Yale University
Department of Computer Science

# Distributed Routing Algorithms for Broadcasting and Personalized Communication in Hypercubes.

Ching-Tien Ho and S. Lennart Johnsson

YALEU/DCS/TR-483
May 1986

Work supported by ONR Contract N00014-84-K-0043. Approved for public release: distribution is unlimited

[1]To appear in the Proceedings of the ICPP 1986.

# Distributed Routing Algorithms for Broadcasting and Personalized Communication in Hypercubes

Ching-Tien Ho and S. Lennart Johnsson

Department of Computer Science
Yale University
New Haven, CT 06520

## Abstract

High communication bandwidth in standard technologies is more expensive to realize than a high rate of arithmetic or logic operations. The effective utilization of communication resources is crucial for good overall performance in highly concurrent systems. In this paper we address two different communication problems in Boolean $n$-cube configured multiprocessors: 1) broadcasting, i.e., distribution of common data from a single source to all other nodes, and 2) sending personalized data from a single source to all other nodes. The well known spanning tree algorithm obtained by bit-wise complementation of leading zeroes (referred to as the SBT algorithm for *Spanning Binomial Tree*) is compared with an algorithm using *multiple spanning binomial trees* (MSBT). The MSBT algorithm offers a potential speed-up over the SBT algorithm by a factor of $\log_2 N$. We also present a *balanced spanning tree* algorithm (BST) that offers a lower complexity than the SBT algorithm for Case 2. The potential improvement is by a factor of $\frac{1}{2}\log_2 N$. The analysis takes into account the size of the data sets, the communication bandwidth, and the overhead in communication. We also provide some experimental data for the Intel *iPSC/d7*.

## 1. Introduction

Broadcasting of data from a single source to all other nodes in a multiprocessor system is an important operation. It is used in many parallel algorithms, for instance, in matrix multiplication, the solution of irreducible linear systems, and forming transitive closure. Examples of a variety of algorithms using specific forms of communication are contained in [6, 10, 11]. The reverse operation, reduction, occurs, for example, in computing inner products, solving linear recurrences, [13], and parallel prefix computation. A different situation occurs if the source node distributes personalized information to all other nodes. In this case no replication of information takes place during distribution (or reduction in the reverse operation). The collection of data to a single node and distribution of personalized messages to all other nodes is a useful operation for the solution of tridiagonal systems under certain combinations of start-up times for communication, communication bandwidth, and problem sizes [12]. Matrix transposition is another example of personalized communication in that every node sends different data to every other node [11].

Data communication in Boolean cubes has received significant interest recently due to the success of the Caltech Cosmic Cube project [19] and the availability of Boolean cube configured concurrent processors from Intel Scientific Computers, NCUBE, Ametek, Floating-Point Systems and Thinking Machines Corp. [7]. The embedding of complete binary trees is treated in [21, 11, 17, 3, 2]. Wu also discusses the embedding of $k$-ary trees, and Bhatt the embedding of arbitrary binary trees. Efficient routing using randomization for arbitrary permutations has been suggested by Valiant [20] . Broadcasting of data from a single source to all other nodes is studied in [17]. We propose a lower bound algorithm that offers a speed-up of a factor of $\log N$[1] over the algorithm in [17]. We also present lower bound algorithms for personalized communication. We give routing

[1] $\log N = \log_2 N$ throughout this paper.

algorithms, and analyze the complexity in detail. The analysis is compared with experimental data.

A Boolean $n$-cube has $N = 2^n$ nodes, diameter $\log N$, $\binom{\log N}{i}$ nodes at distance $i$ from a given node, and $\log N$ disjoint paths between any pair of nodes. The paths are either of the same length as the Hamming distance between the end points of the paths, or the Hamming distance plus two [18]. The fanout of every node is $\log N$, and the total number of communication links is $\frac{1}{2}N \log N$. Any spanning tree can be used to broadcast data from a single source to all other nodes.[2] A node replicates the data as many times as correspond to the out-degree of the node in the spanning tree. In broadcasting one element (or packet), the minimum number of routing steps is $\log N$. Any spanning tree with height $\log N$ can achieve this lower bound, if each node can send out data through all the links connected to it during one step. In case each node can send or receive data through only one link during one step, then only the class of ***spanning binomial trees*** can attain the lower bound, since after each broadcasting step the number of nodes that own the desired data is at most twice that of the previous step. The $\log N$ lower bound is attained only if the number of nodes that own the desired data doubles at each step. This is exactly the definition of a binomial tree. A 0-level binomial tree has only 1 node. An $n$-level binomial tree is constructed out of two $(n-1)$-level binomial trees by adding one edge between the roots of the two trees, and by making either root the new root, [1, 4]. It follows from this recursive construction that:

1. An $n$-level binomial tree has $\binom{n}{i}$ nodes at level $i$.
2. The $n$-level binomial tree is composed of $n$ subtrees[3] each of which is a binomial tree of $0, 1, \ldots, n-1$ levels respectively. The $k$-level subtree has $2^k$ nodes.
3. An $n$-level binomial tree can be obtained from a $k$-level binomial tree, $k < n$, by replacing each node of the $k$-level binomial tree by an $(n-k)$-level binomial tree. The child nodes of a node in the $k$-level tree become children of the root of the replacing $(n-k)$-level binomial tree.

Since an $n$-level binomial tree can be embedded in an $n$-cube as a spanning tree, we call it a *Spanning Binomial Tree* (SBT). Note that the number of nodes at each level $i$ of the binomial tree is equal to the number of nodes at distance $i$ from a node in an $n$-cube.

In broadcasting $M$ elements using a packet size of $B$ elements, and by pipelining the communication from the root towards the leaves along any $\log N$ height spanning tree, the number of routing steps becomes $\lceil \frac{M}{B} \rceil + \log N - 1$, which is not optimal. Since each node has a fanout of $\log N$, a lower bound for the number of routing steps is $\lceil \frac{M}{B \log N} \rceil + \log N - 1$. In order to achieve this lower bound, the data set has to be split into $\log N$ subsets, each of which is communicated over a distinct communications link from the source node. It follows that the nodes adjacent to the source node must be roots of subtrees spanning all but one node of the cube (the source node). The depth of the subtrees is $\log N$, and a tight lower bound for the number of routing steps, assuming concurrent bi-directional communication, is $\lceil \frac{M}{B \log N} \rceil + \log N$.

In sending personalized information from a single source to all other nodes, no replication of information takes place and the total number of packets that the source must send is $\lceil \frac{(N-1)M}{B} \rceil$ for $M$ elements per destination node. The number of routing steps for the ***Spanning Binomial Tree*** algorithm (SBT) is $\log N$ if the maximum packet size is sufficiently large $(NM/2)$; even so, the communication bandwidth is poorly utilized since the transfer time is at least proportional to $NM/2$. In the SBT, half of the nodes belong to one subtree, one quarter to another subtree, etc. A *Balanced Spanning Tree* (BST) is defined in that each subtree has approximately $\frac{N}{\log N}$ nodes. The data transfer on any link is limited to approximately $\frac{N}{\log N}M$. The BST algorithm potentially offers a speed-up of $\log N$ over the SBT algorithm in sending personalized information from a single source to all other nodes. In fact, lower bound algorithms for broadcasting from every node to every

[2] In particular, a Hamiltonian Path is also a spanning tree.

[3] In this paper "subtree" refers to "subtree of the root" unless stated otherwise.

other node and sending personalized data from every node to every other node on a Boolean cube can be attained by using $N$ BST's rooted at each node concurrently. See [8] for details.

In section 2 we introduce the notation and some definitions used throughout the paper. Section 3 considers broadcasting from a single source to all other nodes, and section 4, personalized communication from a single source to all other nodes. Experimental results on the Intel *iPSC/d7* are presented in section 5.

## 2. Notation and Definitions

In the following, $N$ denotes the number of nodes in the Boolean $n$-cube and $n = \log_2 N$ the dimension of the cube. Nodes in the cube are assigned binary addresses such that adjacent nodes differ in precisely one bit. Address bits are numbered from 0 through $n-1$ with the lowest order bit being *bit 0*. Node $i$ is the node that has a binary address equal to $i$, i.e., $i = (a_{n-1}a_{n-2}\ldots a_0)$. Let $\oplus$ be the bit-wise exclusive-or operation. The $j^{th}$ port of a node $i$ connects to the node $k$ that differs from $i$ in the $j^{th}$ bit, i.e., $i \oplus k = (00...01_j0...0)$. There is a port for each address bit, and ports are numbered from 0 through $n-1$. Let $|i|$ denote the number of bits with value one in the binary number $i$; hence $|i \oplus j|$ denotes the Hamming distance between the binary numbers $i$ and $j$.

