hypercube becomes large. Recent research efforts have been concentrated on finding good alternatives to the hypercube. The star graph was shown having many desirable properties of the hypercube, and in several aspects, the star graph is better than the hypercube. However, we observe that the star graph as a network has several disadvantages, compared with the hypercube. In this paper, we propose a class of new networks, the star-hypercube hybrid networks (or the *SH* networks). The *SH* network is a simple combination of both the star graph and the hypercube. This class of networks contains the star graph and the hypercube as subclasses. We show that the *SH* network is an efficient and versatile network for parallel computation, since it shares properties of both the hypercube and the star graph, and remedies several major disadvantages of the hypercube and the star graph. This class of networks provide more flexibility in choosing the size, degree, number of vertices, degree of fault tolerance, etc. in designing massively parallel computing structures feasible for VLSI implementations.

Key Words: *Parallel computing, interconnection network, VLSI, network simulation*

1 INTRODUCTION

It is well known that the hypercube has a rich set of good properties, and consequently it has been recognized an ideal structure for parallel computation [9]. Many efficient parallel algorithms for the hypercube computers have been developed. Nevertheless, according to the current VLSI technology, its implementation feasibility remains questionable when the size of the hypercube becomes large. This is because that the degree and the number of edges in the hypercube grow rapidly as the number of vertices increases. Recent research efforts have been concentrated on finding good alternatives to the hypercube (e.g. [1, 2, 8]). The star graph was shown having many desirable properties of the hypercube [1, 2, 3]. For example, as the hypercube, the star graph is a recursively defined regular graph, it is vertex and edge symmetric, and it admits high fault-tolerance. The major advantage of the star graph over the hypercube is that its degree and diameter are sub-logarithmic as functions of the number of vertices. More specifically, the n-star graph S_n contains $n!$ vertices, and its degree and diameter are $n - 1$ and $\lfloor 3(n-1)/2 \rfloor$, respectively. By contrast, the n-dimensional hypercube has degree and diameter which are logarithmic as functions of the number of vertices. The investigation on combinatorial, fault-tolerant, communication and computational aspects of the star network is still in its early stage. However, the star graph as a network has several disadvantages, compared with the hypercube. For example, its variable incremental factor prohibits its implementation when n becomes large, and its sparser connectivity and less regularity result in less efficient parallel algorithms, and less efficiency in simulating many other useful network structures.

In this paper, we propose a class of new networks, the star-hypercube hybrid networks (or the *SH* networks). The *SH* network is a simple combination of

both the star graph and the hypercube. This class of networks contains the star graph and the hypercube as subclasses. We show that the *SH* network is an efficient and versatile network for parallel computation, since it shares properties of both the hypercube and the star graph, and remedies several major disadvantages of both. This class of networks provide us more flexibility in choosing the size, degree, number of vertices, degree of fault tolerance, etc. in designing massively parallel computer systems.

2 THE HYPERCUBE AND THE STAR GRAPH

The n-dimensional hypercube, denoted by Q_n, is a graph of 2^n vertices, each is labeled by an n-bit binary number. Two vertices u and v of Q_n are connected by an edge if and only their binary labels differ in exactly one bit position. An alternative definition of hypercubes is as follows. Q_1 is a graph of two vertices, labeled by 0 and 1 respectively, connected by an edge. For $n > 1$, Q_n consists of two copies of Q_{n-1}, Q^0_{n-1} whose vertex labels are prefixed by a 0, and Q^1_{n-1}, whose vertex labels are prefixed by a 1. Two vertices $u = 0u_{n-2}u_{n-3} \cdots u_0 \in Q^0_{n-1}$ and $v = 1v_{n-2}v_{n-3} \cdots v_0 \in Q^0_{n-1}$ are connected if and only if $u_{n-2}u_{n-3} \cdots u_0 = v_{n-2}v_{n-3} \cdots v_0$. Hypercubes Q_1, Q_2 and Q_3 are shown in Figure 1(a), (b) and (c), respectively.

A rich set of symmetry properties make the hypercube an ideal structure for parallel computation [9, 14]. The n-dimensional hypercube is a regular graph of degree n. Used as an interconnection network, the hypercube has high bandwidth and high fault-tolerance. It is vertex and edge symmetric. A graph is vertex symmetric if for its any two vertices u and v there is an automorphism of the graph that maps u to v. Similarly, a graph is edge symmetric if for its any two edges e_1 and e_2 there is an automorphism of the graph that maps e_1 to e_2. Such symmetries are very important for a graph to be used as an interconnection network. For example, vertex (edge) symmetry allows for all processors (data links) to be treated as identical. By such symmetries, simple and efficient data routing schemes can be derived and congestions in data communication can be minimized. The diameter of Q_n is n, which is logarithmic with respect to the number of vertices. The recursive definition of Q_n indicates that it is perfectly suitable for divide-and-conquer parallel algorithms. Opposite to its partitionability, the hypercube can be easily expanded by a factor of 2 in size without changing its substructures. Furthermore, the N-processor hypercube can simulate many $O(N)$-processor networks (e.g. ring, array, binary tree, and mesh-of-trees) with only a small constant factor slow-down. The major drawback of the hypercube is that the number of its edges increases rapidly as its dimension grows, and it becomes diffiult to implement the hypercube network when its dimension becomes large.

