# The Postal Network: A Versatile Interconnection Topology[*]

Jie Wu
Dept. of Computer Sci. and Eng.
Florida Atlantic University
Boca Raton, FL 33431
jie@cse.fau.edu
http://www.cse.fau.edu/˜jie

Yuanyuan Yang
Dept. of Computer Science
University of Vermont
Burlington, VT 05405
yang@cs.uvm.edu
http://www.emba.uvm.edu/˜yang

**Abstract**

*The postal network is an interconnection network that possesses many desirable properties which are important in network design and applications. It includes hypercubes and Fibonacci cubes as its special cases. The postal network can also be considered as a flexible version of the hypercube which relaxes the restriction on the number of nodes and thus makes it possible to construct multicomputers with arbitrary sizes. Basically, the postal network forms a series (with series number $\lambda$) that is based on the sequence $N_\lambda(n) = N_\lambda(n-1) + N_\lambda(n-\lambda)$, where $n$ is the dimension and $N_\lambda(n)$ represents the number of nodes in an $n$-dimensional postal network in series $\lambda$. In this paper, we study topological properties of postal networks and relationships between different postal networks. One application of postal networks is also shown in implementing barrier synchronization using a special spanning tree called a postal tree.*

## 1 Introduction

The use of undirected graphs as interconnection topologies for large multicomputer systems has been an active research area in the past decades. The hypercube has been a popular topology because of its strong connectivity, regularity, symmetry, and ability to embed many other topologies. Unfortunately, the number of nodes $2^n$ in an $n$-dimensional hypercube ($n$-cube) grows rapidly as $n$ increases. This limits considerably the choice of the number of nodes in the graph. The Fibonacci cube ($FC$) proposed by Hsu [4] is a special subcube of a hypercube based on Fibonacci numbers. It has been shown that the Fibonacci cube can efficiently emulate many hypercube algorithms. Fibonacci cubes use fewer links than comparable hypercubes and their size does not increase as fast as hypercubes. The structural analysis of the Fibonacci cube has been extensively studied in [4], its applications in [2], and its extensions in [6].

[*]Research supported by the U.S. National Science Foundation under Grant No. OSR-9350540 and the U.S. Army Research Office under Grant No. DAAH04-96-1-0234.

In this paper, we propose a series of network topologies called *postal networks* with their names coming from the *postal model* [1] of communication. Postal networks include both hypercubes and Fibonacci cubes as special cases. Like Fibonacci cubes, postal networks can also be viewed as resulting from a hypercube after some nodes become faulty. Therefore, the postal network not only allows the construction of multicomputers of arbitrary sizes but also exposes the nature of hypercubes operating in a gracefully degraded mode. Basically, the $n$-dimensional postal network in series $\lambda$, $PN_\lambda(n)$, is based on the generalized Fibonacci sequence $N_\lambda(n) = N_\lambda(n-1) + N_\lambda(n-\lambda)$. The postal network series can also be considered as a flexible version of the hypercube which relaxes the restriction on the number of nodes and thus makes it possible to construct multicomputers with arbitrary sizes.

We show that the postal network series still maintains many desirable properties of hypercubes, such as existence of a Hamming distance path between any two nodes and a simple routing algorithm. Moreover, postal networks support efficient collective communication using the postal model [1] which can fine tune the communication structure based on network latency in the underlying system. Specifically, this model incorporates a latency parameter $\lambda$ measuring the inverse of the ratio between the time it takes an originator of a message to send it and the time that passes until the recipient of the message receives it. If $N_\lambda(n)$ represents the maximum number of nodes that can be reached in time $n$ on a one-port model exhibiting $\lambda$. Then the following equation holds:

$$N_\lambda(n) = \begin{cases} N_\lambda(n-1) + N_\lambda(n-\lambda), & \text{if } n \geq \lambda \\ 1, & \text{otherwise} \end{cases}$$

Therefore, if $\lambda$ in the postal network is selected based on the given latency parameter in the underlying network, efficient broadcast and gather operations can be carried out, i.e., a broadcast (gather) operation can be done in a minimum of steps.

The main features of the postal network can be summarized as follows:

- The series of postal networks allows more choices in constructing systems of different sizes.
- The series contains both hypercubes and Fibonacci cubes as its special cases.

- The series number $\lambda$ can be carefully selected to match the latency parameter in the underlying communication network to support efficient collective communication.

Our study focuses on topological properties and communication aspects of the postal network. Relationships between different postal networks are also studied. The purpose of this study is not just to propose a "new" interconnection network, but to extend the existing ones such as hypercubes and Fibonacci cubes. We try to gain some insights on these popular networks, operated under certain relaxed conditions and/or degraded modes, by studying their topological properties, routing capability, and the ability of simulating other structures through embedding.

## 2 Postal Networks

Normally, a graph model is used to represent a point-to-point multicomputer topology. We use graph $G = (V, E)$ to represent an interconnection network, where $V$ is a vertex set with each element representing a processor (also called a node) and $E$ is an edge set with each element corresponding to a communication link connecting two nodes. Let $b_{(m)}$ represent $m$ consecutive bits of $b$; for example, $0_{(4)} = 0000$, and symbol $||$ denote a concatenation operation; for example, $01||\{0,1\} = \{010, 011\}$ and $0_{(2)}1||\{01, 10\} = \{00101, 00110\}$.

**Definition 1**: *Assume that graphs $PN_\lambda(n) = (V_\lambda(n), E_\lambda(n))$, $PN_\lambda(n-1) = (V_\lambda(n-1), E_\lambda(n-1))$, and $PN_\lambda(n-\lambda) = (V_\lambda(n-\lambda), E_\lambda(n-\lambda))$. Then $V_\lambda(n) = 0||V_\lambda(n-1) \cup 10_{(\lambda-1)}||V_\lambda(n-\lambda)$ for $n > \lambda$. As initial conditions for recursion, $V_\lambda(n) = \{0_{(n)}, 0_{(n-1)}1, ..., 010_{(n-2)}, 10_{(n-1)}\}$, $1 \leq n \leq \lambda$. Two nodes in $PN_\lambda(n)$ are connected by an edge in $E_\lambda(n)$ if and only if their labels differ in exactly one bit position.*

A $PN_\lambda(n)$ is called a *postal network* with dimension $n$ and series number $\lambda$. The number of bits in a node address is the same as its dimension. Figure 1 shows examples of $PN_3(n)$ for $n = 1, 2, 3, 4, 5$. A $PN_3$ of dimension $n$ consists of one $PN_3$ of dimension $n-1$ and one $PN_3$ of dimension $n-3$. Figure 2 shows examples of $PN_4(n)$ for $n = 1, 2, 3, 4, 5, 6$.