Let $i = (a_{n-1}a_{n-2}\ldots a_0)$. Define $R$ to be the right rotation function, i.e.,

$$R(i) = (a_0a_{n-1}a_{n-2}\ldots a_1)$$

and $R^j = R^{j-1} \circ R$ to mean a right rotation of $j$ steps. The *rotation* of a graph with binary node addresses is accomplished by applying the same rotation function to all its addresses. This is similarly the case for the *translation* of a graph. Clearly, adjacency is preserved under rotation and translation. The period of a binary number $i$, $P_i$, is the least $j$ such that $i = R^j(i)$. For example, the period of (011011) is 3. A binary number is *cyclic* if its period is less than its length; otherwise it is *non-cyclic*. A *relative address* of node $i$ in a spanning tree rooted at node $s$ is $i \oplus s$. A *cyclic node* is a node with cyclic relative address.[4] If one binary number can be derived by rotating another binary number, then they are in the same *generator set* $G$ (or *necklace* [14]). For example, (001001), (010010) and (100100) are in the same generator set. The number of elements in the generator set $G_i$ of $i$ is $P_i$.

In the graph model of the Boolean cube there is a node (vertex) for each node (processor with local memory) of the cube, and a pair of directed edges for each pair of nodes that differ in precisely one bit. The directed edges between a pair of nodes form a communication link. A *source node*, or a *root node*, is a node that only has edges directed away from it. A *sink node*, or a *leaf node*, only has edges directed to it. Nodes that are neither source (root) nor sink (leaf) nodes are *internal nodes*. The root of a tree is at *level 0* and traversing the edge away from the root increases the level by one. The height of a tree is equal to the label of the last level. The MSBT and BST are constructed out of $n$ subtrees labelled 0 through $n-1$ (from left to right in the Figures of this paper).

The communication is assumed to be packet switched. $M$ denotes the number of elements to be received by a node, $t_c$ the transfer time for an element, and $\tau$ the start-up time for the communication of a packet of maximum $B$ elements. With concurrent bi-directional communication we assume that a pair of adjacent nodes can exchange a pair of messages during the same communication step, or cycle. In a *port-oriented* routing algorithm, all information to be communicated over a port is sent before any communication is performed on any other port. In *packet-oriented* communication, a piece of information corresponding to a packet is communicated on all ports before a second packet is sent on any port.

[4]A cyclic node is defined in terms of a spanning tree.

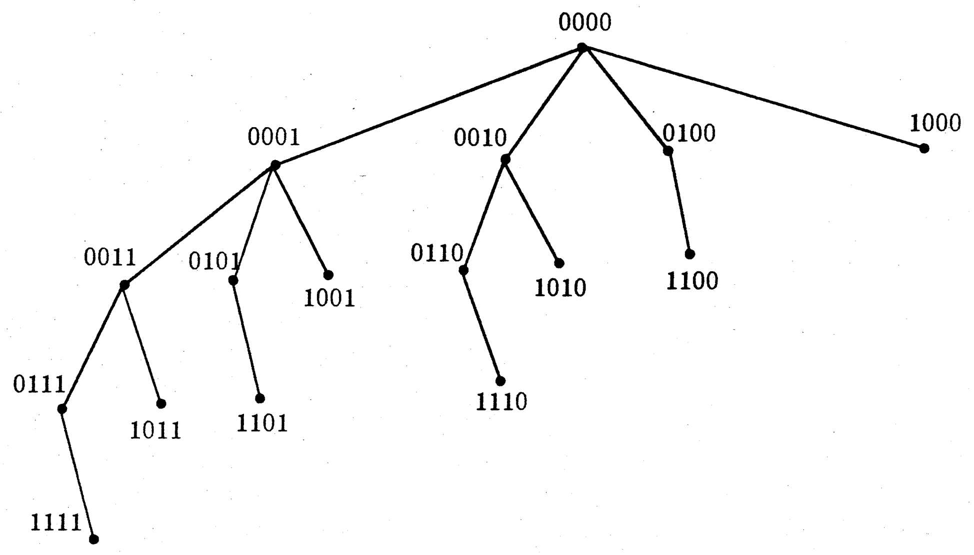

**Figure 1: A spanning tree in a 4-cube**

## 3. Broadcasting

In this section, we will describe and compare routing algorithms for broadcasting based on a Spanning Binomial Tree (SBT) and Multiple Spanning Binomial Trees (MSBT). We first define the SBT and MSBT topologies, then state the communication complexity of the routing algorithms for the two topologies, assuming communication on only one port at a time (which, effectively, is the case with the Intel *iPSC*), and on all $\log N$ ports concurrently.

### 3.1. Spanning Binomial Trees

The familiar spanning tree rooted at node 0 of a Boolean $n$-cube contains the edges that connect a node $i$ with the subset of its neighbors having addresses obtained by complementing any bit of leading zeroes of the binary encoding of $i$, [5, 11, 15, 17, 18]. For an arbitrary source node $s$ the spanning tree is simply translated by a bit-wise exclusive-or operation on all addresses with the address of the source node; i.e., $c = i \oplus s$ is formed. Complementation of those bits of $i$ that correspond to the leading zeroes of $c$ defines the edges of the translated spanning tree. More precisely, let $s = (s_{n-1}s_{n-2}...s_0)$, $i = (a_{n-1}a_{n-2}...a_0)$, and $c = (c_{n-1}c_{n-2}...c_0)$, where $c_m = s_m \oplus a_m$. Let $c_k = 1$ and $c_m = 0, \forall m > k$ with $k = -1$ for $c = 0$, i.e., $k$ is the highest order bit of $c$ that is 1. Let $children(i, s)$ be the set of child nodes of node $i$ in the SBT rooted at node $s$ and $\mathcal{M}_{SBT}(i \oplus s) = \{k+1, \ldots, n-1\}$. Then,

$$children_{SBT}(i,s) = \{(a_{n-1}a_{n-2}\ldots\bar{a}_m\ldots a_0)\},\quad \forall m \in \mathcal{M}_{SBT}(i \oplus s)$$

In implementing the routing algorithm for the SBT topology it is also convenient to introduce the inverse function, i.e., a function that for each node defines its parent. Let $parent(i, s)$ be the parent of node $i$ in the spanning tree rooted at node $s$. Then

$$parent_{SBT}(i,s) = \begin{cases} \phi, & i = s; \\ (a_{n-1}a_{n-2}\ldots\bar{a}_k\ldots a_0), & i \neq s. \end{cases}$$

It is easy to verify that the parent and children functions are consistent. Figure 1 shows a spanning tree generated by the children (or parent) function for the root located at node 0 in a 4-cube.

## 3.2. Multiple Spanning Binomial Trees

The Multiple Spanning Binomial Trees (MSBT) graph can be viewed as being composed of $\log N$ SBT's with one tree rooted at each of the nodes adjacent to the source node. The SBT's are rotated such that the source node of the MSBT graph is in the smallest subtree of each SBT. The MSBT graph is then obtained by reversing the edges from the roots of the SBT's to the source node. After the edge reversal each SBT becomes an ERSBT (Edge Reversed Spanning Binomial Tree). The MSBT graph is not a tree. The diameter of the MSBT graph is $\log N + 1$, since the source node is adjacent to all the roots of the SBT's used in the definition of the MSBT graph, and each SBT is of height $\log N$. The total number of edges in the $\log N$ SBT's is $(N-1)\log N$, which is $\log N$ less than the total number of directed edges in the cube. Hence, if the $\log N$ SBT's are edge-disjoint, then all edges are used, except the edges directed from the roots of the SBT's towards the source node. The SBT's used for the construction of the MSBT graph can be obtained by translation and rotation of the SBT defined before. We refer to the SBT rooted at node $(00\ldots01_j0\ldots0)$ as the $j^{th}$ SBT of the MSBT graph. The $j^{th}$ ERSBT is obtained from the $j^{th}$ SBT by reversing the edge directed to node 0 (the source).