Recently, the star graph was proposed as an alternative to the hypercube. The star graphs belong to a class of networks formed by Cayley graphs, a family of graphs with group theoretic properties. The n-star, denoted by S_n, is a graph of $n!$ vertices, each is labeled by a unique permutation of $\{i_1, i_2, \ldots, i_n\}$, a set of n distinct symbols. In particular, we may assume that $i_k = k$. Two vertices $u = (u_1, u_2, \cdots u_n)$ and $v = (v_1, v_2, \cdots v_n)$ are connected by an edge if and only if for some $j > 1$ such that $u_j = v_1$, $v_j = u_1$, and $u_k = v_k$ for $k \neq 1, j$. As the hypercube, the star graph can also be defined recursively. We say a permutation $p = (i_1, i_2, \cdots, i_n)$ is ended with k if $i_n = k$. Let S^k_{n-1} be the $(n-1)$-star defined by the permutations of $\{1, 2, \ldots, n\} - \{k\}$, and let $S^k_{n-1}k$ denote the $(n-1)$-star obtained from S^k_{n-1} in such a way that all vertices in $S^k_{n-1}k$ are labeled by permutations of $\{1, 2, \ldots, n\}$ ended with k and a vertex u in $S^k_{n-1}k$ is labeled $(i_1, i_2, \ldots, i_{n-1}, k)$ if it is labeled $(i_1, i_2, \cdots, i_{n-1})$ in S^k_{n-1}. S_1 is a graph of a single vertex labeled by a symbol. For $n > 1$, S_n consists of n copies of $(n-1)$-stars, $S^1_{n-1}1$, $S^2_{n-1}2$, $\cdots$ $S^m_{n-1}n$.

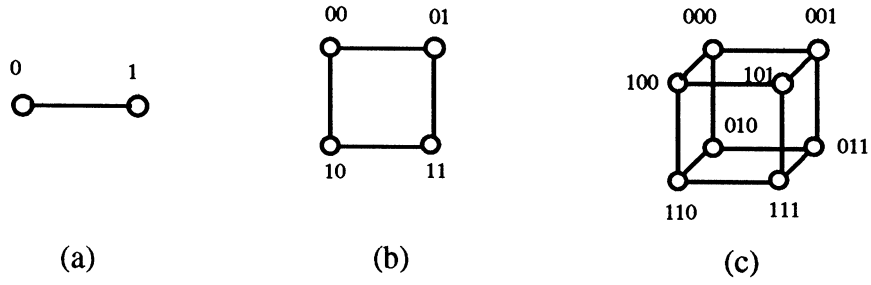

FIGURE 1 Hypercubes (a) Q_1, (b) Q_2 and (c) Q_3.

Two vertices $u = (u_1, u_2, \cdots u_{n-1}, i) \in S^i_{n-1}i$ and $v = (v_1, v_2, \cdots v_{n-1}, j) \in S^j_{n-1}j$, $i \neq j$, are connected if and only if $u_1 = j$ and $v_1 = i$, and $u_k = v_k$ for $k \neq i, j$. Star graphs S_2, S_3 and S_4 are shown in Figure 2 (a), (b) and (c), respectively.

It has been shown that the star graph possesses many symmetry properties of the hypercube [1, 2, 3]. For example, as the hypercube, the star graph is vertex and edge symmetric, and admits high fault-tolerance. The major advantage, among several others, of the star graph is that its degree and diameter are sub-logarithmic, with respect to the number of its vertices. More specifically, the n-star graph S_n contains $n!$ vertices, but its degree and diameter are $n - 1$ and $\lfloor 3(n - 1)/2 \rfloor$, respectively. That is, the star graph is a graph sparser than the hypercube, yet it has a smaller diameter. The investigation on the communication, algorithmic and fault-tolerance aspect of the star graph as an interconnection network is still in its early stage. However, we observed several disadvantages of the star graph.

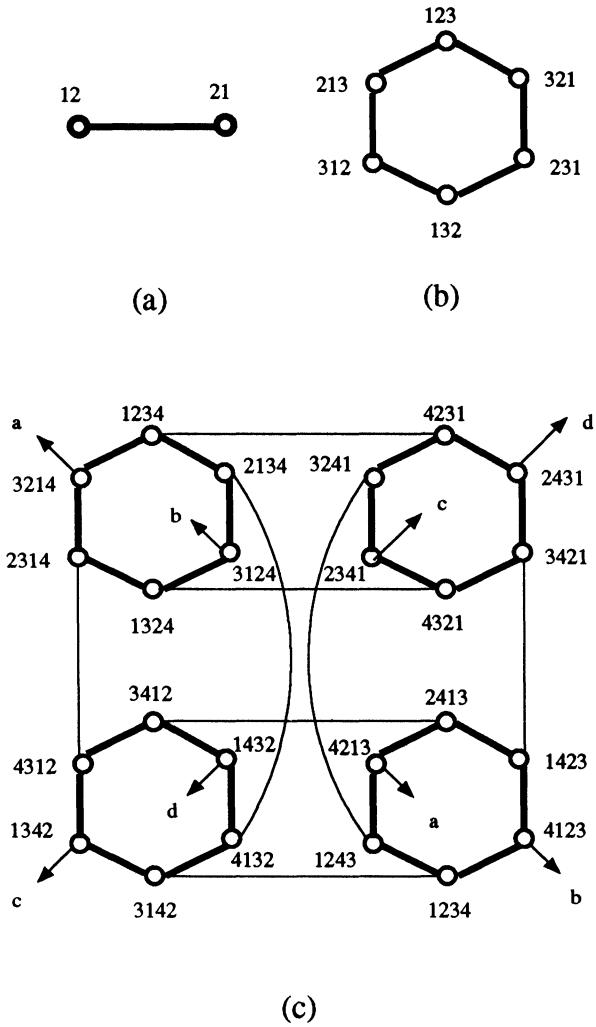

FIGURE 2 Star Graphs (a) S_2, (b) S_3, and (c) S_4.

