Joint Conference of 1996 International Computer Symposium
December 19~21, Kaohsiung, Taiwan, R.O.C.

Optimal Processor Mapping for Linear-Constant Communication on k-ary n-cubes*

Chien-Min Wang[a], Yo-Min Hou[b] and Chiu-Yu Ku[a]

[a]Institute of Information Science, Academia Sinica, Taipei, Taiwan, ROC
Email:{cmwang, cyk}@iis.sinica.edu.tw,
[b]Institute of Computer and Information Science, National Chiao-Tung University
Hsinchu, Taiwan, ROC
Email:ymhou@winston.cis.nctu.edu.tw

Abstract

In this paper, we address the problem of minimizing link contention of linear-constant communication on wormhole-routed k-ary n-cubes. Our research reveals that, for dimension ordered routing algorithms, the degree of link contention of a linear-constant communication can be quite large. To solve this problem, we propose a new approach called processor mapping. In our approach, processors are remapped according to the given communication(s) so that the new communication(s) can be efficiently realized on the k-ary n-cube network. It is proved that for any set of m linear-constant communications, m≤k-1, there exists a processor mapping such that the new communications have minimum link contention. Several computer simulations are conducted and the results clearly show the advantage of the proposed approach.

1. Introduction

A distributed memory parallel computer consists of a large number of identical processing elements and an interconnection network. Each processing element has its own processor, local memory, and other supporting devices. Processors communicate by sending messages through the interconnection network. There are many different kinds of interconnection networks been used to build parallel computers. The most popular ones are k-ary n-cubes and their variants such as rings, meshes, tori, cubes, binary n-cubes, and Omega networks.

The problem of moving messages among processors is called the *message routing problem*. Many researches on the message routing problem are based on the store-and-forward routing, where the message latency is proportional to the product of the message length and the number of routing steps. Hence, most of them concentrated on minimizing the number of routing steps in moving messages among processors. On the other hand, wormhole routing has been widely adopted recently [1], [2], [23] due to its effectiveness of inter-processor communication. With wormhole routing, each message is divided into numbers of flits. The header flit(s) carries address information and governs the route while the remaining flits of the message follow in a pipeline fashion. One of the attractions of wormhole routing is that the storage requirement for each router is significantly less than that of the store-and-forward routing. Another attraction of wormhole routing is that its message latency is much lower than that of store-and-forward routing. In the absence of link contention, the message latency of wormhole routing is relatively insensitive to path length [8], [23].

In this paper, we focus on the problem of minimizing link contention of linear-constant communication on wormhole-routed k-ary n-cubes. Linear-constant communication is a class of communications where the address vectors of destination processors are linear combinations of the address vectors of source processors plus a constant vector. Many important problems like fast Fourier transform, matrix transposition, polynomial evaluation, etc., can be effectively solved in parallel computers which have an efficient scheme to support this type of communications. The problem of moving messages between pairs of source and destination processors can be treated as the problem of realizing the corresponding communication using the k-ary n-cube network among the processors.

Traditional approaches to the message routing problem are trying to find an efficient routing algorithm [6], [7], [9], [11], [15], [16], [26]. However, our research reveals that, for dimension ordered routing algorithms, the degree of link contention of a linear-constant communication can be quite large. To solve this problem, we adopt a new approach, called *processor mapping*. In our approach, processors are mapped according to the given communication(s) so that the new communication(s) can be effi-

*This work was supported by National Science Council, Republic of China, under Grant NSC86-2213-E-001-006.

ciently realized on the k-ary n-cube network. We can prove that for any set of m linear-constant communications, $m \le k-1$, there exists a processor mapping such that the new communications have minimum link contention. We also propose an algorithm to find an optimal processor mapping. Experiments based on computer simulation are conducted and the results clearly show the advantage of the proposed approach.

The rest of this paper is organized as follows. Section 2 introduces the notations, definitions and necessary information. In section 3, we present the concept and theorems of processor mapping. Experimental results based on computer simulation will be shown in Section 4. Finally, conclusions are given in Section 5.

2. Background

The target parallel computer is a k-ary n-cube computer with wormhole routing capability. The routing paths are assumed to be dimension ordered shortest paths. The goal is to perform linear-constant communication efficiently on the k-ary n-cube computer such that the degree of link contention can be minimized. The definitions and notations of these terminologies will be clarified in this section.

2.1 k-ary n-cubes

A k-ary n-cube interconnection network consists of $N=k^n$ nodes in n dimensions and k nodes along each dimension. Each node x in a k-ary n-cube can be identified by an n-digit radix-k integer, $x_{n-1}x_{n-2}...x_0$, where $0 \le x_i \le k-1$ for $0 \le i \le n-1$. We shall use an n-dimensional vector $(x_0, x_1,...,x_{n-1})^t$ to represent it. The ith digit of the address vector, x_i, represents the node's position in the ith dimension. Two nodes x and y are neighbors if and only if there exists j such that $x_j=(y_j \pm 1) \bmod k$ and $x_i=y_i$ for all $i \ne j$. There are two unidirectional links between two neighboring nodes, one for each direction. Note that the use of modular arithmetic in the definition results in *wraparound links* in k-ary n-cubes. When $k>2$, each node has $2n$ neighbors, two in each dimension. When $k=2$, a special case for hypercubes, each node has n neighbors, one in each dimension. Another special case, when $n=1$, is a ring with k nodes.

2.2 Routing strategies

The path selection for a message traversal is based on the dimension ordered routing strategy which reserves links required for a communication path in a strictly increasing order of dimensions. Within each dimension, shortest paths are chosen. For binary hypercubes, this algorithm ensures deadlock-free routing but only one path for a given source-destination pair is allowed. Augmentation must be made to implement deadlock-free routing in k-ary n-cubes because deadlock may occur within a dimension due to the wraparound links. As implemented in the supercomputer Cray T3D[1], virtual channels can be used to prevent this problem. All links are broken up into pairs of virtual channels (high and low channels). They can be implemented by time-multiplexing onto the associated physical links. Upon entering a new dimension (either from another dimension or an injection channel), a message is routed along the high virtual channels if it will not traverse wraparound links. Otherwise it is routed along the low virtual channels in that dimension. This routing restriction coupled with the dimension ordered routing is sufficient for providing deadlock-free routing in k-ary n-cubes.