**Theorem 1**: *A $PN_1(n)$ is an $n$-cube $Q(n)$ and a $PN_2(n)$ is an $n$-dimensional Fibonacci cube $FC(n)$.*

*Proof*: When $\lambda = 1$, $V_1(n) = 0||V_1(n-1) \cup 1||V_1(n-1)$. Therefore, $PN_1(n)$ matches exactly the definition of the $n$-cube $Q(n)$. Similarly, when $\lambda = 2$, $V_2(n) = 0||V_2(n-1) \cup 10||V_2(n-2)$. Recall that the Fibonacci cube is defined as follows [4]: Assume that graphs $FC(n) = (V(n), E(n))$, $FC(n-1) = (V(n-1), E(n-1))$, and $FC(n-2) = (V(n-2), E(n-2))$. Then $V(n) = 0||V(n-1) \cup 10||V(n-2)$. Two nodes in $FC(n)$ are connected by an edge in $E(n)$ if and only if their labels differ in exactly one bit position. As initial conditions for recursion, $V(2) = \{\}$, $V(3) = \{0, 1\}$. Therefore, $PN_2(n)$ also matches the definition of the $n$-dimensional Fibonacci cube $FC(n)$. ■

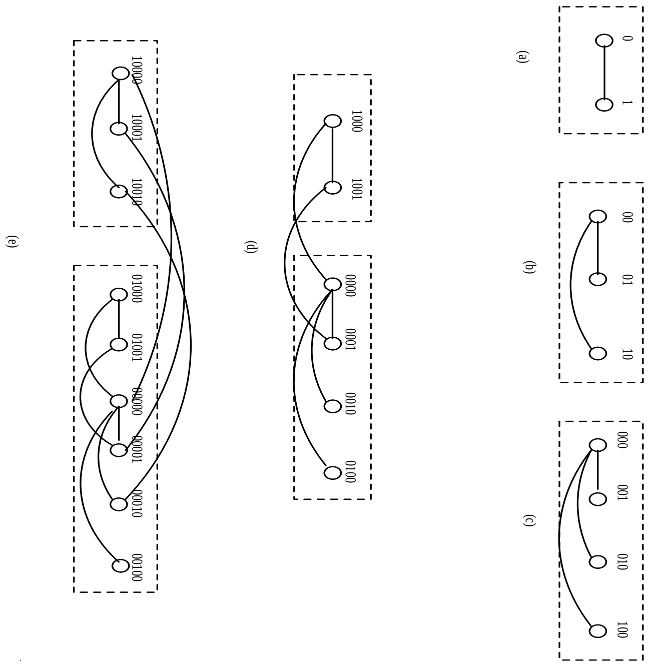

Figure 1: Postal network $PN_3$: (a) $n = 1$, (b) $n = 2$, (c) $n = 3$, (d) $n = 4$, (e) $n = 5$.

Despite its asymmetric structure, the postal network still maintains many desirable properties from the hypercube network. Based on the definition of the postal network, the vertex set $V_\lambda(k)$ of $PN_\lambda(k)$ can be partitioned into $10_{(\lambda-1)}||V_\lambda(k-\lambda)$ and $0||V_\lambda(k-1)$. The following lemma shows the relationship between these two vertex sets.

**Lemma**: *For each node in $10_{(\lambda-1)}||V_\lambda(k-\lambda)$ there is exactly one neighbor in $0||V_\lambda(k-1)$, i.e., the addresses of these two nodes differ in exactly one bit.*

*Proof*: Randomly pick a node in $10_{(\lambda-1)}||V_\lambda(k-\lambda)$. This node comes from a node in $V_\lambda(k-\lambda)$ with $10_{(\lambda-1)}$ attached in front. Based on the recursive definition of the postal network, $0_{(\lambda-1)}||V_\lambda(k-\lambda)$ is a subset of $V_\lambda(k-1)$ and it always appears as the first term of the recursive definition of $V_\lambda(k-1)$ until $V_\lambda(k-1)$ is resolved into $V_\lambda(k-\lambda)$. Therefore, there is at least one node in $0||V_\lambda(k-1)$ that is the neighbor of the selected node in $10_{(\lambda-1)}||V_\lambda(k-\lambda)$. In addition, each node in $10_{(\lambda-1)}||V_\lambda(k-\lambda)$ has exactly one neighbor in $0||V_\lambda(k-1)$, because the address of each node in $10_{(\lambda-1)}||V_\lambda(k-\lambda)$ starts with 1 while the one in $0||V_\lambda(k-1)$ starts with 0. ■

**Theorem 2**: *There exists a Hamming distance path for any two nodes in $PN_\lambda(n)$.*

*Proof*: We prove this theorem by induction on $n$ for any $\lambda$. We first show that this theorem holds for $n = 1, 2, ..., \lambda$. Since each of these networks is a two-level tree that con-

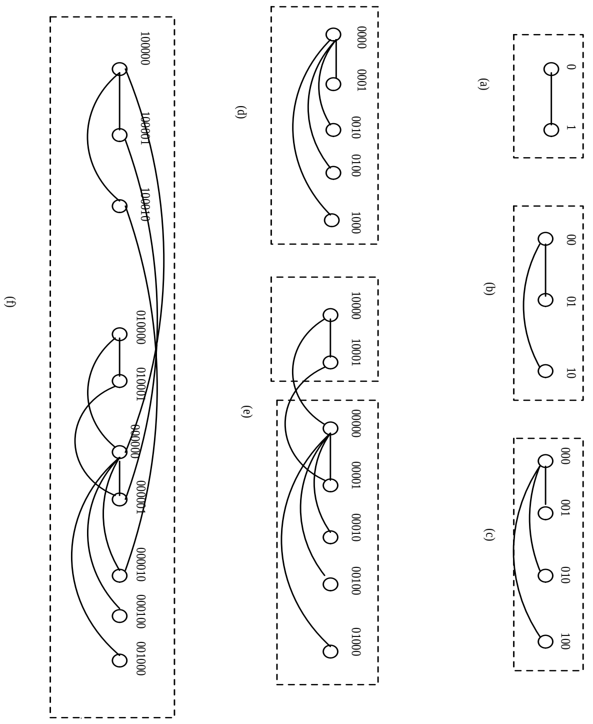

Figure 2: Postal network $PN_4$: (a) $n = 1$, (b) $n = 2$, (c) $n = 3$, (d) $n = 4$, (e) $n = 5$, (f) $n = 6$.

tains $n$ nodes: $0_{(n)}, 0_{(n-1)}1, ..., 010_{(n-2)}, 10_{(n-1)}$. Obviously, the root node $0_{(n)}$ directly connects to all the other nodes. Since any two leaf nodes differ in two bits, they are two Hamming distance apart and a Hamming distance path exists between any two leaf nodes, since each leaf node can reach another leaf node via the root node.