We informally define the *incremental factor* of a recursive network G as the minimum number of copies of smaller networks with the same connectivity property of G that are used in constructing G. Clearly, the incremental factor of the hypercube is 2, a constant, whereas the incremental factor of the n-star graph S_n is n. Because of its linear incremental factor, we claim that the star graph is less regular in structure than the hypercube. Such irregularity may lead to serious problem in the realization of the star networks. The hardware implementation of multiprocessors involves issues in VLSI layout, chip packaging, printed-circuit board layout and connection, inter-board wire connection, etc. Suppose that the current state of the electronic technologies allows for the implementation of a multiprocessor system with the 10-star as the underlying network, which consists of 362,780 processors. To build any larger star-based multiprocessor system, the technology must advance to a stage that about at least four millions of processors can be integrated into the system. When n is large, the construction of star-based systems may become far from reality.

Sur and Srimani proposed a new network structure, called the superstar graph [15]. This is a class of networks based on the star graph. They show that given any $N > 0$, a superstar graph of N vertices can be constructed using copies of star graphs of different sizes. As the star graph, the superstar graph is optimally fault tolerant, and its diameter is sublogarithmic in the number of vertices. While the superstar graph is a good network structure for distributed computing systems, it is not a good alternative for massively parallel computer systems, because it is difficult to design well structured efficient parallel algorithms for it.

Another major disadvantage of the star multiprocessor system is that the parallel algorithms on the star structure may not be as efficient as those on the hypercube structure. We observed that all known algorithms on S_n are no more efficient but more complicated than their counterparts on Q_n (e.g. [4, 12]). While the investigation on the computational aspects of the star graph has just begun, it is reasonable to believe that in general it is more difficult to use the star network for parallel computing, and the algorithms on the star network are less efficient than their counterparts on the hypercube. This is because that, compared with the hypercube, the decompositions of the star network into subnetworks are less regular, and the bandwidth between the subnetworks of S_n is smaller.

All known results indicate that the ability of the star network in simulating other useful networks is not as good as the hypercube (e.g. [5–7]). This is another disadvantage of the star graph. Again, one can observe that this is because all useful networks (including the hypercube) have constant incremental factors, but the star graph has a linear incremental factor, and furthermore, the hypercube has more edges than the star graph.

3 STRUCTURE OF THE SH NETWORK

As discussed in the previous section, several advantageous features of the hypercube are the disadvantageous features of the star graph, and vice versa. It is desirable to design a network which possesses good properties of both and remedy the disadvantages of both. Finding trade-offs between these two networks is the simple idea behind our design of the *star-hypercube hybrid network* (or the *SH network*).

The $SH(x, y)$ network consists of $x!2^y$ vertices. Each vertex u is labeled by a pair $(p(u), b(u))$, where $p(u) = (p_1(u), p_2(u), \ldots, p_x(u))$ and $b(u) = b_y(u)b_{y-2}(u) \cdots b_1(u)$ are called the *permutation label* and the *binary label* of u, respectively. The label $p(u)$ is a permutation of $\{1, 2, \ldots, x\}$, and the label $b(u)$ is a y-bit binary number. Two vertices u and v in $SH(x, y)$ are connected by an edge if and only if exactly one of the following two conditions holds:

(i) $b(u) = b(v)$ and $p(v)$ can be obtained from $p(u)$ by interchanging $p_1(u)$ with some $p_i(u)$, $i = 2, 3, \ldots, x$, and

(ii) $p(u) = p(v)$ and $b(u)$ and $b(v)$ differ in exactly one bit position.

We call an edge that connects two vertices satisfying condition (i) an *s-type edge,* and an edge that connects two vertices satisfying condition (ii) an *h-type edge. SH*(3, 2) *is shown in* Figure 3 *and* Figure 4, *in* which each vertex is labeled by a pair (p, b), where p and b are the permutation label and the binary label of the vertex, respectively.

Conceptually, $SH(x, y)$ can be viewed as a two-level network. Its edges can be partitioned into two sets. Edges in the first set are used to connect vertices into *clusters,* each is a subnetwork of the same structure. These edges are used for *intracluster* data communications. The second set contains the edges connecting vertices from different clusters, and used for *intercluster* data communication. $SH(x, y)$ can be considered as obtained by connecting $x!$ copies of y-dimensional hypercubes by the x-star graphs in such a way that the set of all vertices with the same binary label in these $x!$ y-dimensional hypercubes are connected by an x-star graph. In this sense, the *SH* network can be considered as a star-connected-cube network. Then, each cluster is a y-dimensional hypercube. Also, $SH(x, y)$ can be considered as obtained by connecting 2^y copies of x-star graphs by y-dimensional hypercubes in such a way that the set of all vertices with the same permutation label in these 2^y x-star graphs are connected by a y-dimensional hypercube. In this sense, the *SH* network can be considered as cube-connected-star network. Then, each cluster is an x-star network. If $x!$ is much greater than 2^y, then $SH(x, y)$ is close to a star graph, and if 2^y is much greater than $x!$, the hypercube structure dominates. In Figure 3, $SH(3, 2)$ is shown as a star-connected-cube. In this figure, a cycle of four vertices connected by thick edges is a Q_2. In Figure 4, $SH(3, 2)$ is shown as a cube-connected-star.

The *SH* network can be defined recursively, as the hypercube and the star graph. Clearly, $SH(0, y)$ is the y-dimensional hypercube, and $SH(x, 0)$ is the x-star graph. Hence, the class of *SH* networks contains the hypercube and the star graph as subclasses. $SH(x, y)$ can be constructed using two copies of $SH(x, y - 1)$, $SH^0(x, y - 1)$ whose binary labels are prefixed by a 0, and $SH^1(x, y - 1)$, whose binary labels are prefixed by 1. The new binary bit for each node u is denoted by $b_y(u)$. Two nodes, $u \in SH^0(x, y - 1)$ and $v \in SH^0(x, y - 1)$ are connected by an edge if and only if $p(u) = p(v)$ and $b_i(u) = b_i(v)$ for $1 < i \leq y - 1$. Similarly, $SH(x, y)$ can be constructed by combining x copies of $SH(x - 1, y)$. Therefore, $SH(x, y)$ has an incremental factor either 2 or x. The *SH* networks combine the features of the hypercube and the star graph, and they preserve many properties of both, as illustrated in the next section. This class of networks provide us more flexibility in choosing the size, degree, number of vertices, degree of fault tolerance, etc. in designing massively parallel computer systems.