2.3 Linear-constant communication

In what follows, we define the class of linear-constant communications of a k-ary n-cube with $N=k^n$ nodes, where k is a power of 2. Since every node in the k-ary n-cube can be denoted as a radix-k n-dimensional vector, $(d_0, d_1,..., d_{n-1})^t$, and k is a power of 2, the addition and multiplication in the following definitions can be defined to be in the finite field, **GF(k)** [19].

Definition: A communication is a *linear-constant communication* if there exists a matrix $A_{n\times n}$ and an n-dimensional vector b such that, for every source node x, its destination node is given by the equation, $y=Ax+b$.

Definition: A linear-constant communication is a *linear-constant permutation* if the matrix $A_{n\times n}$ is non-singular, i.e., $rank(A_{n\times n})=n$.

Definition: A linear-constant communication with a matrix $A_{n\times n}$ and an n-dimensional vector b is a *linear-constant gather* if $rank(A_{n\times n})<n$.

Scatter, the dual operation of gather, can be implemented by simply reversing the direction of message transmission of gather. Thus, we can define *linear-constant scatter* as follows.

Definition: A communication is said to be a *linear-constant scatter* if there exists a matrix $A_{n\times n}$, $rank(A_{n\times n})<n$, and an n-dimensional vector b such that, for every destination node y, its source node x is given by the equation, $Ay+b=x$.

Example 1. The communication of matrix-transpose on a k-ary 4-cube is a linear-constant communication with $A_{mt}=\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$ and $b_{mt}=\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$. □

A linear-constant communication defined above is

equivalent to perform a linear transformation of the n-dimensional vector space over the field GF(k) and then add a constant vector. There is one-to-one correspondence between the radix-k matrix of size $n \times n$ and linear transformations on n-dimensional vector space over GF(k). We shall utilize this property in the following sections.

3. Processor Mapping

Link contention can have a severe impact on the throughput of the interconnection network. Let the degree of link contention be defined as the maximum number of paths that contend for the same link in the interconnection network. The goal of this paper is to minimize the degree of link contention for linear-constant communication on dimension ordered wormhole-routed k-ary n-cubes.

First, we shall show that the number of paths contending for the same link is directly related to the ranks of sub-matrices of the matrix A. A sub-matrix of the matrix A is the matrix obtained from A by retaining entries in some row(s) and column(s) and deleting other entries. We use $A_{(i)}$ to denote the ith row of A and $A^{(j)}$ to denote the jth column of A. The following definition defines special sub-matrices and their notations to be used in this paper.

Definition: A sub-matrix of the matrix A obtained by retaining rows in the set R and columns in the set C is denoted as $A_{R,C}$.

We shall use L_i to denote the set of non-negative integers smaller than i. So, given integers i and j, A_{L_i,L_j} is the upper-left sub-matrix of A with i rows and j columns. Example 2 shows the sub-matrix A_{L_3,L_2} of a 4×4 matrix A.

Example 2. Given $A=\begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix}$, A_{L_3,L_2} is equal to $\begin{bmatrix} a_{0,0} & a_{0,1} \\ a_{1,0} & a_{1,1} \\ a_{2,0} & a_{2,1} \end{bmatrix}$. □

Theorem 1: Given a linear-constant communication $y=Ax+b$, the maximum number of paths that contend for the same link at dimension i, denoted as $T_i(A,b)$, can be determined as follows:

(1). if $rank(A_{L_i,L_i})+1=rank(A_{L_i,L_{i+1}})=rank(A_{L_{i+1},L_{i+1}})$, then

$$T_i(A,b)=\begin{cases} 0 & , x_i = y_i \\ k \times k^{i-rank(A_{L_i,L_{i+1}})} & , \text{otherwise.} \end{cases}$$

(2). If $rank(A_{L_i,L_i})+2=rank(A_{L_i,L_{i+1}})+1=rank(A_{L_{i+1},L_{i+1}})$, then $T_i(A,b)=\frac{k}{2}\times k^{i-rank(A_{L_i,L_{i+1}})}$.

(3). if $rank(A_{L_i,L_i})=rank(A_{L_i,L_{i+1}})=rank(A_{L_{i+1},L_{i+1}})$, then

$$T_i(A,b)=\frac{k}{2}\times k^{i-rank(A_{L_i,L_{i+1}})}.$$

(4). if $rank(A_{L_i,L_i})+1=rank(A_{L_i,L_{i+1}})+1=rank(A_{L_{i+1},L_{i+1}})$ and

(a). $rank(A_{L_{i+1},L_i})=rank(A_{L_{i+1},L_{i+1}})$, then

$$T_i(A,b)=\frac{(2+k)}{8}\times k^{i-rank(A_{L_i,L_{i+1}})}.$$

(b). $rank(A_{L_{i+1},L_i})+1=rank(A_{L_{i+1},L_{i+1}})$, then

$$0 \le T_i(A,b) \le \frac{k}{2}\times k^{i-rank(A_{L_i,L_{i+1}})}.$$

Proof: For any ring (k-ary 1-cube) r at dimension i, suppose that r has the set of nodes $\{(z_0,z_1,\ldots,z_{i-1},w,z_{i+1},\ldots,z_{n-1})^t \mid 0 \le w < k\}$. If there is a path from node x to node y that passes some link of r, according to the dimension ordered routing, it must hold that $x=(x_0,x_1,\ldots,x_i,z_{i+1},\ldots,z_{n-1})^t$ and $y=(z_0,z_1,\ldots,z_{i-1},y_i,\ldots,y_{n-1})^t$. Since $y=Ax+b$, we have

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{i-1} \end{bmatrix} = \begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_{i-1} \end{bmatrix} = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,i} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,i} \\ \cdots & \cdots & \cdots & \cdots \\ a_{i-1,0} & a_{i-1,1} & \cdots & a_{i-1,i} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_i \end{bmatrix} + \begin{bmatrix} a_{0,i+1} & a_{0,i+2} & \cdots & a_{0,n-1} \\ a_{1,i+1} & a_{1,i+2} & \cdots & a_{1,n-1} \\ \cdots & \cdots & \cdots & \cdots \\ a_{i-1,i+1} & a_{i-1,i+2} & \cdots & a_{i-1,n-1} \end{bmatrix} \begin{bmatrix} z_{i+1} \\ z_{i+2} \\ \vdots \\ z_{n-1} \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{i-1} \end{bmatrix}$$

Since a path from x to y must pass links from node $(y_0, y_1,\ldots,y_{i-1},x_i,x_{i+1},\ldots,x_{n-1})^t$ to node $(y_0,y_1,\ldots,y_{i-1},y_i,x_{i+1},\ldots,x_{n-1})^t$ at dimension i, there must exists some ring at dimension i that solutions for $(x_0,x_1,\ldots,x_i)$ exist. Hence, we can consider only the situation that there exist solutions for $(x_0, x_1,\ldots,x_i)$ in the following proof.