Assume that this theorem holds for all $n < k$ (for $k > \lambda$, since we have proved the theorem for $n = 1, 2, ..., \lambda$). When $n = k$, $V_\lambda(k) = 0||V_\lambda(k-1) \cup 10_{(\lambda-1)}||V_\lambda(k-\lambda)$. Therefore, nodes in $V_\lambda(k)$ can be partitioned into $0||V_\lambda(k-1)$ and $10_{(\lambda-1)}||V_\lambda(k-\lambda)$.

We randomly select two nodes, if both nodes belong to $0||V_\lambda(k-1)$ (or $10_{(\lambda-1)}||V_\lambda(k-\lambda)$) this theorem holds based on the induction assumption. We only need to consider cases when one node is in $0||V_\lambda(k-1)$ and the other is in $10_{(\lambda-1)}||V_\lambda(k-\lambda)$. Based on Lemma, for each node in $10_{(\lambda-1)}||V_\lambda(k-\lambda)$ there is exactly one neighbor in $0||V_\lambda(k-1)$. The Hamming distance path can be constructed recursively the following two cases:

(1) If the source is in $10_{(\lambda-1)}||V_\lambda(k-\lambda)$, the first link in the path should connect the source to its neighbor in $0||V_\lambda(k-1)$. The rest of the links in the path can be determined recursively in $0||V_\lambda(k-1)$.

(2) If the source is in $0||V_\lambda(k-1)$, we first determine a Hamming distance path within $0||V_\lambda(k-1)$ that connects the source to the neighbor of the destination in $0||V_\lambda(k-1)$. The last link connects the neighbor of the destination in $0||V_\lambda(k-1)$ to the destination node which is in $10_{(\lambda-1)}||V_\lambda(k-\lambda)$. ■

**Theorem 3**: *The diameter of $PN_\lambda(n)$ is the following:*

$$d(PN_\lambda(n)) = \begin{cases} n & \lambda = 1 \\ 2\lceil \frac{n}{\lambda} \rceil - 1 & \lambda \neq 1 \text{ and } (n-1 \bmod \lambda) = 0 \\ 2\lceil \frac{n}{\lambda} \rceil & \lambda \neq 1 \text{ and } (n-1 \bmod \lambda) \neq 0 \end{cases}$$

*Proof:* When $\lambda = 1$, $PN_\lambda(n)$ is an $n$-cube $Q(n)$ which has a diameter of $n$. We only need to consider cases when $\lambda \neq 1$, i.e., there are no adjacent 1 bits in a node address. Based on the recursive definition of the postal network, the maximum number of 1 bits in a node address increases by one when $n$ increases for $\lambda$, i.e., the distance between two 1 bits is at least $\lambda$ and there are at most $\lceil \frac{n}{\lambda} \rceil$ 1 bits in a node address. In the extreme case, the positions of 1 bits in two nodes are all different. When $(n-1 \bmod \lambda) \neq 0$, it is possible that both nodes have $\lceil \frac{n}{\lambda} \rceil$ 1 bits. When $(n-1 \bmod \lambda) = 0$, only one node may have $\lceil \frac{n}{\lambda} \rceil$ bits and the other node may have at most $\lceil \frac{n}{\lambda} \rceil - 1$ 1 bits at different bit positions. The above two situations correspond to two nodes that have the longest Hamming distance in the network, i.e., their distance corresponds to the diameter of the network. ■

Based on the above result, the diameter of a postal network is a fraction of its dimension $n$. However, postal networks with a large $\lambda$ contain fewer nodes than the ones with a small $\lambda$ under the same dimension condition. The number of nodes and links in a given postal network is given in the following theorem.

**Theorem 4**: *Let $N_\lambda(n)$ and $L_\lambda(n)$ be the number of nodes and links in $PN_\lambda(n)$, respectively, then*

$$N_\lambda(n) = \begin{cases} n+1 & n \leq \lambda \\ N_\lambda(n-1) + N_\lambda(n-\lambda) & n > \lambda \end{cases}$$

*and*

$$L_\lambda(n) = \begin{cases} n & n \leq \lambda \\ L_\lambda(n-1) + L_\lambda(n-\lambda) + N_\lambda(n-\lambda) & n > \lambda \end{cases}$$

*Proof*: The expression for $N_\lambda(n)$ is straightforward from the definition of the postal network. The number of links $L_\lambda(n)$ in $PN_\lambda(n)$ is the summation of the number of links $L_\lambda(n-1)$ in $PN_\lambda(n-1)$ and the number of links $L_\lambda(n-\lambda)$ in $PN_\lambda(n-\lambda)$. In addition, we should include links that connect $PN_\lambda(n-1)$ to $PN_\lambda(n-\lambda)$. Based on Lemma, we know that for each node in $PN_\lambda(n-\lambda)$ there exists exactly one neighbor in $PN_\lambda(n-1)$. That is, exactly $N_\lambda(n-\lambda)$ links exist, each of which connects one node in $PN_\lambda(n-1)$ to one node in $PN_\lambda(n-\lambda)$. ■

Note that $N_\lambda(n)$ belongs to a recurrence relation of form

$$\begin{aligned} f(n) &= C_0N(n) + C_1N(n-1) + C_2N(n-2) + \cdots \\ &\quad + C_\lambda N(n-\lambda) \end{aligned} \qquad (1)$$

where $C_i$'s are constants. It is also called a *linear recurrence relation with constant coefficients*. It is also known

Table 1: Number of nodes in $Q = PN_1$, $FC = PN_2$, $PN_3$, and $PN_4$.