4 SOME PROPERTIES OF THE SH NETWORK

The properties of the *SH* network can be derived from the properties of the hypercube and the star graph. In this section, we discuss several fundamental ones. It is straightforward to see that

Property 1: The connectivity of $SH(x, y)$ can be derived from the succinct vertex labels.

This property leads to efficient data routing algorithms, since routing tables can be avoided. Routa-

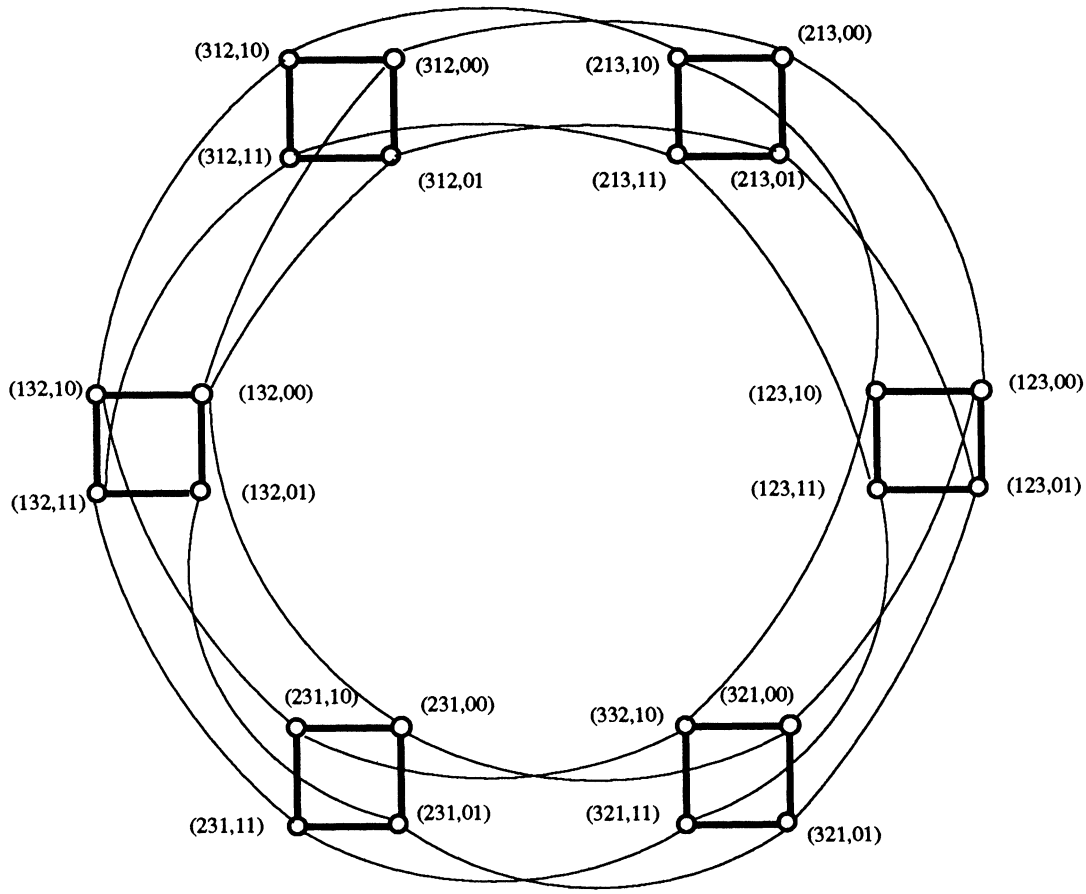

FIGURE 3 *SH*(3, 2), viewed as a star-connected-cube network.

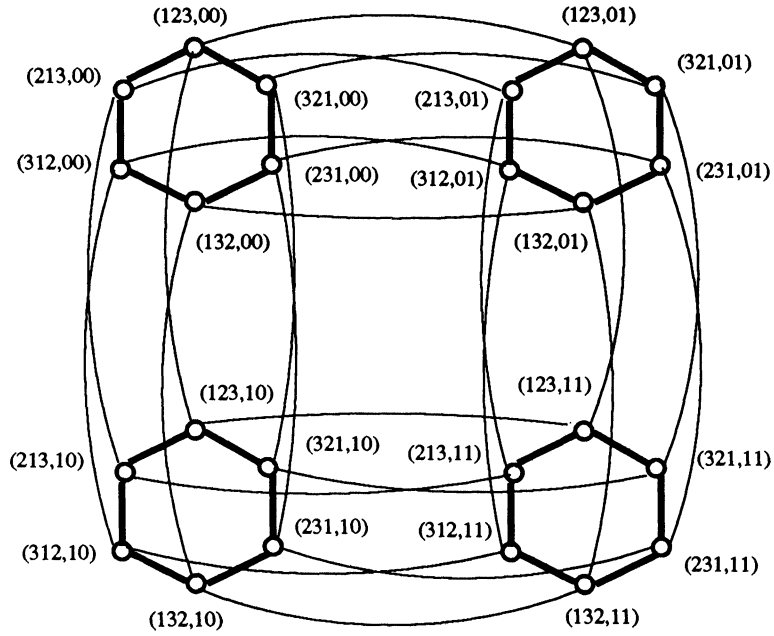

FIGURE 4 *SH*(3, 2), viewed as a cube-connected-star network.

bility with compact routing information is very important for massively parallel computers.