Let $i = (a_{n-1}a_{n-2}\ldots a_0)$ and $k$ be such that $a_k = 1$, and $a_m = 0, \forall m \in \mathcal{M}_{MSBT}(i,j)$ where $\mathcal{M}_{MSBT}(i,j) = \{(k+1) \bmod n, (k+2) \bmod n, \ldots, (j-1) \bmod n\}$. Hence, $k$ is the first bit to the right of bit $j$, cyclically, which is equal to one, if $k \neq j$. For the special case of $i = 0$ we define $k = -1$. For the $j^{th}$ ERSBT of the MSBT graph with source node 0, the set of child nodes, and the parent node, of node $i$ are defined as follows:

$$children_{MSBT}(i,j,0) = \begin{cases} (a_{n-1}a_{n-2}\ldots\overline{a}_j\ldots a_0), & \text{if } k = -1; \\ \{(a_{n-1}a_{n-2}\ldots\overline{a}_m\ldots a_0)\}, & \\ \quad \forall m \in \mathcal{M}_{MSBT}(i,j) \bigcup \{j\}, & \text{if } a_j = 1,\ k \neq j; \\ \{(a_{n-1}a_{n-2}\ldots\overline{a}_m\ldots a_0)\}, & \\ \quad \forall m \in \mathcal{M}_{MSBT}(i,j), & \text{if } a_j = 1,\ k = j; \\ \phi, & \text{if } a_j = 0, k \neq -1. \end{cases}$$

$$parent_{MSBT}(i,j,0) = \begin{cases} \phi, & \text{if } k = -1; \\ (a_{n-1}a_{n-2}\ldots\overline{a}_j\ldots a_0), & \text{if } a_j = 0, k \neq -1; \\ (a_{n-1}a_{n-2}\ldots\overline{a}_k\ldots a_0), & \text{if } a_j = 1. \end{cases}$$

All nodes with bit $j$ equal to zero are leaf nodes of the $j^{th}$ ERSBT, except node 0. Conversly, all nodes with $a_j = 1$ are internal nodes of the $j^{th}$ ERSBT. Figure 2 shows an MSBT graph with source node 0 in a 3-cube.

It can be shown that the $\log N$ directed ERSBT's are edge-disjoint and the height of the MSBT graph is minimal among all possible configurations of $\log N$ edge-disjoint spanning trees[8].

For an arbitrary source node $s$ an MSBT graph is defined by translating the MSBT graph rooted at node 0. The only difference in the definition of the *parent* and *children* functions is that $k$ is determined from $c = i \oplus s$. Hence, for a source node $s$, $k$ is such that $c_k = 1$, and $c_m = 0, \forall m \in \mathcal{M}_{MSBT}(i \oplus s, j)$. For the special case of $c = 0$, $k = -1$.

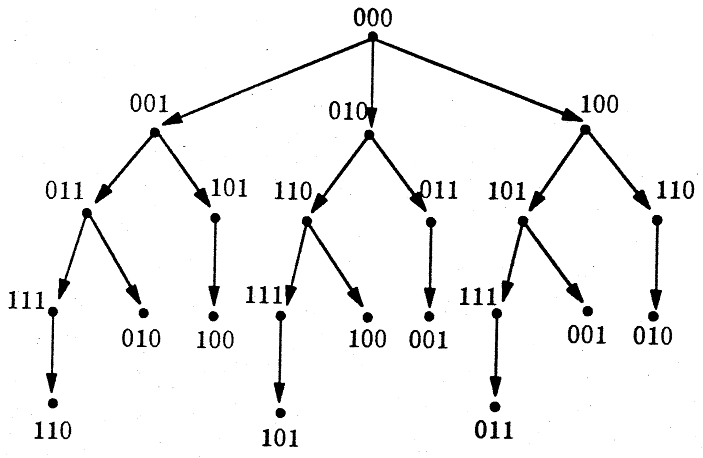

**Figure 2:** Three edge-disjoint directed spanning trees in a 3-cube.

$$children_{MSBT}(i,j,s) = \begin{cases} (a_{n-1}a_{n-2}...\bar{a}_j...a_0), & \text{if } k = -1; \\ \{(a_{n-1}a_{n-2}...\bar{a}_m...a_0)\}, & \\ \quad \forall m \in \mathcal{M}_{MSBT}(i \oplus s, j) \bigcup \{j\}, & \text{if } c_j = 1,\ k \neq j; \\ \{(a_{n-1}a_{n-2}...\bar{a}_m...a_0)\}, & \\ \quad \forall m \in \mathcal{M}_{MSBT}(i \oplus s, j), & \text{if } c_j = 1,\ k = j; \\ \phi, & \text{if } c_j = 0, k \neq -1. \end{cases}$$

$$parent_{MSBT}(i,j,s) = \begin{cases} \phi, & \text{if } k = -1; \\ (a_{n-1}a_{n-2}...\bar{a}_j...a_0), & \text{if } c_j = 0, k \neq -1; \\ (a_{n-1}a_{n-2}...\bar{a}_k...a_0), & \text{if } c_j = 1. \end{cases}$$

## 3.3. Communication Complexity of SBT- and MSBT-based Broadcasting

### 3.3.1. Spanning Binomial Trees

With the communication restricted to one port at a time the data is first sent to the node that is the root of the largest subtree. Since the binomial tree is composed of two $(n-1)$-level binomial trees, the broadcasting operation is now reduced to the broadcasting of data in two same-sized, disjoint, subtrees of the cube. The process is repeated $\log N$ times and the complexity is $T = \lceil \frac{M}{B} \rceil (\tau + Bt_c) \log N$. Clearly $B_{opt} = M$ and $T_{min} = \log N(\tau + Mt_c)$. The data transfer time is independent of the packet size, but the number of start-ups decreases.

With a capability of concurrent communication on all ports, pipelining can be employed extensively. The propagation time to the node farthest away from the source is at least $\log N(\tau + Bt_c)$. When this node has received all packets the broadcasting is terminated. Hence, $T = (\lceil \frac{M}{B} \rceil + \log N - 1)(\tau + Bt_c)$, $B_{opt} = \sqrt{\frac{M\tau}{t_c(\log N - 1)}}$, and $T_{min} = (\sqrt{Mt_c} + \sqrt{\tau(\log N - 1)})^2$. The communication complexity estimates for the SBT are also given in [17] and are included here for easy reference.

### 3.3.2. Multiple Spanning Binomial Trees

We consider the cases with communication restricted to one send *or* one receive operation at a time, one send *and* one receive operation concurrently, and concurrent communication on all ports.

The minimum number of routing steps to broadcast $\log N$ packets is $2\log N$ with communication restricted to one send *and* one receive operation at a time. To realize this lower bound it is required that a routing algorithm be found that allows concurrent communication within all subtrees without violating the constraint on concurrent communication. We describe such a routing algorithm in terms of labelling the MSBT graph with the least label being 0. A valid labelling for the restriction of one receive and one send operation concurrently (per node) and allowing pipelining every $\log N$ cycles, requires that the following three conditions be satisfied:

1. For any node of each subtree the least label on the output edges is greater than the label on the input edge.
2. For any cube node the labels on its input edges are distinct modulo $\log N$. (If there is more than one packet per subtree then the root can send out a new packet to every subtree every $\log N$ cycles.)
3. For any cube node the labels on the output edges are distinct modulo $\log N$.

Let $i = (a_{n-1}a_{n-2}...a_0)$ and $f(i,j)$ be the label of the input edge of node i in the $j^{th}$ subtree for an MSBT graph with source node $s$. Let $c = i \oplus s$, $c_k = 1$ and $c_m = 0, \forall m \in \mathcal{M}_{MSBT}(i \oplus s, j)$. If $c = 0$ then $k = -1$. Define

$$f(i,j) = \begin{cases} \phi, & \text{if } k = -1; \\ j+n, & \text{if } c_j = 0, k \neq -1; \\ k, & \text{if } c_j = 1, k \geq j; \\ k+n, & \text{if } c_j = 1, k < j. \end{cases}$$

It can be proved that function $f$ satisfies these three conditions[8]. From the labelling scheme, the largest label of all the input edges is $2n-1$, i.e., broadcasting the first $\log N$ packets (one packet per subtree) can be done in $2\log N$ steps. The MSBT graph allows $M$ elements to be broadcast in $\lceil \frac{M}{B} \rceil + \log N$ routing steps under the constraint of one receive operation concurrent with one send operation. This is a strict lower bound for $\frac{M}{B} > 1$[8]. Figure 3 shows an MSBT graph for a 3-cube labelled by the function $f$ defined above.