According to Linear Algebra, the number of solutions for $(x_0,x_1,\ldots,x_i)$ must be $k^{i+1-rank(A_{L_i,L_{i+1}})}$. This is also the number of paths passing links of r. Similarly, we can find the number of solutions of $(x_0,x_1,\ldots,x_{i-1})$ to be $k^{i-rank(A_{L_i,L_i})}$. Therefore, by comparing the ranks of the A_{L_i,L_i} and $A_{L_i,L_{i+1}}$, we know that

(a). if $rank(A_{L_i,L_{i+1}})=rank(A_{L_i,L_i})+1$, there is only one solution for x_i and all the paths passing links of r arrive at the same node.

(b). if $rank(A_{L_i,L_{i+1}})=rank(A_{L_i,L_i})$, there are k different solutions for x_i and the paths passing links of r are divided evenly into k groups. Paths in different groups arrive at different nodes.

Similarly, by comparing the ranks of $A_{L_i,L_{i+1}}$ and $A_{L_{i+1},L_{i+1}}$, we know the number of solutions for y_i and that

(a). if $rank(A_{L_i,L_{i+1}})=rank(A_{L_{i+1},L_{i+1}})$, there is only one solution for y_i and all the paths passing links of r leave r at the same node.

(b). if $rank(A_{l_i,l_{i+1}})+1=rank(A_{l_{i+1},l_{i+1}})$, there are k different solutions for y_i and the paths passing links of r are divided evenly into k groups. Paths in different groups leave at different nodes.

From the above discussion, we have

(1). if $rank(A_{l_i,l_i})+1=rank(A_{l_i,l_{i+1}})=rank(A_{l_{i+1},l_{i+1}})$, then all the paths passing links of r arrive at one node, and leave at one node. If these two nodes are the same one, then there is no link contention. Otherwise all the paths contend for links between these two nodes.

Hence, $T_i(A,b)=\begin{cases}0 & , x_i=y_i\\ k\times k^{i-rank(A_{l_i,l_{i+1}})} & , \text{otherwise.}\end{cases}$

(2). if $rank(A_{l_i,l_i})+2=rank(A_{l_i,l_{i+1}})+1=rank(A_{l_{i+1},l_{i+1}})$, then the paths passing links of r arrive at one node, and leave at the k nodes of r. Hence,

$$T_i(A,b)=\frac{k}{2}\times k^{i-rank(A_{l_i,l_{i+1}})}.$$

(3). if $rank(A_{l_i,l_i})=rank(A_{l_i,l_{i+1}})=rank(A_{l_{i+1},l_{i+1}})$, then the paths passing links of r arrive at the k nodes of r, and leave at one node. Hence,

$$T_i(A,b)=\frac{k}{2}\times k^{i-rank(A_{l_i,l_{i+1}})}.$$

(4). if $rank(A_{l_i,l_i})+1=rank(A_{l_i,l_{i+1}})+1=rank(A_{l_{i+1},l_{i+1}})$, then the paths passing links of r arrive at the k nodes of r, and leave at the k nodes of r. Here are two possible cases: k-to-k permutation or scatter-gather. By given the value of x_i to find the number of solutions for y_i, we can distinguish these two cases. This can be done by compare the ranks of A_{l_{i+1},l_i} and $A_{l_{i+1},l_{i+1}}$ Thus,

(a). if $rank(A_{l_{i+1},l_i})=rank(A_{l_{i+1},l_{i+1}})$, then it's a k-to-k scatter-gather.

$$T_i(A,b)=\frac{(2+k)}{8}\times k^{i-rank(A_{l_i,l_{i+1}})}.$$

(b). if $rank(A_{l_{i+1},l_i})+1=rank(A_{l_{i+1},l_{i+1}})$, then it's a k-to-k permutation.

$$0\le T_i(A,b)\le\frac{k}{2}\times k^{i-rank(A_{l_i,l_{i+1}})}.$$ ❑

The above theorem shows that the degree of link contention is determined only by the ranks of sub-matrices of the matrix A. As an example, we shall compute the degree of link contention of matrix-transpose on a 4-ary 4-cube according to Theorem 1.

Example 3. Consider the matrix-transpose in Example 1 on a 4-ary 4-cube. According to Theorem 1, the degree of link contention at each dimension can be computed as follows: $R_0=R_3=\frac{4}{2}\times 4^0=2$ and $R_1=R_2=\frac{4}{2}\times 4^1=8$. By examine the behavior of performing matrix-transpose on a 4-ary 4-cube, it can be verified that link contention happened just as computed above. ❑

Definition: The degree of link contention of a linear-constant communication, $y=Ax+b$, is defined to be $\underset{0\le i\le n-1}{MAX}\{T_i(A,b)\}$, which is denoted by $T(A,b)$.

For convenient, we may use T instead of $T(A,b)$ in case of no confusion. To minimize $T(A,b)$, the matrix A must be "changed" and, at the same time, the linear-constant communication must be correctly performed. To meet these two requirements, we propose a new approach, called *processor mapping*. In this approach, processors are remapped by a linear mapping, which is defined as follows.

Definition: A processor mapping is said to be a *linear mapping* if there exists a matrix $Q_{n\times n}$ such that, for every node x, it is mapped to node $x'=Qx$.

Definition: A processor mapping is a *reordering mapping* if it is a linear mapping with matrix Q and Q is a permutation matrix, i.e., each row and column of Q has exactly one 1.

Definition: A matrix is called an *operator matrix* if it is non-singular. [13]

In order to keep the linear-constant communication to be performed correctly, the communication after processor mapping must be changed as shown in the following theorem.

