# Adaptive Path-Based Multicast on Wormhole-Routed Hypercubes*

*Chien-Min Wang*
*Institute of Information Science*
*Academia Sinica*
*Taipei, Taiwan, ROC*
*Email:cmwang@iis.sinica.edu.tw*

*Yomin Hou*
*Department of Computer and Information Science*
*National Chiao-Tung University*
*Hsinchu, Taiwan, ROC*
*Email: ymhou@cis.nctu.edu.tw*

*Lih-Hsing Hsu*
*Department of Computer and Information Science*
*National Chiao-Tung University*
*Hsinchu, Taiwan, ROC*
*Email: lhhsu@cis.nctu.edu.tw*

## Abstract

*In this paper, we consider the problem of path-based multicasting on wormhole-routed hypercubes. A minimum set of routing restrictions based on the strategy proposed by Li [4] is used for both unicast and multicast. To correctly perform multicast operations, we proposed the natural list approach to order the destination nodes. The proposed approach can be proved to be deadlock-free for both one-port and multi-port systems. Furthermore, it creates only one worm for each multicast operation, and provides adaptive shortest paths between each pair of nodes in the multicast path. The potential adaptivity provided by the proposed approach is superior to previous works. Between each pair of nodes in the multicast path with distance k, on the average, there are at least* $(k+1)!/2^k$ *paths. Unicast and broadcast can be treated as degenerated cases and use the same routing algorithm. Therefore, the proposed algorithm offers a comprehensive routing solution for communication on hypercubes.*

## 1. Introduction

*Multicast* is a collective communication service in which the same message is delivered from a source node to an arbitrary number of destination nodes on a parallel computer. Both *unicast*, which involves a single destination, and *broadcast*, which involves all nodes in the network, are special cases of multicast. Multicast communication has several uses in large-scale multiprocessors, including direct use in various parallel algorithms, implementation of data parallel programming operations, such as replication and barrier synchronization [13], and support of shared-data invalidation and updating in systems using a distributed shared-memory paradigm [5].

Wormhole routing has been widely adopted recently due to its effectiveness in inter-processor communication [1], [9]. With wormhole routing, each message is divided into a number of flits. The header flit(s) carries the address information and governs the route while the remaining flits of the message follow in a pipeline fashion. The pipelined nature provides two attractions. First, in the absence of channel contention, the network latency is relatively insensitive to the path length [9]. Second, only small flit buffers are required for each router [9].

To support multicast communication, different approaches had been considered in previous researches. Some of them used the unicast-based strategy to implement the multicast communication [8], [12]. In this strategy, if there are $k$ destination nodes, then $k$ unicasts will be generated to send the message. However, this strategy raises a problem that the increased traffic load resulted from those unicasts may hinder the system performance.

In order to minimize the amount of network traffic, some researches considered the *path-based* multicast routing. In this approach, a *multicast path* consists of a set of consecutive channels, starting from the source node and traversing each destination in the set. However, there are possible deadlocks due to dependencies on consumption channels in a path-based wormhole-routed network. Boppana *et al.* [2] showed how such a deadlock might happen and proposed the *column-path* routing algorithm to eliminate the problem for two dimensional (2D) meshes. For $k$-ary $n$-cubes, Panda *et al.* [10], [11] proposed a novel framework of a *base-routing-conformed- path* (BRCP) model. In these two algorithms, there may be several worms required for a multicast, and a multi-port model is necessary to perform these worms concurrently.

For multicasting on hypercubes, Lin [6] proposed the UD-path method to overcome the disadvantages of the BRCP model. The UD-path method is an extension of the deterministic path-based method [7] and creates only one worm for each multicast operation. Based on a node la-

* This work was supported by National Science Council, Republic of China, under Grant NSC90-2213-E-001-004.

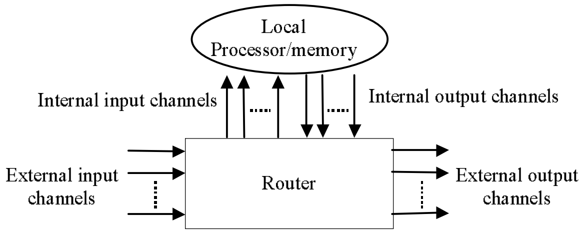



Fig. 1. The node architecture.

beling method, the author defined *up path* (*U-path*), *down path* (*D-path*), and *up-down path* (*UD-path*). For a unicast, all the UD-paths between the source and destination nodes can be used to send the message. For a multicast, the destination nodes are carefully ordered so that there is at least one multicast path, which is an UD-path and can be used to send the message. However, from a node to the next destination node in the multicast path, there are only U-paths or D-paths that can be used. Therefore, the routing algorithms for sending message between two nodes are different for unicast and multicast. Moreover, this approach required at least two input ports (consumption channels) to prevent deadlock.