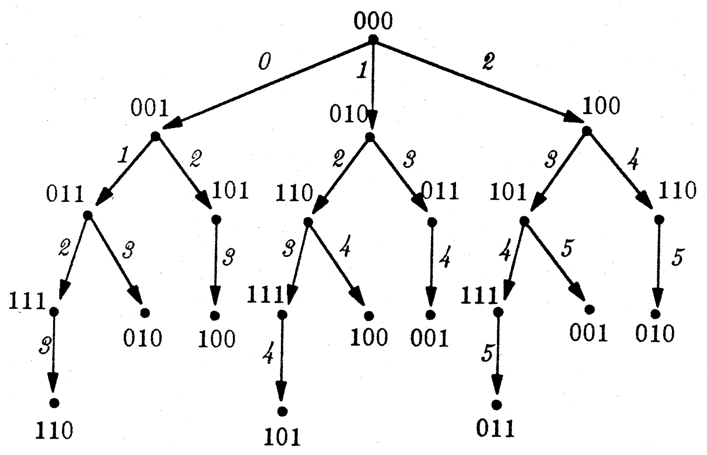

**Figure 3:** Routing in a MSBT graph with communication on one port at a time.

For communication restricted to one send *or* one receive operation per node, we can transform each previously defined cycle into two cycles. Notice that in the previous routing algorithm, all

| Algorithm | 1 $s$ or $r$ | 1 $s$ and $r$ | all ports |
|---|---|---|---|
| HP | $N-1$ | $N-1$ | $N-1$ |
| SBT | $\log N$ | $\log N$ | $\log N$ |
| TCBT | $2\log N-2$ | $2\log N-2$ | $\log N$ |
| MSBT | $3\log N-1$ | $2\log N$ | $\log N+1$ |

**Table 1:** Propagation delays.

| Algorithm | 1 $s$ or $r$ | 1 $s$ and $r$ | all ports |
|---|---|---|---|
| HP | 2 | 1 | 1 |
| SBT | $\log N$ | $\log N$ | 1 |
| TCBT | 3 | 2 | 1 |
| MSBT | 2 | 1 | $1/\log N$ |

**Table 2:** Number of cycles per distinct packet.

the communication links are used in only one direction during the first $n$ routing steps, and in the last routing step. Hence the MSBT graph allows $M$ elements to be broadcast in $2\lceil\frac{M}{B}\rceil+\log N-1$ routing steps under the constraint of at most one receive or one send operation during each step. This is a strict lower bound for $\frac{M}{B}>1$[8].

With a maximum packet size of $B$ and one receive operation concurrent with a send operation the communication complexity for MSBT-based broadcasting is $T=(\lceil\frac{M}{B}\rceil+\log N)(\tau+Bt_c)$, which is minimized for $B_{opt}=\sqrt{\frac{M\tau}{t_c\log N}}$. $T_{min}=(\sqrt{Mt_c}+\sqrt{\tau\log N})^2$. Restricting the communication to one send *or* one receive operation at a time, $T=(2\lceil\frac{M}{B}\rceil+\log N-1)(\tau+Bt_c)$ and $T_{min}=(\sqrt{2Mt_c}+\sqrt{\tau(\log N-1)})^2$.

For the case in which communication on all ports can take place concurrently the communication complexity is $T=(\lceil\frac{M}{B\log N}\rceil+\log N)(\tau+Bt_c)$. With optimal packet size $B_{opt}=\frac{1}{\log N}\sqrt{\frac{M\tau}{t_c}}$, $T_{min}=(\sqrt{\frac{Mt_c}{\log N}}+\sqrt{\tau\log N})^2$.

## 3.4. Comparison and Conclusion

In the following, we also compare the MSBT algorithm with the one based on a Two-rooted Complete Binary Tree (TCBT) [2, 3] or a Hamiltonian Path (HP) as a broadcasting tree respectively. Table 1 shows the propagation delay of various algorithms. Interestingly, broadcasting through a Hamiltonian Path on a hypercube may be faster than broadcasting based on the SBT or even the TCBT, depending on the values of $M$, $t_c$, $\tau$ and $N$. With communication on all the ports concurrently, the MSBT-based algorithm can send out $\log N$ distinct packets every cycle while the SBT- and TCBT-based algorithms can only send out one distinct packet every cycle. Table 2 compares the number of cycles per distinct packet for various algorithms. Some variations exist, such as using two Hamiltonian paths with opposite directions sending distinct data, or using one Hamiltonian path such that the source node is at the center of the path. However, these variations only affect (either increase or decrease) delays, and the number of cycles per packet, by at most a factor of two.

The complexity estimates are summarized in Table 3. A potential for concurrent communication on all ports reduces the number of sequential start-ups and the bandwidth requirement by a factor of approximately $\log N$ for an arbitrary packet size in both SBT- and MSBT-based broadcasting. TCBT-based broadcasting does not fully utilize the bandwidth of a cube. The reduction in communication complexity for concurrent communication on all ports is a factor of 2 or 3. Optimizing the packet size for each situation brings the number of start-ups to $O(\log N)$, irrespective

| Algorithm | T | $B_{opt}$ | $T_{min}$ |
|---|---|---|---|
| HP, 1 *s* or *r* | $(2\lceil \frac{M}{B}\rceil + N - 3)(\tau + Bt_C)$ | $\sqrt{\frac{2M\tau}{(N-3)t_c}}$ | $(\sqrt{2Mt_c} + \sqrt{(N-3)\tau})^2$ |
| HP, 1 *s* & *r* | $(\lceil \frac{M}{B}\rceil + N - 3)(\tau + Bt_C)$ | $\sqrt{\frac{M\tau}{(N-3)t_c}}$ | $(\sqrt{Mt_c} + \sqrt{(N-3)\tau})^2$ |
| SBT, 1 port | $\lceil \frac{M}{B}\rceil \log N(\tau + Bt_c)$ | $M$ | $\log N(Mt_c + \tau)$ |
| SBT, $\log N$ ports | $(\lceil \frac{M}{B}\rceil + \log N - 1)(\tau + Bt_c)$ | $\sqrt{\frac{M\tau}{(\log N-1)t_c}}$ | $(\sqrt{Mt_c} + \sqrt{\tau(\log N - 1)})^2$ |
| TCBT, 1 *s* or *r* | $(3\lceil \frac{M}{B}\rceil + 2\log N - 5)(\tau + Bt_c)$ | $\sqrt{\frac{3M\tau}{(2\log N-5)t_c}}$ | $(\sqrt{3Mt_c} + \sqrt{\tau(2\log N - 5)})^2$ |
| TCBT, 1 *s* & *r* | $2(\lceil \frac{M}{B}\rceil + \log N - 2)(\tau + Bt_c)$ | $\sqrt{\frac{M\tau}{(\log N-2)t_c}}$ | $2(\sqrt{Mt_c} + \sqrt{\tau(\log N - 2)})^2$ |
| TCBT, $\log N$ ports | $(\lceil \frac{M}{B}\rceil + \log N - 1)(\tau + Bt_c)$ | $\sqrt{\frac{M\tau}{t_c(\log N-1)}}$ | $(\sqrt{Mt_c} + \sqrt{\tau(\log N - 1)})^2$ |
| MSBT, 1 *s* or *r* | $(2\lceil \frac{M}{B}\rceil + \log N - 1)(\tau + Bt_c)$ | $\sqrt{\frac{2M\tau}{t_c(\log N-1)}}$ | $(\sqrt{2Mt_c} + \sqrt{\tau(\log N - 1)})^2$ |
| MSBT, 1 *s* & *r* | $(\lceil \frac{M}{B}\rceil + \log N)(\tau + Bt_c)$ | $\sqrt{\frac{M\tau}{t_c\log N}}$ | $(\sqrt{Mt_c} + \sqrt{\tau\log N})^2$ |
| MSBT, $\log N$ ports | $(\lceil \frac{M}{B\log N}\rceil + \log N)(\tau + Bt_c)$ | $\frac{1}{\log N}\sqrt{\frac{M\tau}{t_c}}$ | $(\sqrt{\frac{Mt_c}{\log N}} + \sqrt{\tau\log N})^2$ |