Theorem 2: Given a linear-constant communication with a matrix A and a vector b, the new communication after the linear mapping with an operator matrix Q is a linear-constant communication with a matrix-QAQ^{-1} and a vector Qb.

Proof: Fig. 1 is a good explanation for this theorem. For a source node x in the communication with A and b, the destination node y can be computed by the equation, $y=Ax+b$. After linear mapping by Q, we have $x'=Qx$ and $y'=Qy$. Since Q is non-singular, we can derive $x=Q^{-1}x'$. Hence, we have $y'=Qy=Q(Ax+b)=QAQ^{-1}x'+Qb$. ❑

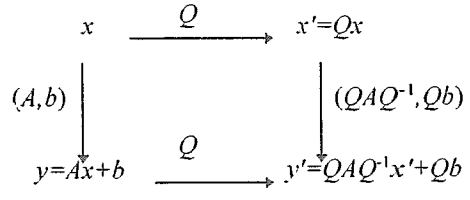

Figure 1. The new communication after linear mapping with an operator matrix Q for $y=Ax+b$.

In other words, the degree of link contention after the linear mapping is determined by QAQ^{-1}. By choosing an appropriate linear mapping Q, the degree of link contention could be greatly reduced. In the following, we will

begin to find an optimal linear mapping for a linear-constant communication. First, we notice that, there are two situations in Theorem 1 that $T_i(A,b)=0$ at dimension i. They only happen when $y_i=x_i$ for all x, i.e., $A_{(i)}=I_{(i)}$ and $b_i=0$. Since $T_i(A,b)=0$ means no communication at dimension i, we would like to keep it unchanged when performing processor mapping. It can be accomplished by applying reordering mapping Q before applying other mapping, where $Q_{(j')}=I_{(j)}$ for $j'=\begin{cases} j & , 0 \le j \le i-1 \\ j-1, & i+1 \le j \le n-1. \\ n-1, & j=i \end{cases}$

Because $\left[QAQ^{-1}\right]_{I_{n-1},I_{n-1}} = A_{I_n-\{i\},I_n-\{i\}}$, we can derive $T_i(QAQ^{-1},Qb)=T_j(A,b)$. Hence, we can just consider the sub-matrix $A_{I_n-\{i\},I_n-\{i\}}$ when minimizing link contention of A. If there is a set of rows R such that $T_s(A,b)=0$ for $s\in R$, then, by recursively applying above method, we can just consider the sub-matrix A_{I_n-R,I_n-R} when minimizing link contention of A. Without loss of generality, we may assume $T_i(A,b)>0$ for $0\le i\le n-1$ in the following discussion. As proved in the following theorems and corollaries, there must exists a linear mapping such that the new communication has minimum link contention.

Theorem 3: For any matrix $A_{n\times n}$, there exists an operator matrix $Q_{n\times n}$ such that

$$rank(\left[QAQ^{-1}\right]_{I_i,I_i})=i, \text{ for } 1\le i\le rank(A).$$

Proof: The proof is by induction on the integer i. First we shall prove this theorem is true for $i=1$. If $a_{0,0}\neq 0$, then $rank(A_{I_1,I_1})=1$. If $a_{0,0}=0$, let $A^{(j)}$ be a non-zero column and Q be the permutation matrix exchanging $A_{(j)}$ and $A_{(0)}$. Accordingly, Q^{-1} must be the permutation matrix exchanging $A^{(j)}$ and $A^{(0)}$. Hence, there exists l such that $(QAQ^{-1})_{l,0}\neq 0$. Let $\hat{Q}$ be the operator matrix adding $A_{(l)}$ to $A_{(0)}$. Accordingly, $\hat{Q}^{-1}$ must be the operator matrix subtracting $A^{(l)}$ by $A^{(0)}$. We can derive

$$rank(\left[\hat{Q}\left(QAQ^{-1}\right)\hat{Q}^{-1}\right]_{I_1,I_1})=rank(\left[\left(\hat{Q}Q\right)A\left(\hat{Q}Q\right)^{-1}\right]_{I_1,I_1})=1.$$

Hence, this theorem is true for $i=1$.

Now suppose that it is true for $1\le i\le k$. There must exists an operator matrix Q such that $rank(\left[QAQ^{-1}\right]_{I_i,I_i})=i$ for $1\le i\le k<rank(A)$. We shall prove that it is also true for $1\le i\le k+1$. If $rank(\left[QAQ^{-1}\right]_{I_{k+1},I_{k+1}})\neq k+1$, it must held that $rank(\left[QAQ^{-1}\right]_{I_{k+1},I_{k+1}})=k$. Since Q and Q^{-1} are non-singular, we have $rank(QAQ^{-1})=rank(A)$. There must exists a column $A^{(j)}$, $j\ge k$, that is linearly independent of columns in $\left[QAQ^{-1}\right]_{I_n,I_k}$. Let $\hat{Q}$ be the permutation matrix exchanging $A_{(j)}$ and $A_{(k)}$. Accordingly, $\hat{Q}^{-1}$ must be the permutation matrix exchanging $A^{(j)}$ and $A^{(k)}$. Hence, we have $rank(\left[\hat{Q}\left(QAQ^{-1}\right)\hat{Q}^{-1}\right]_{I_n,I_{k+1}})=k+1$. In other words, there must exists a row l, $l\ge k$, in the sub-matrix $\left[\hat{Q}\left(QAQ^{-1}\right)\hat{Q}^{-1}\right]_{I_n,I_{k+1}}$ that is linearly independent of rows in $\left[\hat{Q}\left(QAQ^{-1}\right)\hat{Q}^{-1}\right]_{I_k,I_{k+1}}$. Let $\tilde{Q}$ be the operator matrix adding $A_{(l)}$ to $A_{(k)}$. Accordingly, $\tilde{Q}^{-1}$ must be the operator matrix subtracting $A^{(l)}$ by $A^{(k)}$. We can derive

$$rank(\left[\tilde{Q}\left(\hat{Q}\left(QAQ^{-1}\right)\hat{Q}^{-1}\right)\tilde{Q}^{-1}\right]_{I_i,I_i})=i$$

for $1\le i\le k+1\le rank(A)$.