| $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| $N_1(k)$ | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 |
| $N_2(k)$ | 2 | 3 | 5 | 8 | 13 | 21 | 34 | 55 | 89 |
| $N_3(k)$ | 2 | 3 | 4 | 6 | 9 | 13 | 19 | 28 | 41 |
| $N_4(k)$ | 2 | 3 | 4 | 5 | 7 | 10 | 14 | 19 | 26 |

Table 2: Number of links in $Q = PN_1$, $FC = PN_2$, $PN_3$, and $PN_4$.

| $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| $L_1(k)$ | 1 | 4 | 12 | 32 | 80 | 192 | 448 | 1024 | 2304 |
| $L_2(k)$ | 1 | 2 | 5 | 10 | 20 | 38 | 71 | 130 | 235 |
| $L_3(k)$ | 1 | 2 | 3 | 6 | 11 | 18 | 30 | 50 | 81 |
| $L_4(k)$ | 1 | 2 | 3 | 4 | 7 | 12 | 19 | 28 | 42 |

as a $\lambda$th-*order recurrence relation*. Assume that $r_i$'s are distinct roots of the *characteristic equation*

$$C_0N^{\lambda} + C_1N^{\lambda-1} + C_2N^{\lambda-2} + \cdots + C_{\lambda} = 0 \qquad (2)$$

Then

$$N(n) = A_1r_1^n + A_2r_2^n + A_3r_3^n + \cdots + A_{\lambda}r_{\lambda}^n \qquad (3)$$

where $A_1, A_2, A_3, ..., A_{\lambda}$ are constants which are to be determined by the boundary conditions, i.e., the known values of $N(i)$'s. For example, when $\lambda = 2$,

$$N_2(n) = N_2(n-1) + N_2(n-2)$$

The corresponding characteristic equation is $N^2 - N - 1 = 0$ which has the two distinct roots $r_1 = \frac{1+\sqrt{5}}{2}$, $r_2 = \frac{1-\sqrt{5}}{2}$. It follows that

$$N_2(n) = A_1\left(\frac{1+\sqrt{5}}{2}\right)^n + A_2\left(\frac{1-\sqrt{5}}{2}\right)^n$$

is the solution, where $A_1$ and $A_2$ are to be determined from the boundary conditions $N_2(1) = 1$ and $N_2(2) = 2$. Note that if the characteristic equation (2) has multiple roots (i.e., not all roots are distinct), $N(n)$ is calculated differently. See [3] for details.

There is no general procedure for determining the solution of a difference equation, especially for a high-order equation. That is, for a large value of $\lambda$, a closed form expression for either $N_{\lambda}(n)$ or $L_{\lambda}(n)$ is unlikely. Tables 1 and 2 show $N_{\lambda}(n)$ and $L_{\lambda}(n)$ for different $\lambda$'s and $n$'s. From these tables, we can see that when series number $\lambda$ increases, the increase rates for the number of nodes and links both reduce. To estimate $N_{\lambda}(n)$, we have $N_{\lambda}(n) = N_{\lambda}(n-1) + N_{\lambda}(n-\lambda) \leq N_{\lambda}(n-1) + N_{\lambda}(n-1) \leq 2^n$. Also, we have $N_{\lambda}(n) = N_{\lambda}(n-1) + N_{\lambda}(n-\lambda) \geq N_{\lambda}(n-\lambda) + N_{\lambda}(n-\lambda) \geq 2^{\lfloor \frac{n}{\lambda} \rfloor}$. Combining the above two conditions, we have $2^{\lfloor \frac{n}{\lambda} \rfloor} \leq N_{\lambda}(n) \leq 2^n$

Furthermore, we can obtain a tighter upper bound on $N_{\lambda}(n)$ when $\lambda > 1$ as given in the following theorem. The theorem indicates that as $\lambda$ gets larger, the number of nodes in a $PN_{\lambda}(n)$, $N_{\lambda}(n) = O\left(\left(1 + \frac{1}{\sqrt{\lambda-1}}\right)^{n-1}\right)$, is much smaller than that of a hypercube, $N_1(n) = 2^n$.

**Theorem 5:** *When* $\lambda > 1$,

$$N_{\lambda}(n) \leq (\lambda+1)\left(1 + \frac{1}{\sqrt{\lambda-1}}\right)^{n-1}$$

*Proof*: We prove this theorem by induction on $n$ for any given $\lambda > 1$. First, for $1 \leq n \leq \lambda$, we have

$$N_{\lambda}(n) = n + 1 \leq \lambda + 1 \leq (\lambda+1)\left(1 + \frac{1}{\sqrt{\lambda-1}}\right)^{n-1}$$

Now, suppose the upper bound holds for $n = k-1, k-2, \ldots, k-\lambda\ (k > \lambda)$. We prove that it also holds for $n = k$. Note that

$$\begin{aligned} N_{\lambda}(k) &= N_{\lambda}(k-1) + N_{\lambda}(k-\lambda) \\ &\leq (\lambda+1)\left[\left(1 + \tfrac{1}{\sqrt{\lambda-1}}\right)^{k-2} + \left(1 + \tfrac{1}{\sqrt{\lambda-1}}\right)^{k-\lambda-1}\right] \\ &= (\lambda+1)\left(1 + \tfrac{1}{\sqrt{\lambda-1}}\right)^{k-\lambda-1}\left[\left(1 + \tfrac{1}{\sqrt{\lambda-1}}\right)^{\lambda-1} + 1\right] \end{aligned}$$

only need to show that

$$\left(1 + \frac{1}{\sqrt{\lambda-1}}\right)^{\lambda-1} + 1 \leq \left(1 + \frac{1}{\sqrt{\lambda-1}}\right)^{\lambda}$$

Or equivalently,

$$\left(1 + \frac{1}{\sqrt{\lambda-1}}\right)^{\lambda} - \left(1 + \frac{1}{\sqrt{\lambda-1}}\right)^{\lambda-1} \geq 1 \qquad (4)$$

In fact, from the left hand side of (4), we have

$$\left(1 + \frac{1}{\sqrt{\lambda-1}}\right)^{\lambda-1}\left(\frac{1}{\sqrt{\lambda-1}}\right) \geq \left(1 + \frac{\lambda-1}{\sqrt{\lambda-1}}\right)\frac{1}{\sqrt{\lambda-1}} \geq 1$$

Thus, Theorem 5 holds. ■

Similarly, we can estimate $L_{\lambda}(n)$. The following theorem gives the relationship between $L_{\lambda}(n)$ and $N_{\lambda}(n)$ and an upper on $L_{\lambda}(n)$.