**Table 3:** Communication complexity based on various graphs.

| Communication Assumption | Algorithm | one packet | $\frac{M}{B} \gg \log N$ | $B = B_{opt}$, $\tau\log N \gg Mt_c$ | $B = B_{opt}$, $\tau\log N \ll Mt_c$ |
|---|---|---|---|---|---|
| 1 *send* or *recv* | SBT/MSBT | $\frac{\log N}{\log N+1}$ | $\frac{1}{2}\log N$ | 1 | $\frac{1}{2}\log N$ |
| 1 *send* or *recv* | TCBT/MSBT | $\frac{2\log N-2}{\log N+1}$ | 1.5 | 2 | 1.5 |
| 1 *send* and *recv* | SBT/MSBT | $\frac{\log N}{\log N+1}$ | $\log N$ | 1 | $\log N$ |
| 1 *send* and *recv* | TCBT/MSBT | $\frac{2\log N-2}{\log N+1}$ | 2 | 2 | 2 |
| all ports | SBT,TCBT/MSBT | $\frac{\log N}{\log N+1}$ | $\log N$ | 1 | $\log N$ |

**Table 4:** Communication complexity compared to the MSBT routing.

of whether communication on one port or $\log N$ ports at a time is possible.

The MSBT-based broadcasting always offers a reduction in the bandwidth requirement for individual communication links by a factor of approximately $\log N$ over SBT-based broadcasting. With communication on all ports concurrently, the MSBT-based broadcasting has a communication complexity that is lower than that of TCBT-based broadcasting by a factor of $\log N$. Even with communication only on one port at a time, MSBT-based broadcasting still is faster than TCBT-based broadcasting by a factor of 1.5 or 2. The communication complexities of broadcasting based on the SBT and the TCBT are compared with that based on the MSBT in Table 4.[5]

## 4. Personalized Communication

In personalized communication no replication of information takes place during distribution, nor is there any reduction during the reverse operation. In broadcasting, the bandwidth requirement grows with the distance $i$ from the source node precisely as the number of nodes grow. In personalized communication the bandwidth requirement instead decreases in proportion to the number of nodes less than or equal to distance $i$ from the source. The root is the "bottleneck" in personalized communication. In this section we define a pruning strategy for the MSBT graph

[5] Notice that the entry for the last column and the last row in the table is based on the assumption that $B = B_{opt}, \tau\log^2 N \ll Mt_c$.

that generates a *balanced spanning tree* (BST) of height $\log N$. If concurrent communication on all $\log N$ ports (of the root) is possible, then a lower bound for the transmission time is $\frac{N}{\log N}Mt_c$, and a lower bound for the number of start-ups is $\log N$. The BST makes possible personalized communication in a time corresponding to this lower bound.

## 4.1. A Balanced Spanning Tree

In the SBT topology a node $i$ belongs to the $j^{th}$ subtree iff $a_j = 1$, $a_k = 0$, $k < j$. In the MSBT graph a node is an internal node of the $j^{th}$ ERSBT if $a_j = 1$. Bit $j$ can be considered as a *base* for the $j^{th}$ subtree. For the BST we define the base as follows:

Let $J_i = \{j_1, j_2, \ldots, j_m\}$, where $j_1 < j_2 < \ldots j_m$, $R^u(i) = R^v(i)$, $u, v \in J_i$, and $R^u(i) < R^l(i)$, $u \in J_i$, $l \notin J_i$. $|J_i| = n/P_i$ where $P_i$ is the period of $i$. Then $base(i) = j_1$ and node $i$ is assigned to subtree $j_1$, i.e., the value of the base equals the minimum number of right rotations such that the rotated number has a minimum value among all the rotated values.[6] For example, $base((011010)) = 3$ and $base((110110)) = 1$. The period of (011010) is 6 and the period of (110110) is 3. For ease of notation we omit the subscript on $j$ in the following. For the definition of the *parent* and *children* functions we first find the position $k$ of the first bit cyclically to the right of bit $j$ that is equal to 1, i.e., $a_k = 1$, and $a_m = 0, \forall m \in \mathcal{M}_{MSBT}(i,j)$, ($k = j$, if every bit but $j$ is 0). For $i = 0$, $k = -1$. Then

$$parent_{BST}(i,0) = \begin{cases} \phi, & \text{if } i = 0; \\ (a_{n-1}a_{n-2}...\bar{a}_k...a_0), & \text{otherwise.} \end{cases}$$

$$children_{BST}(i,0) = \begin{cases} \{(a_{n-1}a_{n-2}...\bar{a}_m...a_0)\}, \forall m \in \{0,1,\ldots,n-1\}, & \text{if } i = 0; \\ \{q_m = (a_{n-1}a_{n-2}...\bar{a}_m...a_0)\}, & \\ \quad \forall m \in \mathcal{M}_{MSBT}(i,j) \text{ and } base(q_m) = base(i), & \text{if } i \neq 0. \end{cases}$$

The $parent_{BST}$ function preserves the base, since for any node $i$ with base $j$, $a_k$ is the highest order bit of $R^j(i)$. Complementing this bit cannot change the base. It is also readily seen that the $parent_{BST}$ and $children_{BST}$ functions are consistent.

Figure 4 shows the spanning tree generated by the algorithm above for the root located at node 0 in a 5-cube.

For an arbitrary source node $s$ we translate the BST rooted at node 0 to node $s$ by performing for each node the bit-wise exclusive-or function of its address and the address of the source node. The base of a node is determined from $c = i \oplus s$, and the children and parent functions are readily modified.

Let $J_{i,s} = \{j_1, j_2, \ldots, j_m\}$, where $j_1 < j_2 < \ldots j_m$, $R^u(c) = R^v(c)$, $u, v \in J_{i,s}$, and $R^u(c) < R^l(c)$, $u \in J_{i,s}$, $l \notin J_{i,s}$. Then $base(c) = j_1$. Then $k$ is defined by $c_k = 1$ and $c_m = 0, \forall m \in \mathcal{M}_{MSBT}(i \oplus s, j)$ with $k = -1$ if $c = 0$.

$$parent_{BST}(i,s) = \begin{cases} \phi, & \text{if } c = 0; \\ (a_{n-1}a_{n-2}...\bar{a}_k...a_0), & \text{otherwise.} \end{cases}$$

$$children_{BST}(i,s) = \begin{cases} \{(a_{n-1}a_{n-2}...\bar{a}_m...a_0)\}, \forall m \in \{0,1,\ldots,n-1\}, & \text{if } c = 0; \\ \{q_m = (a_{n-1}a_{n-2}...\bar{a}_m...a_0)\}, & \\ \quad \forall m \in \mathcal{M}_{MSBT}(i \oplus s, j) & \\ \quad \text{and } base(q_m \oplus s) = base(i \oplus s), & \text{if } c \neq 0. \end{cases}$$

**Lemma 4.1.** *The number of nodes in a subtree is of order* $O(\frac{N}{\log N})$.

*Proof.* With $A$ cyclic nodes there are at least $(N - A)/\log N$ nodes in a subtree. Denoting the number of generator sets for cyclic nodes by $B$ it follows that the maximum number of nodes in

[6] The notion of *base* is similar to the idea of distinguished node used in [14] in that *base* = 0 distinguishes a node from a generator set (necklace).