Since $\left(\tilde{Q}\hat{Q}Q\right)^{-1}=Q^{-1}\hat{Q}^{-1}\tilde{Q}^{-1}$, it can be derived that

$$rank(\left[\left(\tilde{Q}\hat{Q}Q\right)A\left(\tilde{Q}\hat{Q}Q\right)^{-1}\right]_{I_i,I_i})=i$$

for $1\le i\le k+1\le rank(A)$.

Since $\left(\tilde{Q}\hat{Q}Q\right)$ is also an operator matrix, this theorem is true for $1\le i\le k+1\le rank(A)$. Therefore, by mathematic induction, this theorem must be true for $1\le i\le rank(A)$. This complete the proof of this theorem. □

Corollary 1: For any linear-constant communication, the new communication with the linear mapping in Theorem 3 has minimum link contention.

Proof: For any linear-constant communication $y=Ax+b$, from Theorem 1, $T_i(A,b)$ can be rewritten as $\alpha_i \times k^{i-rank(A_{I_i,I_{i+1}})}$ for all i, $0\le i\le n-1$, where $1\le \alpha_i \le k$. Hence, we can consider only i-$rank(A_{I_i,I_{i+1}})$ for each dimension i in the following proof.

(i) First, consider the case $rank(A)=n$. For any linear mapping Q, note that $\underset{0\le i\le n-1}{MAX}\left\{i-rank(\left[QAQ^{-1}\right]_{I_i,I_{i+1}})\right\}$ ≥ 0. From Theorem 3, there exists a linear mapping $\tilde{Q}$ such that i-$rank(\left[\tilde{Q}A\tilde{Q}^{-1}\right]_{I_i,I_{i+1}})=0$, for $0\le i\le n-1$. Hence, this theorem is true for a linear-constant permutation.

(ii) Then, consider the case $rank(A)<n$. For any linear mapping Q, note that $\underset{0\le i\le n-1}{MAX}\left\{i-rank(\left[QAQ^{-1}\right]_{I_i,I_{i+1}})\right\}$ $\ge (n-1)-rank(\left[QAQ^{-1}\right]_{I_{n-1},I_n})\ge (n-1)-rank(A)$. From Theorem 3, there exists a linear mapping $\tilde{Q}$ such that i-$rank(\left[\tilde{Q}A\tilde{Q}^{-1}\right]_{I_i,I_{i+1}})=0$, for $0\le i\le rank(A)$, and i-$rank(\left[\tilde{Q}A\tilde{Q}^{-1}\right]_{I_i,I_{i+1}})=i$-$rank(\tilde{Q}A\tilde{Q}^{-1})$, for $rank(A)+1\le i\le n-1$.

Hence,

$$\underset{0\le i\le n-1}{MAX}\left\{ i-rank\left(\left[\tilde{Q}A\tilde{Q}^{-1}\right]_{I_{i+1}.I_i}\right)\right\}$$

$$=(n\text{-}1)\text{-}rank(\tilde{Q}A\tilde{Q}^{-1})=(n\text{-}1)\text{-}rank(A).$$

Therefore, this theorem is also true for a linear-constant scatter/gather.

From the above discussion, we know that the new communication with the linear mapping in Theorem 3 has minimum link contention. ❑

Corollary 2: For any linear-constant permutation $y=Ax+b$, there exists a linear mapping Q such that $T_i(QAQ^{-1},Qb) \le \frac{k}{2}$, for $0 \le i \le n$-1.

Corollary 3: For any linear-constant gather/scatter $y=Ax+b$, there exists a linear mapping Q such that $T_i(QAQ^{-1},Qb) \le \frac{k}{2} \times k^{n-1-rank(A)}$ for $0 \le i \le n$-1.

Example 4. Consider the matrix-transpose in Example 3 on a 4-ary 4-cube. From Theorem 3, we can find Q, Q^{-1}, and the new communication as follows,

$$Q=Q^{-1}=\begin{bmatrix}1&0&1&0\\0&1&0&1\\0&0&1&0\\0&0&0&1\end{bmatrix} \text{ and } \begin{bmatrix}y'_0\\y'_1\\y'_2\\y'_3\end{bmatrix}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\1&0&1&0\\0&1&0&1\end{bmatrix}\begin{bmatrix}x'_0\\x'_1\\x'_2\\x'_3\end{bmatrix}+\begin{bmatrix}0\\0\\0\\0\end{bmatrix}.$$

The degree of link contention at each dimension for the new communication becomes $T_0=T_1=0$ and $T_2=T_3=\frac{4}{2}\times 4^0=2$. Compared with Example 3, the degree of link contention is greatly reduced from 8 to 2 by the linear mapping.❑

Lemma 1: Let $A_0, A_1,..., A_{m-1}$ be m different matrices with size $n\times n$ in GF(k) and $m\le k$-1. If $rank(\left[A_0\right]_{I_n.I_h}) < h \le rank(A_0)$, then there exists an operator matrix Q such that

$rank(\left[QA_0Q^{-1}\right]_{I_n.I_h})=rank(\left[A_0\right]_{I_n.I_h})+1$, and

$rank(\left[QA_rQ^{-1}\right]_{I_n.I_h}) \ge rank(\left[A_r\right]_{I_n.I_h})$ for any A_r, $1\le r\le m$-1.

The above lemma is concerned with columns of matrices. It can be easily shown that it also holds for rows. Hence, by applying Lemma 1 to Theorem 4, we may find an optimal linear mapping for no more than k-1 linear-constant communications as described in the following theorem and corollary.

Theorem 4: Given m matrices $A_0, A_1,..., A_{m-1}$, $m\le k$-1, there exists a linear mapping with an operator matrix $Q_{n\times n}$ such that $rank(\left[QA_rQ^{-1}\right]_{I_i.I_i})=i$, $1\le i\le rank(A_r)$, for any A_r, $0\le r\le m$-1.

Proof: The proof of this theorem is similar to the proof of Theorem 3 except that, at each step, if we need to find an operator matrix, we have to find one for all the m matrices A_r, $0\le r\le m$-1 by Lemma 1. This completes the proof of this theorem. ❑

Corollary 4: Let $\{y=A_rx+b_r | 0\le r\le m\text{-}1\}$ be a set of m linear-constant communications to be performed, where $m\le k$-1. With the linear mapping Q in Theorem 4, each new communication $y=QA_rQ^{-1}+Qb_r$, $0\le r\le m$-1, has minimum link contention.