**Theorem 6:** *When* $\lambda > 1$,

$$L_{\lambda}(n) < \left(\frac{n-2}{\lambda+1} + 1\right)N_{\lambda}(n) \leq (n + \lambda - 1)\left(1 + \frac{1}{\sqrt{\lambda-1}}\right)^{n-1}$$

*Proof*: We prove this theorem by induction on $n$ for any given $n > 1$. First, when $n = 1$, $L_{\lambda}(1) = 1$ and $N_{\lambda}(1) =$

2. Clearly, $L_\lambda(n) < (\frac{n-2}{\lambda+1}+1)N_\lambda(n)$ given $\lambda > 1$. When $2 \le n \le \lambda$, we have

$$L_\lambda(n) = N_\lambda(n) - 1 < \left(\frac{n-2}{\lambda+1}+1\right) N_\lambda(n)$$

Now, suppose the upper bound holds for $n = k-1, k-2, \ldots, k-\lambda$ $(k > \lambda)$. We will prove it also holds for $n = k$. Note that

$$\begin{aligned}
L_\lambda(k) &= L_\lambda(k-1) + L_\lambda(k-\lambda) + N_\lambda(k-\lambda) \\
&< \left[\frac{(k-1)-2}{\lambda+1}+1\right] N_\lambda(k-1) + \\
&\quad \left[\frac{(k-\lambda)-2}{\lambda+1}+1\right] N_\lambda(k-\lambda) + N_\lambda(k-\lambda) \\
&= \left(\frac{k-2}{\lambda+1}+1\right)[N_\lambda(k-1)+N_\lambda(k-\lambda)] - \\
&\quad \frac{1}{\lambda+1}[N_\lambda(k-1)+N_\lambda(k-\lambda)] \\
&< \left(\frac{k-2}{\lambda+1}+1\right)[N_\lambda(k-1)+N_\lambda(k-\lambda)] \\
&= \left(\frac{k-2}{\lambda+1}+1\right) N_\lambda(k)
\end{aligned}$$

Thus, we have

$$L_\lambda(n) < \left(\frac{n-2}{\lambda+1}+1\right) N_\lambda(n)$$

Moreover, by Theorem 5, we obtain an upper bound on $L_\lambda(n)$:

$$\begin{aligned}
L_\lambda(n) &< \left(\frac{n-2}{\lambda+1}+1\right)(\lambda+1)\left(1+\frac{1}{\sqrt{\lambda-1}}\right)^{n-1} \\
&= (n+\lambda-1)\left(1+\frac{1}{\sqrt{\lambda-1}}\right)^{n-1}
\end{aligned}$$

■

## 3 Routing

In this section, we study routing algorithms for the postal network. Efficient interprocessor communication is a key to the performance of a point-to-point multicomputer system. We consider here *unicasting*, which is a one-to-one communication between a source and a destination. Although the postal network is asymmetric, a simple routing algorithm can still be constructed based on the following result that determines whether a given bit sequence belongs to a node address in a postal network.

**Theorem 7:** *An $n$-bit sequence is the address of a node in $PN_\lambda(n)$ if and only if any two 1 bits (if any) are separated by at least $\lambda$ bit positions.*

*Proof*: Let's consider a systematic way of generating all the possible $n$-bit sequences such that any two 1 bits (if any) are separated by at least $\lambda$ bit positions. Assume that we have constructed $S_\lambda(i)$'s for all $i < n$, where $S_\lambda(i)$ stands for all the possible $i$-bit sequences such that any two 1 bits (if any) are separated by at least $\lambda$ bit positions. Let's consider the leftmost bit (the 1st bit) of the nodes in $S_\lambda(n)$: If it is 0, then the number of different arrangements for the rest of $n-1$ bits should all be in $S_\lambda(n-1)$. If it is 1, then based on the constraint that any two 1 bits must be separated by at least $\lambda$ bit positions, the next $\lambda - 1$ bits must be all 0's. The number of different arrangements for the rest of $n-\lambda$ bits should all be in $S_\lambda(n-\lambda)$. Based on the above observation, it is clear that $PN_\lambda(n)$ and $S_\lambda(n)$ are the same. ■

The above result provides a simple way of defining a postal network and more importantly it offers a simple way of generating all the nodes in a postal network. For example, nodes in $PN_4(6)$ are 000000, 000001, 000010, 000100, 001000, 010000, 100000, 010001, 100001, 100010. Among these nodes, the node addresses have two 1 bits or less that are separated by at least 4 bit positions. We use $distance_1(node)$ to represent minimum distance between 1 bits in a given node address. When there is at most one 1 bit in a node address, the $distance_1$ of the corresponding node is $\infty$. For example, $distance_1(000100) = \infty$ and $distance_1(010001) = 4$.

We consider here adaptive and minimal routing. An adaptive routing algorithm allows all messages to use any minimal paths. The challenge is to exploit all the possible routes while still keeping routing distance minimum. Since the postal network can be considered as an incomplete hypercube with several missing nodes, the traditional dimension-ordered routing is no longer applicable here. For example, consider two nodes 01000 (the source $s$) and 10010 (the destination $d$), the exclusive-or of their addresses is 01000 $\oplus$ 10010 = 11010. If the dimensions are resolved following an increasing order of dimensions: 1, 2 and 4, an illegal intermediate node 11000 (with two neighboring 1 bits) will be generated in the corresponding path: 01000 $\rightarrow$ 11000 $\rightarrow$ 10000 $\rightarrow$ 10010.

In order to avoid generating illegal intermediate nodes, we should ensure that each intermediate node is legal. That is, the distance between two 1 bits (if any) in the node address should be at least $\lambda$. To make the routing algorithm adaptive, no additional constraint is added, i.e., a dimension can be randomly selected as long as it meets the above requirement.