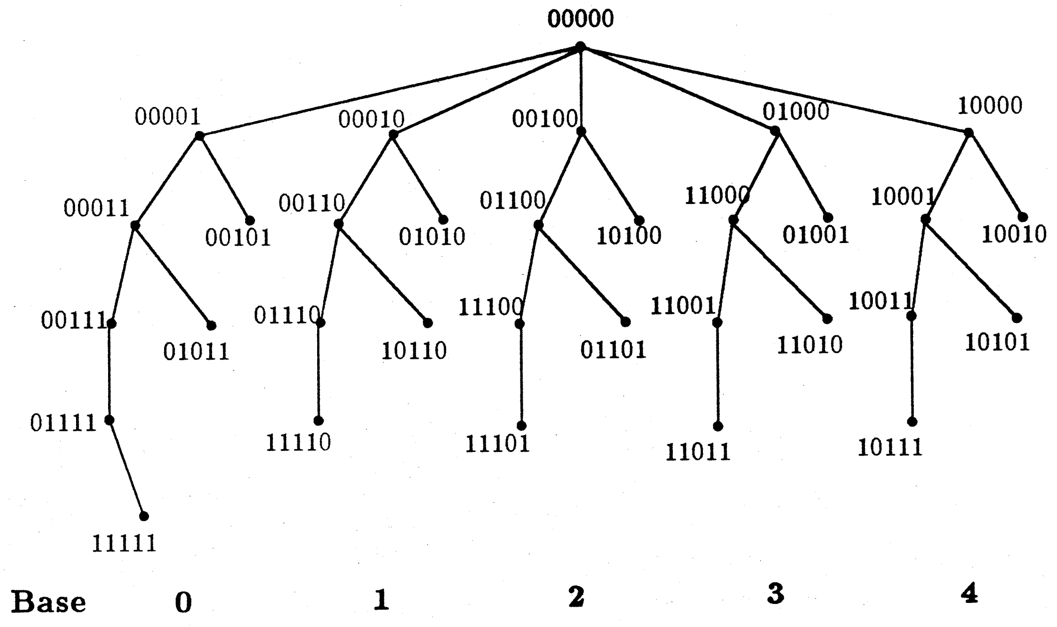

**Figure 4:** A balanced spanning tree in a 5-cube.

a subtree is $(N - A)/\log N + B - 1$. To derive bounds on $A$ we use the complex plane diagram used by Hoey and Leiserson [9] in studying the shuffle-exchange network. Leighton[14] shows that $B = O(\sqrt{N})$.

Full necklaces, i.e., non-cyclic nodes, are mapped to circles. Degenerate necklaces, i.e., cyclic nodes, are mapped to the origin. In the context of the shuffle-exchange graph each node that is mapped to the origin of the complex plane is adjacent (via an exchange edge) to a node at position $(1,0)$ or $(-1,0)$. Hence, for every full necklace of $\log N$ nodes there are at most 2 cyclic nodes. Node 0 is adjacent to a node of a full necklace, and so is node $N-1$ (for $\log N > 2$). It follows that an upper bound on $A$ is $2\frac{N-\log N}{2+\log N}$ and the number of nodes in a subtree is at least $\frac{N+2}{2+\log N}$. The relative difference in the number of nodes in the maximum and minimum subtrees approaches 0 for $N \to \infty$.

∎

Table 5 gives the sizes of the maximum subtrees generated according to the definition of the BST for up to 20-dimensional cubes. The relative difference approaches 0 rapidly. The last column contains the ratio of BST(max) to $(N-1)/\log N$.

Some properties of the BST are listed below. For detailed proofs see [8].

1. The height of one subtree is $\log N$, and the height of all other subtrees is $\log N - 1$.
2. The maximum fanout of any node at level $i$ in a BST is $\lceil \frac{\log N - i}{2} \rceil$ for $1 \leq i \leq logN$.
3. Let $\phi(i,j)$ be the number of nodes at distance $j$ from node $i$ in the subtree rooted at node $i$. Then, $\phi(i,j) \geq \phi(k,j)$ where node $k$ is a child of node $i$.[7]
4. Excluding node $i \oplus s = (11\ldots1)$, all the subtrees of the BST are isomorphic if $\log N$ is a prime number.
5. Subtrees $P$ to $\log N - 1$ contain no cyclic nodes with period $P$.
6. Any cyclic node is a leaf node of the BST.

[7]This property is required in deriving the communication complexity for the BST routing.

| $n$ | BST(max) | $(N-1)/\log N$ | ratio |
|---|---|---|---|
| 2 | 2 | 1.50 | 1.33 |
| 3 | 3 | 2.33 | 1.29 |
| 4 | 5 | 3.75 | 1.33 |
| 5 | 7 | 6.20 | 1.13 |
| 6 | 13 | 10.50 | 1.24 |
| 7 | 19 | 18.14 | 1.05 |
| 8 | 35 | 31.88 | 1.10 |
| 9 | 59 | 56.78 | 1.04 |
| 10 | 107 | 102.30 | 1.05 |
| 11 | 187 | 186.09 | 1.00 |
| 12 | 351 | 341.25 | 1.03 |
| 13 | 631 | 630.08 | 1.00 |
| 14 | 1181 | 1170.21 | 1.01 |
| 15 | 2191 | 2184.47 | 1.00 |
| 16 | 4115 | 4095.94 | 1.00 |
| 17 | 7711 | 7710.06 | 1.00 |
| 18 | 14601 | 14563.50 | 1.00 |
| 19 | 27595 | 27594.05 | 1.00 |
| 20 | 52487 | 52428.75 | 1.00 |

**Table 5:** A Comparison of maximum subtree sizes of the Balanced Spanning Tree and values of $(N-1)/\log N$.

## 4.2. The Complexity of Personalized Communication Based on the SBT and BST Topologies

### 4.2.1. Spanning Binomial Trees

For SBT-based distribution restricted to communication on one port at a time, the communication complexity for a maximum packet size of $B$ is $T \simeq (N-1)Mt_c+\tau(NM/B+\log\lceil\frac{B}{M}\rceil-1)$, $M \leq B \leq NM/2$, which is minimized for $B = NM/2$ yielding $T = (N-1)Mt_c + \tau\log N$. For $B \leq M$, $T \simeq (NM/B-1)(Bt_c+\tau)$. There exist several algorithms of this complexity. One such algorithm sends the cumulative data for the largest subtree to the root of this subtree first. Both nodes then recursively execute the same algorithm in their own $(n-1)$-subcube.

**Lemma 4.2.** *In distributing data from the root in a level-by-level order starting from level* $\log N$, *the time to complete the distribution is determined by the root, which terminates the distribution in a time proportional to the lower bound for a sufficiently large packet size.*

*Proof.* With potential concurrent communication on $\log N$ ports, the root can send data for level $\log N - i$ during step $i$, $0 \leq i \leq \log N - 1$, assuming a sufficiently large packet size. The amount of data sent from the root during step $i$ to the largest subtree is $\binom{\log N-1}{i}M$. The amount of data sent during the same step from the node at level $j$ to its largest subtree, which is the largest subtree at that level, is $\binom{\log N-1-j}{i-j}M$, where $1 \leq j \leq \log N-1$. Since $\binom{\log N-1}{i} \geq \binom{\log N-1-j}{i-j}$ for all $i,j \geq 1$, and any other subtree rooted at level $j$ is isomorphic to a subgraph of the highest subtree rooted at level $j$, the transfer time is determined by the transfer time of the root. For a sufficiently large packet size, i.e., $B \geq \binom{\log N-1}{\frac{\log N-1}{2}}M \approx \frac{NM}{\sqrt{2\pi(\log N-1)}}$, $T_{min} = \frac{1}{2}NMt_c + \log N\tau$.

∎

With a potential for concurrent communication on $\log N$ ports, a reduction in the transfer time by a factor of 2 is possible compared to communication on one port at a time. We conclude that

| Algorithm | $T_{min}$ |
|---|---|
| SBT, 1 port | $(N-1)Mt_c + \log N\tau$ |
| SBT, $\log N$ ports | $N/2Mt_c + \log N\tau$ |
| TCBT, 1 port | $\leq (2N - 2\log N - 1)Mt_c + (2\log N - 2)\tau$ |
| TCBT, $logN$ port | $(\frac{3}{4}N - 1)Mt_c + \log N\tau$ |
| BST, 1 port | $\leq N(1 + \frac{2\log\log N}{\log N})Mt_c + (2\log N - 2)\tau$ |
| BST, $\log N$ ports | $\simeq (N-1)/\log N Mt_c + \log N\tau$ |

**Table 6:** Communication complexity of personalized communication.

for SBT-based algorithms the packet size is of greater importance than concurrent communication on all ports.