Corollary 2 and 3 can also be applied to Theorem 4 to find the degree of link contention for the new communications. Following the method in the proof of Theorems 3 and 4, we can easily design an algorithm to find an optimal linear mapping for any m different linear-constant communications, $m\le k$-1. Example 5 shows a linear mapping Q obtained from above theorems for two linear-constant communications: matrix-transpose and digit-reverse.

Example 5. Consider the folowing linear-constant communications,

(1) matrix-transpose, as in Example 1, and

(2) digit-reverse, $\begin{bmatrix}y_0\\y_1\\y_2\\y_3\end{bmatrix}=\begin{bmatrix}0&0&0&1\\0&0&1&0\\0&1&0&0\\1&0&0&0\end{bmatrix}\begin{bmatrix}x_0\\x_1\\x_2\\x_3\end{bmatrix}+\begin{bmatrix}0\\0\\0\\0\end{bmatrix}$,

on a 4-ary 4-cube. From Theorem 4 we can find that $Q=Q^{-1}=\begin{bmatrix}1&0&1&1\\0&1&2&1\\0&0&1&0\\0&0&0&1\end{bmatrix}$, and the new communications are

(1') $\begin{bmatrix}y'_0\\y'_1\\y'_2\\y'_3\end{bmatrix}=\begin{bmatrix}1&1&2&0\\2&1&0&2\\1&0&1&1\\0&1&2&1\end{bmatrix}\begin{bmatrix}x'_0\\x'_1\\x'_2\\x'_3\end{bmatrix}+\begin{bmatrix}0\\0\\0\\0\end{bmatrix}$ and

(2') $\begin{bmatrix}y'_0\\y'_1\\y'_2\\y'_3\end{bmatrix}=\begin{bmatrix}1&1&3&1\\1&2&3&3\\0&1&2&1\\1&0&1&1\end{bmatrix}\begin{bmatrix}x'_0\\x'_1\\x'_2\\x'_3\end{bmatrix}+\begin{bmatrix}0\\0\\0\\0\end{bmatrix}$.

After the linear mapping, the degree of link contention now becomes 2 instead of 8 in the original communications. ❑

4. Performance Study

To compare the dimension ordered routing with our approach, we make some experiments by simulating the network behavior of a k-ary n-cube. Two neighboring routers are connected by a pair of uni-directional links. Each link is broken up into pairs of virtual channels (high and low channels). A router can communicate with its local processor through a pair of ports. A separate buffer with a slot for one flit is associated with each virtual channel. The flit at the buffer is transmitted in a cycle to the router connected at the other end of the link. The

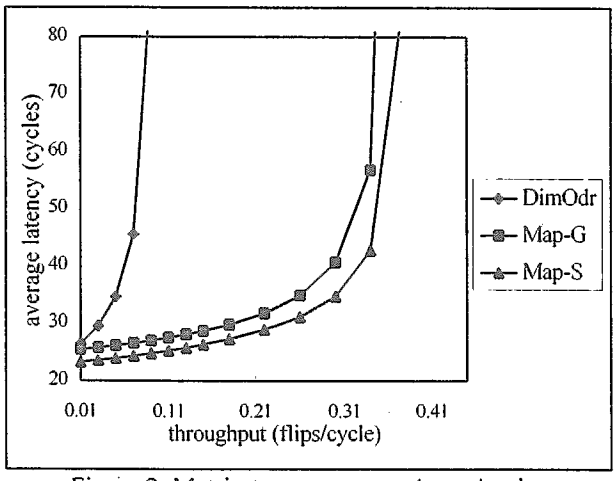

Figure 2. Matrix-transpose on a 4-ary 4-cube.

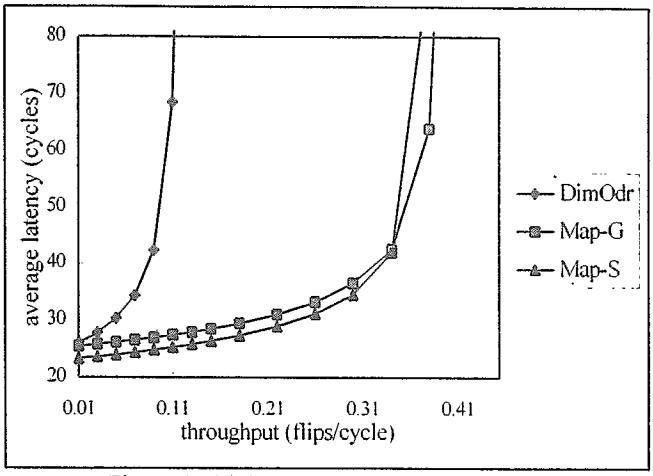

Figure 3. Digit-reverse on a 4-ary 4-cube.

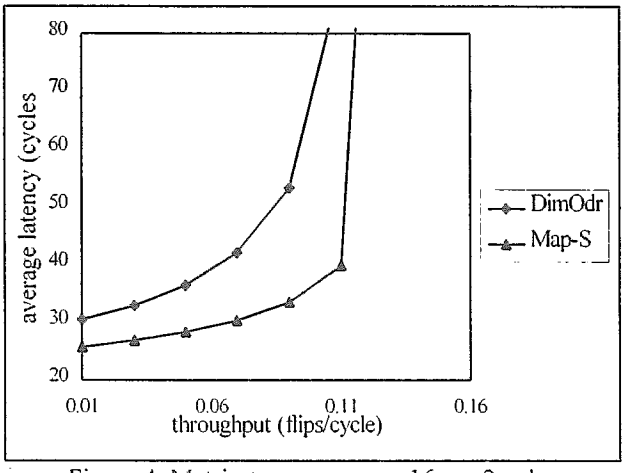

Figure 4. Matrix-transpose on a 16-ary 2-cube.