Consider a unicasting from $s$ to $d$ in $PN_\lambda(n)$. Let $s^i$ denote complementing the $i$th bit of $s$, for example $10010^2 = 11010$, and $r(i)$ denote the $i$th bit of $r$. The adaptive and minimal routing algorithm for the postal network is the following:

```
For source node $s$ with message $m$:

1. $r := s \oplus d$; /* calculate relative address $r$ */
2. randomly select $i$ such that $r(i) = 1$ and
   $distance_1(s^i) \ge \lambda$; /* select a neighbor */
3. send $(m, r^i)$ to $s^i$.
   /* send message $m$ together with the updated
```

```
    relative address to the selected neighbor */
```

For all intermediate nodes $t$ (including destination $d$):

```
1. receive $(m, r)$; /* receive message $m$ together
   with relative address $r$ */
2. if $r = 0$ then node $t$ is the destination and stop;
3. randomly select $i$ such that $r(i) = 1$ and
   $distance_1(t^i) \geq \lambda$;
4. send $(m, r^i)$ to $t^i$.
```

To carry out step 3 of the above procedure, we first pick up a neighbor. If the address of this neighbor meets the condition in Theorem 7 (this can be done in constant time), it is done; otherwise, another neighbor is selected. In the worst case, all the neighbors, except the last one, fail the condition. That is, if the current node is $k$ distance away from the destination, it may need $k - 1$ selection steps. Thus, step 3 may need $O(n)$ time in the worst case.

Consider $(s, d) = (100010, 000001)$ with $r = 100011$ in $PN_4(6)$. At the first step, two legal neighbors (of $s$) are 100000 and 000010. 100011 is an illegal neighbor, since its $distance_1(10001) = 1$ which is less than $\lambda = 4$. During the second step, at node 100000, there are two choices of the next intermediate node: 100001 and 000000; at node 000010, there is only one choice which is node 000000. Therefore, three minimal routing paths can be generated:

$$100010 \rightarrow 100000 \rightarrow 100001 \rightarrow 000001$$

$$100010 \rightarrow 100000 \rightarrow 000000 \rightarrow 000001$$

$$100010 \rightarrow 000010 \rightarrow 000000 \rightarrow 000001$$

Note that if $distance_1(s \oplus d = r) \geq \lambda$, there is no constraint on selecting intermediate nodes. Routing will be the same as in a regular hypercube. For example, consider $(s, d) = (10000, 00001)$ in $PN_3(5)$, i.e., $\lambda = 3$. Clearly, $distance_1(s \oplus d = 10001) \geq 3$. The e-cube routing in hypercubes can be applied for this case. In fact, $s$ and $d$ are contained in 2-cube $*000*$, where $*$ is a don't care. $*000*$ contains four nodes 00000, 00001, 10000, and 10001. The following theorem shows a general case.

**Theorem 8**: *In a $PN_\lambda(n)$, if $distance_1(s \oplus d) \geq \lambda$ then $s$ and $d$ are two nodes in a $k$-cube, where $k \geq H(s, d)$.*

The above theorem can be easily derived from the above observation. To determine the smallest cube that contains $s$ and $d$, assuming that $distance_1(s \oplus d) \geq \lambda$, one simple approach is to replace all the 1 bits in $s \oplus d$ by $*$ and the remaining bits are replaced by the corresponding bits in $s$ (or $d$). For example, $s \oplus d = 11011 \oplus 01001 = 10010$, the smallest cube that contains $s$ and $d$ is $*10*1$.

The minimal routing approach can be easily extended to the one for non-minimal routing. For example in non-minimal routing, we can separate 1's (called *preferred dimensions*) from 0's (called *spare dimensions*) in $r = s \oplus d$. In minimal routing, only preferred dimensions are resolved by changing each of them to 0. In non-minimal routing, spare dimensions can also be used which are changed from 0's and 1's, but to reach the destination, these 1's still need to be changed back to 0's. At each step, the $distance_1$ condition still needs to be enforced to ensure that each intermediate node is legal. Consider a unicasting from $s =$100000 to $d =$000000 in $PN_4$, $s \oplus d = 100000$, a non-minimal routing that uses preferred dimension 1 and spare dimension 6 generates the following path $100000 \rightarrow 100001 \rightarrow 000001 \rightarrow 000000$.

## 4 Embedding

In this section, we study relationships between regular hypercubes, Fibonacci cubes, and postal networks. Efficient embedding of a guest network $G$ into a host network $H$ is important in parallel/distributed processing, especially for a newly proposed network used as a host network. Not only do embedding results demonstrate computational equivalence (or near-equivalence) between networks of different topology, but efficient embeddings lead to efficient simulations of algorithms originally designed for $G$ on host $H$.

Based on the definition of the postal network series, link connections follow the same rule: Two nodes are connected if and only if their addresses differ in exactly one bit position. Therefore, to show that one postal network contains another postal network as its subgraph, we only need to show that the vertex set of the former contains the vertex set of the latter. Specifically, a graph $H$ contains a graph $G$ if one of the following two conditions holds: (1) $V(H)$ contains $V(G)$. (2) $V'(H)$ contains $V(G)$, where $V'(H)$ is derived from $V(H)$ by removing certain bit positions of all the nodes in $V(H)$.

For example, $PN_2(4)$ contains $PN_3(4)$ based on Condition (1), because $V(PN_2(4)) = \{0000, 0001, 0010, 0100, 0101, 1000, 1001, 1010\}$ contains $V(PN_3(4)) = \{0000, 0001, 0010, 0100, 1000, 1001\}$. Also, $PN_2(4)$ contains $PN_2(3)$ based on Condition (2), because $V(PN_2(3)) = \{000, 001, 010, 100, 101\}$ is derived by removing either the 1st or 4th bit of all the nodes in $V(PN_2(4))$.

The following theorem shows relationships between postal networks within the same series.

**Theorem 9:** *$PN_{\lambda_1}(n_1)$ contains $PN_{\lambda_1}(n_2)$ as its subgraph if and only if $n_1 > n_2$.*

*Proof*: Based on the definition of the postal network, $PN_{\lambda_1}(n_1)$ contains $PN_{\lambda-1}(n_1 - 1)$. Clearly $PN_{\lambda_1}(n_1)$ also contains $PN_{\lambda-1}(n_2)$, where $n_1 > n_2$. ■

The following theorem shows that for a given dimension $n_1$, a network with a small series number (with a small $\lambda$ value) contains a network with a large series number.