### 4.2.2. Balanced Spanning Trees

With BST-based personalized communication restricted to one port at a time the root can send data to the subtrees cyclically. With a maximum packet size $B > M$, data for several nodes can be merged into one packet. The receiving node has sufficient time to retransmit pieces on all its ports, should that be required, since a new packet only arrives every $\log N$ cycles. The root requires a time of $T \simeq \frac{(N-1)M}{B\log N}(\tau + Bt_c)\log N$, which, if the data to the most remote nodes is transmitted first, is also the time to completion. For $B = M$, $T = (N-1)(\tau + Mt_c)$, i.e., the same as in the SBT algorithm. For $B \geq \frac{N}{\log N}M$ the root of the BST need only perform one communication per subtree, and it completes the communication in a time of $T = \tau\log N + (N-1)Mt_c$. But, unlike the SBT algorithm, the communication is not terminated when the root is done. The message to the last visited subtree needs to traverse $\log N - 2$ communication links. The bandwidth requirement of each subtree can be shown to be $\approx \frac{2N\log\log N}{\log N}$ [8]. An upper bound on the time for personalized communication based on the BST with unbounded packet size is $T = (2\log N - 2)\tau + N(1 + \frac{2\log\log N}{\log N})Mt_c$. The number of start-ups is almost twice that of the SBT-based personalized communication, and the total transfer time is higher by a lower order term. The time for personalized communication based on the BST is minimized for $B \geq \frac{N}{\log N}M$.

With a potential for communication on $\log N$ ports at a time, the time for personalized communication based on the BST topology is $T \simeq \frac{(N-1)M}{B\log N}(\tau + Bt_c)$ for $B \leq M$ and $T \simeq \sum_{i=1}^{\log N}\lceil\binom{\log N}{i}\frac{M}{B\log N}\rceil(\tau + B_it_c)$, $B_i = min(B, \lceil\binom{\log N}{i}\frac{M}{B\log N}\rceil)$ for $B > M$. This complexity estimate is valid if data for nodes at distance $i$ are sent during step $\log N - i$, $1 \leq i \leq \log N$. If $B = M$ then $T \simeq \frac{N-1}{\log N}(\tau + Mt_c)$. The communication time is minimized for $B \geq \frac{N}{\log^{3/2} N}M$ by using a *level-by-level* algorithm as in lemma 4.2. By property 3. of the BST, it follows that the amount of data sent from the root to any subtree during step $i$ is no less than the amount of data sent from any node during the same step. Hence, $T_{min} = \tau\log N + \frac{N-1}{\log N}Mt_c$, the minimum possible.

### 4.3. Comparison and Conclusion

With communication on one port at a time, if the fixed maximum packet size $B \leq M$ holds, then the complexity of SBT- and BST-based personalized communication is the same. For $B > M$, the SBT-based routing algorithm yields a lower complexity than the BST-based routing. For a sufficiently large maximum packet size the SBT-based algorithm has $\log N$ start-ups compared to $2\log N - 2$ start-ups for the BST-based algorithm. The transmission times are comparable, though the transmission time for the BST routing is higher. Note that, as in broadcasting, the minimum number of start-ups can be accomplished for sufficiently large maximum packet size.

With concurrent communication on $\log N$ ports the number of start-ups and the transmission time of BST-based routing is lower than that for the SBT by a factor of $\frac{1}{2}\log N$ for a maximum packet size $B \leq M$. With a sufficiently large packet size all routings yield a minimum of $\log N$ start-ups, but the BST routing has a total transmission time that is lower than that of SBT by a factor of $\frac{1}{2}\log N$. Moreover, it is achieved at a maximum packet size of $\frac{N}{\log^{3/2} N}M$, compared to a maximum packet size of $\frac{N}{\sqrt{2\pi(\log N-1)}}M$ for the SBT routing. We conclude that if communication can be performed on all $\log N$ ports concurrently, the communication complexity of the BST routing may be lower by a factor of $\frac{1}{2}\log N$ compared to the SBT routing. Table 6 lists the communication complexity for optimal packet size. The timing for TCBT routing is also included for easy comparison. See [8] for detailed analysis.

## 5. Experimental Results

### 5.1. Single Source Broadcasting Based on the SBT and MSBT

The Intel iPSC has a maximum packet size of $1k$ bytes. We refer to this packet size as the *internal* packet size and the user defined packet size as the *external* packet size. Figure 5 shows the measured time to completion of a broadcasting operation based on the SBT topology for cubes of various dimensions and a number of different external packet sizes. As expected, the communication time increases almost linearly for external packet sizes below $1k$ bytes. Figure 6 shows the measured time of SBT- and MSBT-based broadcasting for a message of $60k$ bytes with each packet being $1k$ bytes and for cube dimensions ranging from 2 to 6. Figure 7 shows the speed-up of broadcasting based on the MSBT topology over the SBT topology. The measured speed-up is approximately $\log N$, as predicted.

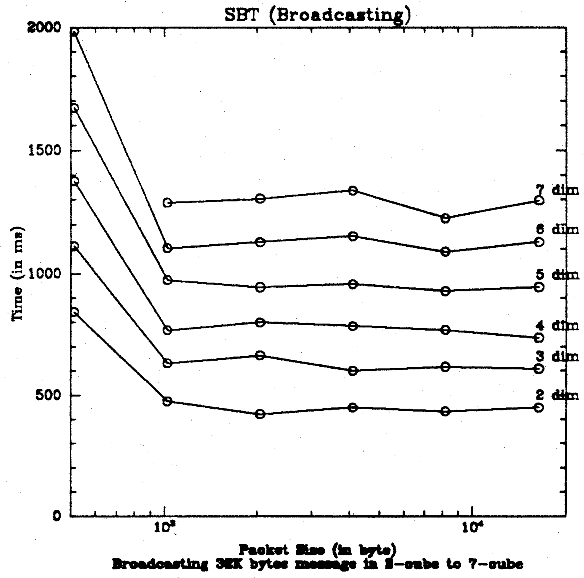

**Figure 5:** Broadcasting using SBT.

### 5.2. Single Source Personalized Communication

In implementing the SBT routing on the Intel *iPSC*, the root processes the data in descending order starting with the relative address $N-1$. This order implies that data is transmitted over ports in an order corresponding to the transition sequence in a binary-reflected Gray code[16]. Hence, port 0 is used every other cycle, port 1 every fourth cycle, etc. Internal nodes retransmit a

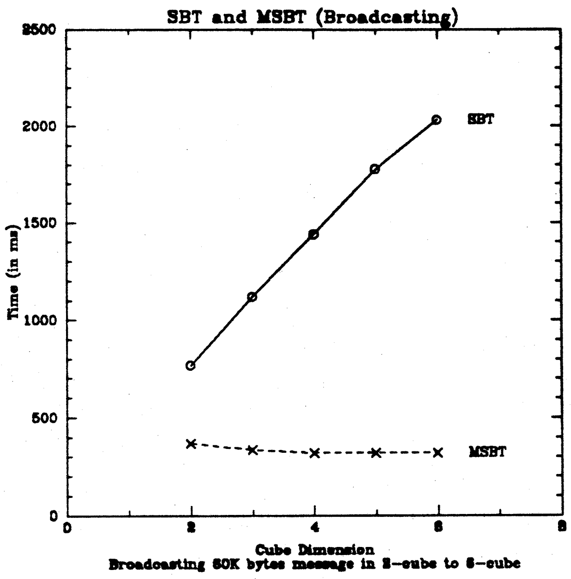

**Figure 6:** Broadcasting using SBT and MSBT.

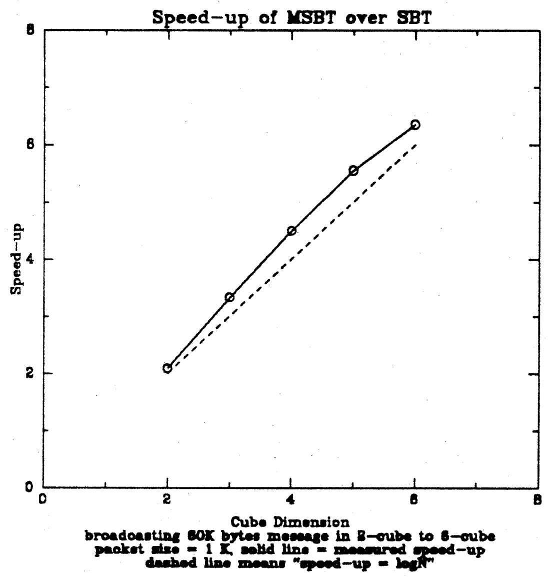

**Figure 7:** Speedup of MSBT vs SBT.

message on a port chosen from among the ones that correspond to the leading zeroes in its relative address. The choice is made according to a binary-reflected Gray code on the leading zeroes.