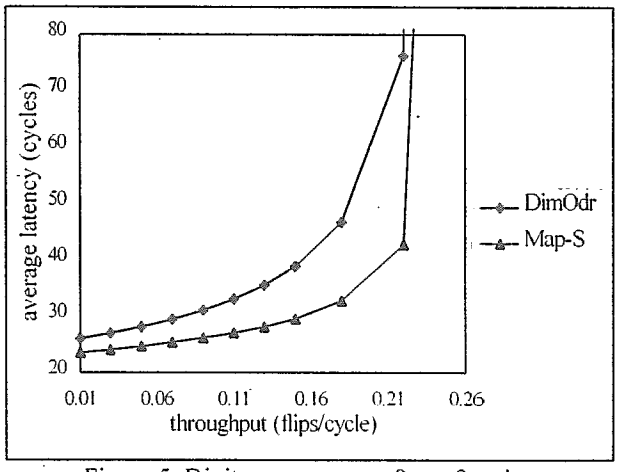

Figure 5. Digit-reverse on an 8-ary 3-cube.

processors generate messages at time intervals given by a negative exponential distribution random variable. Each message has 20 flits. When more than one input link contain header flits waiting for the same available output link, the arbitration policy is in favor of the header flit that arrived at the router first.

The measures of interest in this performance study are average message latency and average sustainable network throughput. The message latency is the number of cycles spent by a message in traveling from its source processor to its destination, taking the queuing delay into account. The average network throughput indicates the average number of flits delivered per cycle per processor. It is sustainable if the number of messages queued at their source processors is small and bounded. For a given system, the average message latency grows as the throughput increases. At low throughput, the message latency is contributed mainly by the message length because there is little queuing delay involved. As the throughput increases, more link contention and longer queuing delay happened, give rise to a higher message latency. One system exhibits better communication performance than another if it has a lower message latency for any given throughput.

Network performance is significantly effected by the communication patterns, which are application dependent. In this paper, we consider three different communications: *matrix-transpose*, *digit-reverse*, and *uniform-random*. They are chosen because they are frequently used in many scientific and engineering applications. When performing matrix-transpose, each node $(d_0,d_1,\ldots,d_{n/2-1},d_{n/2},\ldots,d_{n-1})^t$ sends messages to node $(d_{n/2},d_{n/2+1},\ldots,d_{n-1},d_0,d_1,\ldots,d_{n/2-1})^t$. Figs. 2 and 4 show the simulation results of matrix-transpose on a 4-ary 4-cube and a 16-ary 2-cube, respectively. Both of them can be used to transpose a 16×16 matrix. When performing digit-reverse, each node $(d_0, d_1,\ldots,d_{n-1})^t$ sends messages to node $(d_{n-1},d_{n-2},\ldots,d_0)^t$. It is useful when performing a radix-k FFT. Figs. 3 and 5 show the simulation results of digit-reverse on a 4-ary 4-cube

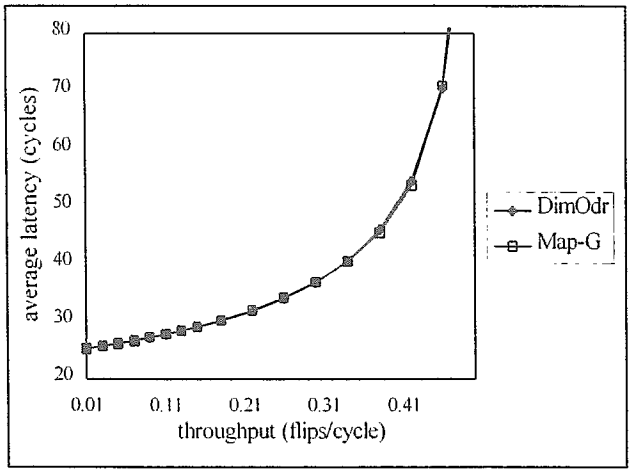

Figure 6. Uniform-random on a 4-ary 4-cube.

and an 8-ary 3-cube, respectively. These communications are linear-constant permutations. We may find linear mapping for them and see how the performance can be improved. The last communication pattern, uniform-random, is a general pattern representing the random access of a parallel computer. In our simulation for this communication, messages are generated randomly and independently by all the nodes, and the destinations of messages are evenly distributed across all the nodes. Fig. 6 shows the simulation result of uniform-random on a 4-ary 4-cube.

In these figures, Map-S denotes the simulation result using an optimal linear mapping for the specific communication pattern, which is proved to have minimum link contention in Theorem 3. On the other hand, Map-G denotes the simulation result using the linear mapping chosen for matrix-transpose and digit-reverse by the algorithm proposed in Theorem 4.

From Theorem 1, we can compute the degree of link contention for the first two communication patterns to be simulated. They are T=8 for matrix-transpose on a 16-ary 2-cube, T=4 for digit-reverse on an 8-ary 3-cube, and T=8 for matrix-transpose and digit-reverse on a 4-ary 4-cube. From simulation results, it can be observed that, for any of these communication patterns, the network throughput of the dimension ordered routing is always less than $1/T$. In other words, the maximum sustainable throughput that can be achieved is approximately inversely proportional to the degree of link contention. After linear mapping, the degree of link contention for those communications will become T=8 for matrix-transpose on a 16-ary 2-cube, T=4 for digit-reverse on an 8-ary 3-cube, and T=2 for matrix-transpose and digit-reverse on a 4-ary 4-cube. It can also be observed that the average latency becomes much lower after linear mapping, especially for Map-S and Map-G on a 4-ary 4-cube. Since the degree of link contention is 2 for both Map-S and Map-G on a 4-ary 4-cube, theoretically their throughput can approach 0.5. The actual value is somewhat smaller because of the queuing delay between messages generated by the same processor. As shown in Figs. 2 and 3, their values are about 0.4 instead of 0.5. Note that, for any given sustainable network throughput, the message latencies of Map-G and Map-S are much lower than that of the dimension ordered routing.

The simulation result for uniform-random is shown in Fig. 6. Obviously, it is not a linear-constant communication. Hence, we can not find an optimal linear mapping for it. In order to investigate the effects of linear mapping on uniform-random, Map-G is applied in the simulation. Each source-destination pair (x,y) in uniform-random now becomes $([Q_{\text{Map-G}}]x, [Q_{\text{Map-G}}]y)$ after Map-G is applied. In this way, the original communication can be correctly performed after processor mapping. Fig. 6 shows that the performance of Map-G is about the same as that of the dimension ordered routing. This means that linear mapping will not degrade the performance of uniform-random. Furthermore, its sustainable throughput can be more than 0.4, which is quite large comparing with the throughput of other communication patterns.