**Theorem 10:** *$PN_{\lambda_1}(n_1)$ contains $PN_{\lambda_2}(n_1)$ as its subgraph if and only if $\lambda_1 < \lambda_2$.*

*Proof*: We prove this theorem by induction. When $n_1 \leq \lambda_1$, $PN_{\lambda_1}(n_1) = PN_{\lambda_2}(n_1)$. When $\lambda_1 < n_1 \leq \lambda_2$, we can easily verify that $PN_{\lambda_1}(n_1)$ contains $PN_{\lambda_2}(n_1)$. When $n_1 > \lambda_2$, based on the recursive definition of both series and the induction assumption, $PN_{\lambda_1}(n_1)$ contains $PN_{\lambda_2}(n_1)$. ■

The following theorem shows that under certain conditions a network with a large series number contains a network with a small series number.

**Theorem 11:** *Given $PN_{\lambda_1}(n_1)$, $PN_{\lambda_2}(n_2)$, and $\lambda_1 > \lambda_2$. If $\lceil \frac{n_1}{\lambda_1} \rceil > \lceil \frac{n_2}{\lambda_2} \rceil$ then $PN_{\lambda_1}(n_1)$ contains $PN_{\lambda_2}(n_2)$ as its subgraph. If $\lceil \frac{n_1}{\lambda_2} \rceil = \lceil \frac{n_2}{\lambda_2} \rceil$ and $n_1 - c\lambda_1 \geq n_2 - c\lambda_2$ then $PN_{\lambda_1}(n_1)$ contains $PN_{\lambda_2}(n_2)$ as its subgraph, where $c$ is a constant.*

*Proof*: We reorganize postal networks in series $\lambda$ in groups, each of which contains $\lambda$ consecutive networks, i.e., the $n$th network in series $\lambda$ is in group $\lceil \frac{n}{\lambda} \rceil$. For example, in series $PN_3(n)$, $\{PN_3(1), PN_3(2), PN_3(3)\}$ forms group 1, $\{PN_3(4), PN_3(5), PN_3(6)\}$ forms group 2, and so on. Therefore, each network in the series has a group number $\lceil \frac{n}{\lambda} \rceil$ and a number within the group $n - (\lceil \frac{n}{\lambda} \rceil - 1)\lambda$. The theorem is reduced to prove the following: Consider two networks from different series with the same group number and the same number within the group, the one with a larger series number contains the one with a smaller series number. The above fact can be proved by induction from group to group using the following fact: The $i$th network in group $j$ ($j > 1$) is constructed from the network that just precedes it and the $i$th network in group $j - 1$. ■

**Corollary**: *$PN_\lambda(n_1)$ contains the $n_2$-cube $Q(n_2)$ as its subgraph if and only if $n_1 + \lceil \frac{n_1}{\lambda_1} \rceil (1 - \lambda_1) \geq n_2$.*

This corollary is derived directly from Theorem 11 and can be used to derive the largest hypercube as a subgraph of a given postal network. For example, for $PN_2(9)$ the largest hypercube is an $n_2$-cube such that $9 + \lceil \frac{9}{2} \rceil (1-2) \geq n_2$, i.e., $n_2 = 4$.

We have the following simple process to determine all the largest subcubes in $PN_\lambda(n_1)$: We start with an $n_1$-bit of 0's and then replace $n_2$ bits (which is determined from the above Corollary) of 0's by $*$'s such that any two $*$'s are separated by at least $\lambda$. For example, given $PN_4(6)$, based on the above Corollary, the largest subcube is a 2-cube. All the possible 2-cubes in $PN_4(6)$ are $*0000*$, $0*000*$, and $*000*0$.

## 5 Application: Barrier Synchronization Using Postal Trees

In this section, we first study the *postal tree*, $PT_\lambda(n)$, which is a special spanning tree of the postal network $PN_\lambda(n)$. We then look at one of its applications in implementing barrier synchronization, an important type of collective communication in a multicomputer system.

Many numerical problems can be solved using iterative algorithms that successively compute better approximations to an answer, terminating when either the final answer has been computed or the final answer has converged. These algorithms normally require all the iterative processes to be synchronized at the end of each iteration. More specifically, these processes, $process(i)$, can be described by the following algorithm:

```
do not_converged →
    code to implement process $i$
    barrier (wait for all $n$ processes to complete)
od
```

In the above algorithm, **barrier** represents a barrier synchronization point which waits for all $n$ processes to complete. This type of synchronization is called *barrier synchronization* [5] because the delay point at the end of each iteration represents a barrier that all processes have to arrive at before any of them is allowed to pass. There are many ways of implementing barriers, among them *tree barrier* [7] is the widely used one. In this approach, two phases are used. In the *reduction* phase, all participating processes engage in a reduction operation by sending and/or receiving synchronization messages following a tree structure where the root node eventually receives the reduced message and decides that all the processes have arrived at the barrier. In the next phase which is called distribution phase, the root node broadcasts a synchronization message following the same tree to inform all the processes to proceed. Normally, the reduction phase is carried out by a collective communication called gather and the distribution phase is implemented by another collective communication called broadcast. Both of which use a spanning tree in the given network to collect and distribute messages.

In order to determine an optimal spanning tree structure for tree barrier, we have to look at the underlying communication mechanism. If communication delay is considered as part of overall performance, the *postal model* [1] can be used which is based on $\lambda = l/s$, where $s$ is the time it takes for a node to send the next message and $l$ is the network latency. For example, assume $s = 2$ and $l = 4$ then $\lambda = 2$. If the source node sends a message at time 0, it has to wait $s = 2$ time steps before sending the message to another neighbor at time 2. The message sent at time 0 will reach the corresponding neighbor at time $l = 4$. Similarly, the one sent at time 2 will reach another neighbor at time 6, etc. Under the one-port model (in which each node can send and receive one message at a time), the binomial tree is optimal when $\lambda = 1$. An optimal tree for a specific $\lambda$ is constructed based on:

$$N_\lambda(n) = \begin{cases} N_\lambda(n-1) + N_\lambda(n-\lambda), & \text{if } n \geq \lambda \\ 1, & \text{otherwise} \end{cases}$$

where $N_\lambda(n)$ represents the maximum number of nodes that can be reached in time $n$ on a one-port model exhibiting $\lambda$. Note that before time $n < \lambda$, only the source node has a copy of the message, although several copies have been sent from the source node since time 0 and they are still in transit. Clearly, the parameter $\lambda$ in the postal model matches the series number $\lambda$ in the postal network. It is easy to derive the corresponding optimal tree structure as: $PT_\lambda(n)$ is constructed out of $PT_\lambda(n-1)$ and $PT_\lambda(n-\lambda)$, with the root node of $PT_\lambda(n-\lambda)$ as the child of the root node of $PT_\lambda(n-1)$. As initial conditions,