For the implementation of the BST-based algorithm the routing order needs to be determined for each subtree of the root. Excluding cyclic nodes, the subtrees are isomorphic. The root only needs to keep one table of length $\approx \frac{N}{\log N}$ with each entry of size $\log N$ bits. The order of the entries corresponds to the transmission order for each port. The table entry points to the messages transmitted over port 0. The pointers for the other ports are simply obtained by (right) cyclic shifts of the table entries. The cyclic nodes can be handled by finding the period $P$ for each cyclic table entry, and not transmitting the message corresponding to this table entry for ports with index $j \geq P$.

For each subtree a depth-first or a reversed breadth-first order are viable transmission orders. With reversed breadth-first order we mean a breadth-first traversal of the subtree starting from the last level ($\log N - 1$ or $\log N - 2$ depending on subtree). The source node determines the order, and internal nodes can either route according to the destination address if it is included, or by the use of tables. If tables are used, then in the case of depth-first communication order it suffices for each internal node to keep a count for each port. Since the number of ports used in each subtree is at most $\frac{\log N}{2}$, and the number of nodes in the entire subtree is approximately $\frac{N}{\log N}$, a bound on the table size in each node is $\log^2 N$ bits. A breadth-first communication order can be implemented by internal nodes keeping a table of how many nodes there are at a given level in each of its subtrees. The table has at most $\log^2 N$ entries. An upper bound for the number of nodes in a subtree at any level is $\frac{N}{\log^{3/2} N}$, and the total table size in a node is approximately $\log^3 N$ bits. Hence, without a more sophisticated encoding the depth-first communication order is more effective with respect to table space. The measurements presented in Figure 8 are based on an implementation using a depth first order.

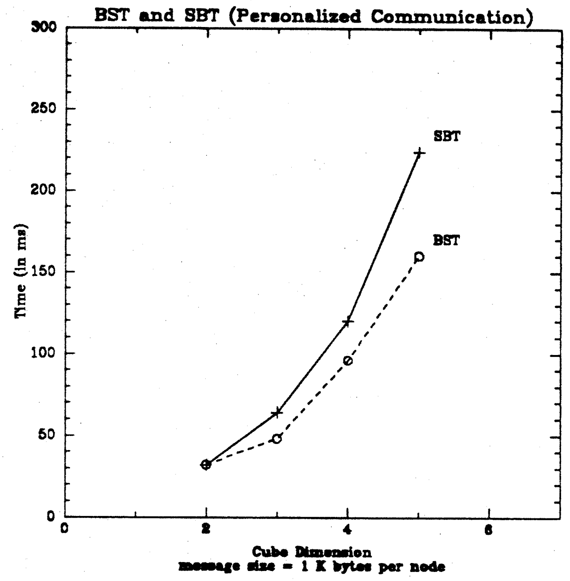

Figure 8: personalized Communication using BST and SBT.

With communication on one port at a time the expected time for personalized communication based on the SBT topology or the BST topology is the same. The observed advantage of the BST- over the SBT-based communication is due to the fact that the BST can take better advantage of the overlap between communication on different ports. In the SBT case, the node with relative address (00...01) is not yet finished retransmitting the last packet received when a new packet arrives. In the BST a subtree receives a packet once every $\log N$ cycles, and full advantage of the 20% overlap in communications actions is taken.

## 6. Conclusion

We have shown that the Boolean $n$-cube topology allows for the embedding of $n$ edge-disjoint binomial trees, and we presented routing algorithms for broadcasting that have a complexity equal to the lower bound both for communication restricted to one port at a time and for concurrent communication on all $n$ ports of a node. We have also defined a balanced spanning tree for personalized communication. Each subtree of the balanced spanning tree we have defined has approximately $\frac{N}{\log N}$ nodes. For communication on one port at a time, personalized communication based on the balanced spanning tree has the same complexity as personalized communication based on a binomial tree for certain maximum packet sizes, and has at most a factor of 2 higher complexity in other cases. With concurrent communication on all ports, the routing based on the balanced spanning tree is superior by a factor of $\frac{1}{2}\log N$ for a variety of combinations of maximum packet sizes, start-up times, transfer rates, and data sizes.

Experimental results on the Intel *iPSC/d7* confirm the results of the analysis.

## 7. Acknowledgement

Many thanks go to Andrea Pappas and Chris Hatchell for their assistance with the manuscript. The generous support of the Office of Naval Research under contract N00014-84-K-0043 is gratefully acknowledged.

## References

[1] A.V. Aho and J.E. Hopcroft and J.D. Ullman, Data Structures and Algorithms, Addison-Wesley, 1983, pp. 164–165.

[2] S. Bhatt and I. Ipsen, *How to Embed Trees in Hypercubes,* Technical Report YALEU/CSD/RR-443, Yale University, Dept. of Computer Science, December 1985.

[3] S.R. Deshpande and R.M. Jenevein, *Scaleability of a Binary Tree on a Hypercube,* Technical Report TR-86-01, University of Texas at Austin, January 1986.

[4] M.J. Fischer, Efficiency of Equivalence Algorithms, *Complexity of Computer Computations,* Plenum Press, 1972, pp. 153–167.

[5] G.C. Fox and S.W. Otto and A.J.G. Hey, *Matrix Algorithms on a Hypercube I: Matrix Multiplication,* Technical Report Hm 206, Caltech, October 1985.

[6] D. Gannon and J. Van Rosendale, *On the Impact of Communication Complexity in the Design of Parallel Numerical Algorithms,* IEEE Trans. Computers, C-33/12 December (1984), pp. 1180–1194.

[7] W.D. Hillis, *The Connection Machine,* MIT Press, 1985.

[8] C.-T. Ho and S.L. Johnsson, *Tree Embeddings and Optimal Routing for Data Distribution in Hypercubes,* Technical Report YALEU/DCS/RR-475, Department of Computer Science, Yale University, 1986.

[9] D. Hoey and C.E. Leiserson, A layout for the shuffle-exchange network, *Proc. 1980 International Conference on Parallel Processing,* IEEE Computer Society, August 1980.

[10] S.L. Johnsson, *Odd-Even Cyclic Reduction on Ensemble Architectures and the Solution Tridiagonal Systems of Equations,* Technical Report YALE/CSD/RR-339, Department of Computer Science, Yale University, October 1984.

[11] ———, *Communication Efficient Basic Linear Algebra Computations on Hypercube Architectures,* Technical Report YALEU/CSD/RR-361, Dept. of Computer Science, Yale University, January 1985.

[12] ———, *Solving Tridiagonal Systems on Ensemble Architectures,* SIAM J. Sci. Stat. Comp., (1986). Also available as Report YALEU/CSD/RR-436, November 1985.

[13] P.M. Kogge and H.S. Stone, *A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations,* IEEE Trans. Computers, C-22/8 (1973), pp. 786–792.

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

[15] O.A. McBryan and E.F. Van de Velde, Hypercube Algorithms and Implementations, *2nd SIAM Conference on Parallel Computing,* November 1985.

[16] E.M. Reingold and J. Nievergelt and N. Deo, *Combinatorial Algorithms,* Prentice Hall, 1977.

[17] Y. Saad and M.H. Schultz, *Data Communication in Hypercubes,* Technical Report YALEU/DCS/RR-428, Department of Computer Science, Yale University, October 1985.

[18] ———, *Topological Properties of Hypercubes,* Technical Report YALEU/DCS/RR-389, Department of Computer Science, Yale University, June 1985.

[19] C.L. Seitz, *The Cosmic Cube,* Communications of the ACM, 28/1 (1985), pp. 22–33.

[20] L.G. Valiant and G.J. Brebner, Universal Schemes for Parallel Communication, *Proc. of the 13th ACM Symposium on the Theory of Computation,* ACM, 1981, pp. 263–277.

[21] A.Y. Wu, *Embedding of Tree Networks,* Journal of Parallel and Distributed Computing, 2/3 (1985), pp. 238–249.