Property 2: $SH(x, y)$ is a regular graph of degree $x + y - 1$. The number of vertices and the number of edges in $SH(x, y)$ are $x!2^y$ and $x!2^{y-1}(x + y - 1)$, respectively. The diameter of $SH(x, y)$ is $\lfloor 3(x - 1)/2 \rfloor + y$. The average distance between vertices in $SH(x, y)$ is $O(x + y)$.

The average distance of $SH(x, y)$ can be verified by the results of [1, 2]. Let us verify the diameter of $SH(x, y)$. By the definition of $SH(x, y)$, for any two vertices u and v in $SH(x, y)$ such that the distance between $p(u)$ and $p(v)$ in S_x is $\lfloor 3(x - 1)/2 \rfloor$ and the distance between $b(u)$ and $b(v)$ is y in Q_y, the distance between u and v is no less than $\lfloor 3(x - 1)/2 \rfloor + y$. On the other hand, using the shortest path algorithm for S_x, a path of length $\lfloor 3(x - 1)/2 \rfloor$ from u to v can be found by finding a path from u to w, where $p(w) = P(v)$ and $b(w) = b(u)$ by the rules given in [1]. Then, a path of length y from w to v can be found. Therefore, the diameter of $SH(x, y)$ is $\lfloor 3(x - 1)/2 \rfloor + y$. Note that this proof also gives an algorithm for one-to-one data routing in $SH(x, y)$ along a shortest path.

For $SH(x, y)$, if we choose x and y properly, say letting $x = y = n$, then the degree and diameter of are sublogarithmic as functions of the number of vertices in $SH(n, n)$. In general, if $y < O(x \log x)$, the diameter of $SH(x, y)$, the ratio between the number of edges and the number of vertices in $SH(x, y)$, and the average distance between vertices in $SH(x, y)$ are sublogarithmic as functions of the number of vertices and the number of edges in $SH(x, y)$, as the star graph. In contrast, these parameters for Q_n are logarithmic with respect to the number of vertices and the number of edges.

Property 3: The SH network is vertex symmetric, s-type edge symmetric and h-type edge symmetric.

Here, by s-type (h-type) edge symmetry we mean that all s-type (h-type) edges can be viewed identical. Consider an arbitrary pair of vertices u and v of $SH(x, y)$. By the vertex symmetry of the star graph [1, 2], there is an automorphism of $SH(x, y)$ that maps u to w such that $p(w) = p(v)$ and $b(w) = b(u)$. Then, by the vertex symmetry of the hypercube graph, there is an automorphism of $SH(x, y)$ that maps w to v. Therefore, $SH(x, y)$ is vertex symmetric. Consider any given pair of two s-type edges, e_1 connecting $u_{1,1}$ and $u_{1,2}$ and e_2 connecting $v_{1,1}$ and $v_{1,2}$, of $S(x, y)$. By the definition of $SH(x, y)$ and the vertex symmetry of the hypercube, we know that there is an automorphism of $SH(x, y)$ that maps e_1 to e' that connects $u'_{1,1}$ and $u'_{1,2}$ such that $b(u'_{1,1}) = b(v_{1,1})$ and $b(u'_{1,2}) = b(v_{1,2})$. Then, by the edge symmetry of the star graph, we know that there is an automorphism of $SH(x, y)$ that maps e' to e_2. Similarly, we can show that there is an automorphism of $SH(x, y)$ that maps any h-type edge to any other h-type edge in $SH(x, y)$.

Property 4: The SH network is Hamiltonian and bipartite.

We know that both the hypercube and the star graph are Hamiltonian [7, 9], i.e. they contain Hamiltonian cycles. Also, both of them are bipartite. To show that $SH(x, y)$ contains a Hamiltonian cycle, we view $SH(x, y)$ as a star-connected-cube. Let $(p_0, p_2, \ldots, p_{x!-1})$ be a Hamiltonian cycle C_S in S_x, where each p_i is a permutation of $\{1, 2, \ldots, x\}$, and b_1 and b_2 be the binary labels of two adjacent vertices in a Hamiltonian cycle C_H in the y-dimensional hypercube. Denote by H_i the subgraph of $SH(x, y)$ induced by vertices w such that $p(w) = p_i$, i.e. each H_i is a y-dimensional hypercube. Let

$E_{s1} = \{(u, v) | p(u) = p_i, p(v) = p_{(i+1) mod(x!)}$ and $b(u) = b(v) = b_1$ for even $i\}$,

$E_{s2} = \{(u, v) | p(u) = p_i, p(v) = p_{(i+1) mod(x!)}$ and $b(u) = b(v) = b_2$ for odd $i\}$, and

$E_h = \{$edges on the Hamiltonian path from u to v in $H_i | 0 \le i \le x! - 1\}$.

Then, by the fact that $x!$ is even for $x > 1$, the edges in $E_{s1} \cup E_{s2} \cup E_h$ form a Hamiltonian cycle of $SH(x, y)$. To prove that $SH(x, y)$ is bipartite, we only need to show that there is no cycle in $SH(x, y)$ that contains an odd number of edges. First, in any cycle C of $SH(x, y)$ there must be even number of s-type edges, since otherwise C cannot be a cycle. Mapping all h-type edges in C to an H_i, all images of these edges in H_i form a cycle. Since H_i is a hypercube, and any cycle of H_i contains even number of edges, the total number of h-type edges in C is even. Therefore, $SH(x, y)$ is bipartite.