With these results, it is obvious that our approach may greatly reduce the message latency and, at the same time, significantly improve the throughput. Furthermore, no extra hardware supports and sophisticated routing strategies are needed. Only the dimension ordered wormhole routing is assumed in our approach. Therefore, it is of practicable use.

5. Conclusions

In this paper, we address the problem of minimizing the maximum number of paths contending for the same link when performing linear-constant communication on dimension ordered wormhole-routed k-ary n-cubes. A new approach called processor mapping is proposed to solve this problem. We have proved that, for a set of no more than k-1 linear-constant communications, there exists a linear mapping such that the new communications after processor mapping has minimum link contention. Simulation results clearly show significant performance improvement provided by the proposed approach when compared with the dimension ordered routing strategy. With these results, compiler techniques can be used to reduce the message latency without the need of extra hardware costs.

References:

[1] *Cray T3D System Architecture Overview Manual*, Cray Research, Inc..

[2] *nCUBE 2 Supercomputers Manual, NCUBE Company*, 1990.

[3] Seth Abraham and Krishnan Padmanabhan, "Performance of the Direct Binary *n*-Cube Network for Multiprocessors," *IEEE Transactions on Computers*, Vol. 38, No. 7, Jul. 1989, pp. 1000-1011.

[4] Francisco Arguello, Javier D. Bruguera, ans Emilio L. Zapata, "A Parallel Architecture for the Self-Sorting FFT Algorithm," *Journal of Parallel and Distributed Computing*, Vol. 31, No. 1, 1995, pp.88-97.

[5] Rajendra Boppana and C. S. Raghavendra, "Optimal Self-Routing of Linear-Complement Permutations in Hypercubes," *The Fifth Distributed Memory Computing Conference*, Apr. 1990, pp. 800-808, The University of South Carolina.

[6] Hsing-Lung Chen and Han-Shing Yihng, "Generalized Wormhole Routing Strategies in Hypercubes," *Journal of Information Science and Engineering*, Vol. 10, 1994, pp.387-341.

[7] Ge-Ming Chiu, Suresh Chalasani and C. S. Raghavendra, "Flexible, Fault-Tolerant Routing Criteria for Circuit-Switched Hypercubes," *Proc. IEEE 11th International Conference on Distributed Computing Systems*, 1991, pp.582-589.

[8] Willian J. Dally, "Performance Analysis of *k*-ary *n*-cube Interconnection Networks," *IEEE Transactions on Computers*, Vol. 39, No. 6, Jun. 1990, pp. 775-785.

[9] Willian J. Dally and Charles L. Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," *IEEE Transactions on Computers*, Vol. C-36, No. 5, May. 1987, pp. 547-553.

[10] Jeffrey T. Draper and Joydeep Ghosh, "A Comprehensive Analytical Model for Wormhole Routing in Multicomputer Systems," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 23, 1994, pp. 202-214.

[11] C. J. Glass and Lionel M. Ni, "The Turn Model for Adaptive Routing," *Journal of the Association for Computing Machinery*, Vol. 41, No. 5, Sep. 1994, pp. 874-902.

[12] Anshul Gupta and Vipin Kumar, "The Scalability of FFT on Parallel Computers," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 4, No. 8, Aug. 1993, pp. 922-932.

[13] Serge Lang, *Linear Algebra*, Springer-Verlag, 3rd Edition, 1987.

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

[15] Qiang Li, "Minimum Deadlock-Free Message Routing Restrictions in Binary Hypercubes," *Journal of Parallel and Distributed Computing*, Vol. 15, No 2, 1992, pp.153-159.

[16] Daniel H. Linder and Jim C. Harden, "An Adaptive and Fault Tolerant Wormhole Routing Strategy for *k*-ary *n*-cubes," *IEEE Transactions on Computers*, Vol. 40, No. 1, Jan. 1991, pp. 2-12.

[17] Hiroshi Masuyama, "Algorithms to realize an arbitrary BPC permutation in chordal ring networks and mesh connected networks," *IEICE Trans. Inf. Syst. (Japan)*, Vol. E77-D, No. 10, Oct. 1994, pp. 1118-1129.

[18] Hiroshi Masuyama, Yuichiro Morita and Etsuko Masuyama, "A realization of an arbitrary BPC permutation in hypercube connected computer networks," *IEICE Trans. Inf. Syst. (Japan)*, Vol. E78-D, No. 4, Apr. 1995, pp. 428-435.

[19] Robert J. McEliece, *Finite Fields for Computer Scientists and Engineers*, Kluwer Academic Publishers, 1987.

[20] Philip K. McKinley, Yih-jia Tsai, and David F. Robinson, "A Survey of Collective Communication in Wormhole-Routed Massively Parallel Computers", *IEEE Transactions on Computers*, Vol. 28, No. 12, Dec. 1995, pp. 39-50.

[21] David Nassimi and Sartaj Sahni, "An Optimal Routing Algorithm for Mesh-Connected Parallel Computers," *Journal of the Association for Computing Machinery*, Vol. 27, No. 1, Jan. 1980, pp. 6-29.

[22] David Nassimi and Sartaj Sahni, "Optimal BPC Permutations on a Cube Connected SIMD Computer," *IEEE Transactions on Computers*, Vol. C-31, No. 4, Apr. 1982, pp. 338-341.

[23] Lionel M. Ni and Philip K. McKinley, "A Survey of Wormhole Routing Techniques in Direct Networks," *IEEE Computer*, Vol. 26, Feb. 1993, pp. 62-76.

[24] J. Nussbaumer, *Fast Fourier Transform and Convolution Algorithms*, New York: Springer-Verlag, 1982.

[25] Youcef Saad and Martin H. Schultz, "Topological Properties of Hypercubes," *IEEE Transactions on Computers*, Vol. 37, No. 7, Jul. 1988, pp. 867-872.

[26] C. S. Yang, Y. M. Tsai, S. L. Chi, and Shepherd S. B. Shi, "Adaptive wormhole routing in *k*-ary *n*-cubes," *Parallel Computing*, Vol. 21, No. 12, 1995, pp. 1925-1943.