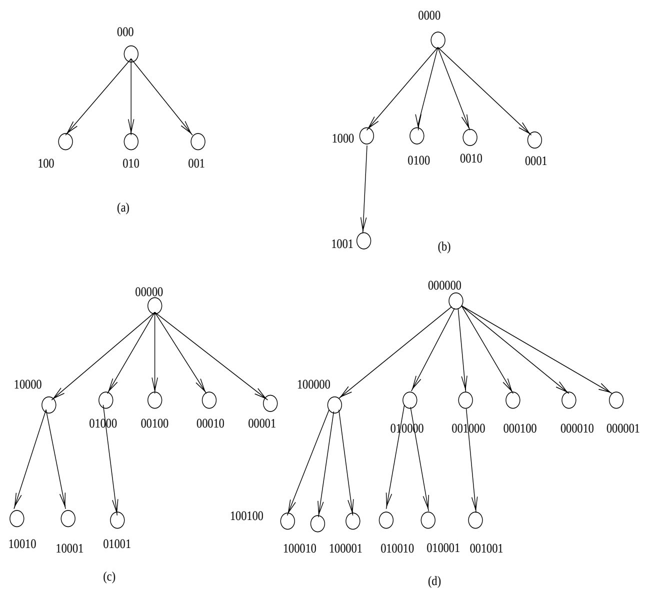

Figure 3: Postal trees: (a) $PT_3(3)$, (b) $PT_3(4)$, (c) $PT_3(5)$, and (d) $PT_3(6)$.

$PT_\lambda(i)$ ($1 \leq i < \lambda$) consists of $i+1$ nodes in a two-level tree. Clearly, $PT_\lambda(n)$ is a spanning tree of $PN_\lambda(n)$. More formally, we have the following definition of a postal tree.

**Definition 2**: *A postal tree $PT_\lambda(n)$ of $PN_\lambda(n)$ is defined as follows: (Base) $PT_\lambda(n) = PN_\lambda(n)$ for $1 \leq n \leq \lambda$. (Recursion) For $\lambda < n$, a $PT_\lambda(n)$ consists of $PT_\lambda(n-1)$ and $PT_\lambda(n-\lambda)$ by connecting the root of $PT_\lambda(n-\lambda)$ as the child of the root of $PT_\lambda(n-1)$.*

Figure 3 shows the structure of postal trees $PT_3(n)$ for $n = 3, 4, 5, 6$. Clearly, $PT_3(n)$ has the same vertex set as $PN_3(n)$. Hence, $PT_3(n)$ is a spanning tree of $PN_3(n)$. Figure 4 shows two different spanning trees in a fully-connected network with eight nodes, when $\lambda = 6$ with $s = 1$ and $l = 6$. Clearly, the binomial tree implementation (Figure 4 (a)) requires 18 units to complete a broadcast and it is no longer optimal. The optimal tree (Figure 4 (b)) needs only 12 units to complete a broadcast. Note that this postal model can be applied to any topology as long as it has sufficient connectivity.

Based on the above analysis, we can see that parameter $\lambda$ plays an important role in selecting networks from different series. Because $\lambda$ defines the ratio of the time it takes for a node to send the next message to the communication latency, it should be carefully selected to minimize communication delay especially for collective communication. For example, if we are to select a network of eight nodes from a series and $\lambda = 1$, $PN_1(3)$ is better than $PN_2(4)$. Here we simplify the selection process without considering other factors. In designing an actual multicomputer system, different factors should be considered and weighted against each other.

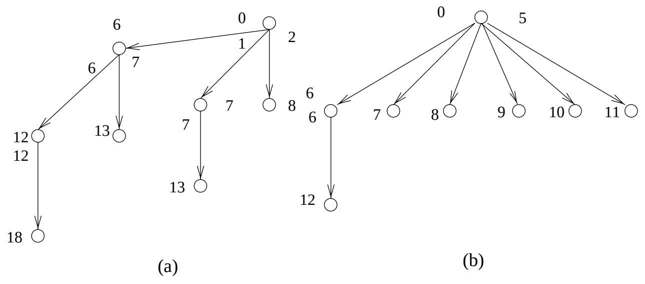

Figure 4: Comparison with $\lambda = 6$: (a) binomial trees $PT_1(3)$ and (b) optimal spanning tree $PT_6(6)$.

## 6 Conclusions

We have proposed a series of networks called postal networks that contain both hypercubes and Fibonacci cubes as their special cases. We have shown that these networks still maintain some desirable properties of hypercubes. Relationships between different networks have also been studied. Postal networks can be used to complement several existing network topologies such as hypercubes and Fibonacci cubes. Because each postal network can also be considered as an incomplete hypercube after some nodes become faulty, the study of postal networks will also provide some insights on the behavior of the cube-based systems operated in a degraded mode. Our future work will focus on embedding other popular structures such as binary trees in postal networks. Another interesting issue is to determine postal networks from a given faulty hypercube.

## References

[1] A. Bar-Noy and S. Kionis. Designing broadcasting algorithms in the postal model for message-passing systems. *Proc. of 1992 Symposium on Parallel Algorithms and Architecture*. 1992, 13-22.

[2] B. Cong, S. Q. Zheng, and S. Sharma. On simulations of linear arrays, rings and 2-d meshes on Fibonacci cube networks. *Proc. of the 7th International Parallel Processing Symposium*. 1993, 748-751.

[3] C. L. Liu. *Elements of Discrete Mathematics*. McGraw-Hill, Inc. 1985.

[4] W. J. Hsu. Fibonacci cubes – a new interconnection topology. *IEEE Transactions on Parallel and Distributed Systems*. 4, (1), Jan. 1993, 3-12.

[5] H. F. Jordan. A special purpose architecture for finite element analysis. *Proc. of 1978 International Conference on Parallel Processing*. 1978, 263-266.

[6] J. Wu. Extended Fibonacci cubes. *IEEE Transactions on Parallel and Distributed Systems*. 8, (12), Dec. 1997, 1203-1210.

[7] P. C. Yew, N. F. Tzeng, and D. H. Lawrie. Distributing hot-spot addressing in large-scale multiprocessors. *IEEE Transactions on Computers*. 36, (4), April 1987, 388-395.