Now, let us consider the fault-tolerant features of the SH network. A graph is said to be *f-fault tolerant* if it always remains connected when no more than f vertices are removed. The *fault-tolerance* of a graph G is defined as the maximum f for which G is f-fault tolerant. A graph whose fault tolerance is $d - 1$, where d is its degree, is said *maximally fault tolerant*. It is well known that Q_n and S_n are maximally fault tolerant [1, 9]. By the definition of $SH(x, y)$, we know that $SH(x, y)$ is $(x + y - 2)$-fault tolerant, and consequently, it is also maximally fault tolerant. The *fault diameter* of a graph G with fault tolerance f is the maximum diameter of any subgraph obtained from G by deleting f vertices. A family of graphs G_n

is said *strongly resilient* if the fault diameter of G_n is at most $d(G_n) + c$, where $d(G_n)$ is the diameter of G_n and c is a constant. It is known that both Q_n and S_n are strongly resilient, and c is 1 and 3 for Q_n and S_n, respectively [1]. From these results and by the definition of $SH(x, y)$, we know that

Property 5: $SH(x, y)$ is $(x + y - 2)$-fault tolerant, and consequently, it is also maximally fault tolerant. Furthermore, $SH(x, y)$ is strongly resilient, and the constant c in its fault diameter is no greater than 4.

The $SH(x, y)$ is $(x + y - 1)$-connected. Hence, by Menger's theorem, we have the following property of $SH(x, y)$.

Property 6: In $SH(x, y)$, given any two vertices s and d, there exist $(x + y - 1)$ vertex-disjoint paths between them; given a vertex s and a set of $(x + y - 1)$ distinct vertices $\{d_1, d_2, \ldots, d_{x+y-1}\}$, there exist $(x + y - 1)$ vertex-disjoint paths, one from s to each d_i, $1 \le i \le x + y - 1$.

Since $(x + y - 1)$ is the degree of $SH(x, y)$, this property further indicates another optimal fault tolerance feature of $SH(x, y)$. In fact, all these paths can be computed efficiently using the existing algorithms for the star graph and the hypercube [6, 13, 14]. In similar ways, many other properties of the *SH* network can be derived from the properties of the star graph and the hypercube.

5 ALGORITHMS ON THE SH NETWORK

By the structure of the *SH* network, it is natural to believe that the algorithm design techniques for the *SH*-network based computer system should be a combination of those used for the star based and hypercube based systems. To demonstrate this, let us consider two problems: the data broadcasting problem and the sorting problem. Broadcasting a message from any vertex u to all vertices of $SH(x, y)$ can be carried out as follows. First, use the data broadcasting algorithm for the x-star to broadcast the message from u to all vertices in $S = \{v | b(v) = b(u)\}$. This requires $O(x \log x)$ time [1, 4]. Then, in parallel, use the data broadcasting algorithm for the Q_n to broadcast from each $v \in S$ to all vertices in $H_v = \{w | p(w) = p(v)\}$. Clearly, in $O(x \log x + y)$ parallel steps all vertices in $SH(x, y)$ receive the message broadcasted from vertex u. Note that the diameter of $SH(x, y)$ is $\lfloor 3(x - 1)/2 \rfloor + y$. Let $T_{bc}(G)$ denote the time required for broadcasting a message in network G, and $d(G)$ denote the diameter of G. The ratio $d(G)/T_{bc}(G)$ can be used to measure the efficiency of data broadcasting in G. Clearly, the efficiency of data broadcasting in *SH* is no worse than that in the star graph, but no better than that in the hypercube.

The time complexity of best known sorting algorithms on the S_n network for sorting a set of elements, at most one in each processor, is $O(n^3 \log n)$ [4, 12]. By slightly modifying these algorithms and using the hypercube Bitonic sorting algorithm as a subalgorithm, one can obtain a sorting algorithm on $SH(x, y)$ with time complexity $O(x^3 \log x + y \log^3 y)$. Clearly, such an algorithm is more efficient than currently best known sorting algorithms on the star network.

Let $E_H(P)$, $E_S(P)$, and $E_{SH}(P)$ be the efficiencies of best parallel algorithms of the same type (such as *SIMD* or *MIMD*) for solving a problem P on the hypercube, the star and the *SH* networks, respectively. Since the *SH* network is a combination of the hypercube and the star networks, for all problems P, $\min \{E_H(P), E_S(P)\} \le E_{SH}(P) \le \max \{E_H(P), E_S(P)\}$. In general, we believe that $E_H(P) \le E_{SH}(P) \le E_S(P)$.

6 SIMULATING OTHER NETWORKS BY THE SH NETWORK

Given two networks G and H, called the *guest* and the *host,* respectively. An *embedding* f of G into H is a mapping of the vertices of G into the vertices of H. The ratio $|V(H)|/|V(G)|$ is called the *expansion* of f, where $V(G)$ and $V(H)$ denote the vertex sets of G and H, respectively (note: $|V(H)| \le |V(G)|$). The maximum distance between the images of adjacent vertices of G in H is called the *dilation* of f. Graph embedding is used to determine how to simulate a network, the guest, by another, the host. The efficiency of such simulation is measured partly by the dilation and expansion of the embedding. In general, we want to find embeddings with small dilations, which result in small slow down factors of the simulations, and small expansions, which yield better processor utilizations of the simulations. Usually, there are trade-offs between these two parameters.

We have shown that the *SH* network contains Hamiltonian cycles. Thus, the *SH* network can be used to optimally simulate linear arrays and rings (with dilation 1 and expansion 1). The $O(n \log n)$ time data broadcasting algorithm given in [1] implies a dilation 1, one-to-one embedding of a symmetric binary tree of depth $O(n \log n)$ into S_n. By this result, it is easy to verify that there is a dilation 1, one-to-one embedding of a symmetric binary tree of depth

$O(x \log x \div y)$ into $S(x, y)$. Since the SH network contains the hypercube and the star networks as subnetworks, one can expect that in general the performance of simulations of other networks by the SH networks is a trade-off of the performances of simulations of these networks by the hypercube and the star networks.

7 SIMULATING THE HYPERCUBE BY THE SH NETWORK

The star graph has been considered a good alternative to the hypercube as the underlying interconnection structure for parallel multiprocessor systems. Since many efficient hypercube algorithms have been developed, the problem of simulating the hypercube by the star network has been investigated (e.g. [10, 11]). Due to the drastic structural difference between these two networks, the embeddings of the hypercube into the star graph appeared not good, considering both dilation and expansion parameters. Results on trade-offs between dilations and expansions of embeddings of the hypercube into the star graph have been reported in [10, 11]. Since the SH network contains hypercube clusters as subgraphs, it is quite obvious to expect the performance of simulating the hypercube by the SH network to be improved, compared with simulating the hypercube by the star network. Indeed, using any previous results on embedding the hypercube into the star graph, we can obtain embeddings of the hypercube into the SH network with smaller dilations and/or smaller expansions. The following argument illustrates a simple and general method of obtaining hypercube-into-SH embeddings that preserve the dilation, but achieve significant improvement in expansions, compared with the hypercube-into-star embeddings.

Theorem 1: If Q_n can be embedded into $S_{f(n)}$ with dilation k, then Q_n can be embedded into $SH(f(n - p), p)$, $1 \le p \le n - 1$, with dilation at most k.

The idea of the proof of this theorem is as follows. Divide the n bits of the binary vertex labels of Q_n into two parts of p bits and $n - p$ bits, respectively. The binary numbers of the p-bit part are mapped to the binary labels of vertices in $SH(f(n - p), p)$, and binary numbers of the $(n - p)$-bit part are mapped to the permutation labels of vertices in $SH(f(n - p), p)$. Thus, each subgraph of Q_n induced by the same p-bit numbers is mapped to a p-dimensional hypercube cluster of $SH(f(n - p), p)$, for which the dilation is 1. Similarly, the binary numbers of $(n - p)$-bit part are mapped to the $S_{f(n-p)}$ clusters of $SH(f(n - p), p)$ with dilation k.

It is shown in [10], $Q_{n2^{n-1}}$ can be embedded into S_{2^n} with dilation 3. For this embedding, the expansion is $2^n!/2^{n2^n}$. For comparison, we have the following claim.

Corollary 1: $Q_{n2^{n-1}}$ can be embedded into $SH(2^{n-p}, p)$ with dilation 3 and expansion $2^{n-p}!/2^{n2^{n-1}-p}$.

Let $r_{expn}(d_q, d_s, (d_{sh,x}, d_{sh,y})) = expn(d_q, d_s)/expn(d_q, (d_{sh,x}, d_{sh,y}))$, where $expn(d_q, d_s)$ is the expansion of embedding Q_{d_q} into S_{d_s}, and $expn(d_q, (d_{sh,x}, d_{sh,y}))$ is the expansion of embedding Q_{d_q} into $SH(d_{sh,x}, d_{sh,y})$. That is, $r_{expn}(d_q, d_s, (d_{sh,x}, d_{sh,y}))$ is the ratio of the expansions of embeddings of Q_{d_q} into S_{d_s} and $SH(d_{sh,x}, d_{sh,y})$. For $p = n/2$, $r_{expn}(n2^{n-1}, 2^n, (2^{n/2}, n/2)) = 2^n!/(2^{n/2}!2^{n/2})$. It is also shown in [10], $Q_{n\log n+cn}$ can be embedded into S_n with dilation 18, where c is some constant. For this embedding, the expansion is $n!/2^{n\log n+cn}$. By our method, we have the following result.

Corollary 2: $Q_{n\log n+cn}$ can be embedded into $SH(n - p, p)$ with dilation 18 and expansion $(n - p)!/2^{n\log n+cn-p}$.

For $p = n/2$, $r_{expn}(n \log n + cn, n, (n/2, n/2)) \le 2^{n/2}!/(2^{n/2}!2^{n/2})$. The improvement in expansions using SH network and our embedding techniques is significant, compared with embedding the hypercube into the star graph. Similar improvements can be obtained for *all* embedding results reported in [10, 11].

8 SIMULATING THE STAR NETWORK BY THE SH NETWORK

The problem of simulating the star network by the hypercube was considered in [5]. Compared with simulating the star network by the hypercube, the performance of simulating the star network by the SH network can be expected much better simply because that $SH(x, y)$ contains 2^y copies of S_x as subgraphs. Indeed, this is true. However, proving this requires techniques different from that for proving the performance of simulating the hypercube by the SH network. For brevity, we state some of our results and omit proofs. First, we are particularly interested in constant dilation embedding of the star graph into the SH network.

Theorem 2: S_n can be embedded into $SH(p, (N - p)(n - 1))$ with dilation at most 4.

The expansion of this embedding is $p!2^{(n-p)(n-1)}/n!$. The best known constant dilation embedding of the star into the hypercube has expansion $2^{n^2}/n!$. The

improvement in the expansion of constant-dilation star-into-*SH* embedding over the constant-dilation star-into-hypercube embedding is significant. Now, consider the embedding of star network into the *SH* network with small expansion. We have

Theorem 3: S_n can be embedded into $SH(p, (n - 1) + \lceil \log(n - p) \rceil !)$ with dilation at most $\lceil \log(n - p) \rceil ! + 2$.

When $p = n/2$, the dilation is $\lceil \log(n/2) \rceil ! + 2$, and expansion is $n/2$. The known smallest dilation of $(n/2)$-expansion embedding of the star graph into the hypercube is $n(\log n - 2)$ [5]. Therefore, by embedding S_n into the *SH* network we can significantly reduce the dilation without increasing the expansion. It is not difficult to show that the trade-offs of dilations and expansions of embeddings of the star graph into the *SH* networks are always better than those of the best known embeddings of the star graph into the hypercube, the ones given in [5].

9 CONCLUDING REMARKS

We proposed a class of new networks, the *SH* network, which includes the star graph and the hypercube as subclasses. The *SH* network is a simple combination of the star graph and the hypercube, and it has the best features of both networks. By choosing x and y carefully, high-performance parallel computer systems using $SH(x, y)$ network as the underlying interconnection structure can be implemented. When needed, the system can be upgraded easily. The algorithms developed for the star and the hypercube based computer systems can be adopted for the *SH* network with similar or improved performance. Due to the space limit, we only discussed several aspects of the *SH* network. The evidence provided is sufficient to conclude that the *SH* network is a versatile network structure for parallel computation.

Another class of graphs, called the pancake graph, defined on the symmetry group has also been shown an attractive alternative to the hypercube as interconnection network [2]. The pancake graph has properties similar to those of the star graph. As a simple generalization of our approach, we can define a new class of networks, which may be named the *pancake-hypercube hybrid network* (or the *PH network*) in a way similar to the definition of the *SH* network. The *PH* network is a regular graph, and it shares properties of both the pancake graph and the hypercube. By an analysis similar to the one given in this paper, one can show that trade-offs of different aspects can be made in constructing a multiprocessor systems using the *PH* network.

References

[1] S. Akers, D. Harel, and B. Krishnamurthy, "The Star Graph: an Attractive Alternative to the n-Cube," *Proceedings of 1987 International Conference on Parallel Processing,* pp. 393–400.

[2] S. Akers, and B. Krishnamurthy, "A Group Theoretic Model for Symmetric Interconnection Networks," *IEEE Transactions on Computers,* Vol. 38, pp. 555–565, 1989.

[3] S. Akers, and B. Krishnamurthy, "Group Graphs as Interconnection Networks," *Proceedings of 14th International Conference on Fault-Tolerant Computing,* pp. 422–427, 1984.

[4] S.G. Akl, K. Qiu, and I. Stojmenovic, "Data Communication and Computational Geometry onthe Star and Pancake Interconnection Networks," *Proceedings of the 3rd IEEE Symposium on Parallel and Distributed Processing,* pp. 415–422, 1991.

[5] S. Bettayeb, B. Cong, M. Girou, and H. Sudborough, "Simulating Permutation Networks on Hypercubes," *Proceedings of 1st Latin American Symposium on Theoretical Informatics,* Sao Paulo, Brazil, April 1992. *Lecture Notes in Computer Science* Vol. 583, pp. 61–71.

[6] M. Dietzfelbinger, S. Madhavapeddy, and I.H. Sudborough, "Three Disjoint Path Paradigms in Star Networks," *Proceedings of 3rd IEEE Symposium on Parallel and Distributed Processing,* pp. 400–406, 1991.

[7] J. Jwo, S. Lakshmivarahan, and S. Dhall, "Embedding of Cycles and Grids in the Star Graphs," *Proceedings of the 2nd IEEE Symposium on Parallel and Distributed Processing,* pp. 540–547, 1990.

[8] K.H. Kwon, S. Latifi, E.K. Park, and S.Q. Zheng, "Hypercube-Like Networks with Reduced Interconnection Degrees." To appear *Journal of Computer and Software Engineering.*

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

[10] Z. Miller, D. Pritikin, and H. Sudborough, "Small Dilation Embeddings of Hypercubes into Star Networks," Preprint, U. of Texas at Dallas, Richardson, TX 75083.

[11] M. Nigam, S. Sahni, B. Krishnamurthy, "Embedding Hamiltonian and Hypercube in Star Interconnection Graphs," *Proceedings of International Conference on Parallel Processing,* pp. 340–343, 1990.

[12] A. Menn and A.K. Somani, "An Efficient Sorting Algorithm for the Star Interconnection Network," *Proceedings of International Conference on Parallel Processing,* pp. 1–8, 1990.

[13] M.O. Rabin, "Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance," *Journal of ACM,* Vol. 36, No. 2, pp. 335–348, 1989.

[14] Y. Saad and M.H. Schultz, "Topological Properties of Hypercubes," *IEEE Transactions on Computers,* Vol. 37, No. 7, pp. 867–872, 1988.

[15] S. Sur and P.K. Srimani, "Superstar: A New Optimally Fault Tolerant Network Architecture," *Proceedings of 11th International Conference on Distributed Computing Systems,* pp. 590–597, 1991.

Biographies

SI-QING ZHENG received the B.S. degree in Electrical Engineering from Jilin University, China, in 1973. From 1976 to 1980, he was a research engineer at Chinese Academy of Sciences, Beijing, China. He received the M.S. degree in Mathematical Sciences

from the University of Texas at Dallas in 1982, and the Ph.D. degree in Computer Science from the University of California, Santa Barbara, in 1987. In August 1987, he joined the faculty of Department of Computer Science, Louisiana State University, where he is currently an associate professor. Dr. Zheng's research interests include VLSI, parallel and distributed computing, computer networks, and computational geometry. He has published 70 refereed research articles in these and related research areas.

BIN CONG received the B.S. degree in Computer Science from Nanjing University, Nanjing, China, in 1982, and the Ph.D. degree in Computer Science from University of Texas at Dallas in 1991. Presently, he is an assistant professor in the Department of Computer Science at South Dakota State University, Brookings, South Dakota. His research interests include parallel and distributed computing, computer networking, and algorithm design.

SAÏD BETTAYEB received the *Diplôme d'Ingénieur d'État* in Computer Science from the University of Constantine, Algeria, in 1982, and the M.S. and Ph.D. degrees in Computer Science from Northwestern University, Evanston, Illinois, in 1986 and 1988 respectively. He is currently an Assistant Professor in the Department of Computer Science, Louisiana State University. His research interests include combinatorial algorithms, parallel computation, graph layout, and theory of computation.