In this paper, we proposed an adaptive multicast routing algorithm for wormhole-routed hypercubes. The proposed method is based on the adaptive unicast routing proposed by Li [4]. For each multicast operation, the proposed approach creates only one worm and provides adaptive shortest paths between each pair of nodes in the multicast path. All the multicasts, including unicasts and broadcasts, follow the same rules for routing. Therefore, the proposed algorithm offers a comprehensive routing solution for communication on hypercubes. It is proved that the proposed approach is deadlock-free and can be applied on either one-port or multi-port architecture. In performance analysis, the potential adaptivity of the proposed algorithm is compared with that of the UD-path algorithm. The result clearly shows the significant performance improvement provided by the proposed approach.

The rest of this paper is organized as follows. Section 2 introduces some background information. The new approach is described in section 3. In Section 4, the performance analysis is presented. Finally, conclusions are given in Section 5.

## 2. Background

Although the readers are assumed to be familiar with hypercube networks, it is still necessary to clarify the definitions and terminologies in order to have a well-defined model. The detail is specified in subsection 2.1. In subsection 2.2, we show the concept of the path-based routing. In subsection 2.3, the possible deadlocks in a path-based wormhole-routed network are introduced. Finally, in subsection 2.4, we show the adaptive unicast routing proposed by Li [4], which is the base routing algorithm of the proposed approach.

### 2.1 The wormhole-routed hypercube networks

An $n$-dimensional hypercube is a directed graph which contains $N = 2^n$ nodes and $n\times 2^n$ channels. Each node $x$ corresponds to an $n$-bit binary number, where $x(i)$ denotes the $i$th bit of $x$, $0\leq i\leq n-1$. If there are $k$ different bits between the binary strings of two node $x$ and $y$, then $k$ is said to be the *distance* of these two node. Two nodes are connected with a pair of channels, one for each direction, if and only if their distance is 1. If a channel is from node $x$ to $x'$ and $x(k) \neq x'(k)$, then it is said to be at dimension $k$ and denoted by $c_{x,x'}^{k}$. In case $x(k)=0$, then the channel is called *positive* and denoted by $c_{x,x'}^{k+}$, otherwise it is *negative* and denoted by $c_{x,x'}^{k-}$. The superscripts and subscripts may be omitted when they are irrelevant to the context.

In a wormhole-routed hypercube computer, communications are handled by *routers*, one for each node as shown in Fig. 1. The *external channels* connect the router to neighboring routers, and the *internal channels* connect to its local processor. The internal input channels are also known as *consumption channels*. The *port model* refers to the number of internal channels in each node. If each node possesses exactly one pair of internal input/output channels, the system is called a *one-port* architecture. In a one-port architecture, a local processor must transmit messages sequentially, and messages that are destined to the same node have to be received sequentially. If each node possesses more than one pair of internal input/output channels, the system is called a *multi-port* architecture. An *all-port* system is a special case of a multi-port system, in which every external channel has a corresponding internal channel. We shall use $c_i^{IN(j)}$ and $c_i^{OUT(j)}$ to denote the $j$th internal input and output channel of node $i$, respectively. If the one-port model is used, the number $j$ will be omitted.

### 2.2 Path-based routing

Lin *et al.* [7] have developed an approach to hardware-supported multicast, called *path-based routing*. In this approach, a *multicast path* for a source and a set of destinations consists of a set of consecutive channels, starting from the source node and traversing each destination in the set. The message sent by the source node may

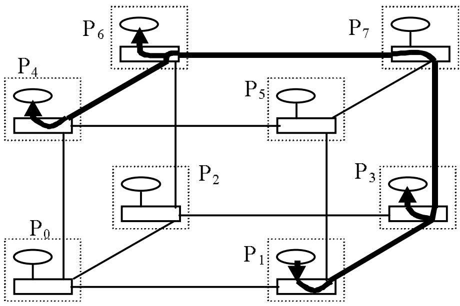



Fig. 2. A possible multicast path for <1: 3, 6, 4>.

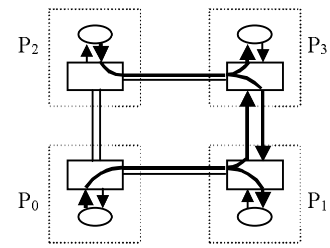
be replicated at intermediate destination nodes and forwarded along the multicast path to the next destination node.

A multicast operation can be denoted as ($s$, { $d_0$, $d_1$, $d_2$, …, $d_{r\text{-}1}$ }), where $s$ is the source node and {$d_0$, $d_1$, $d_2$, …, $d_{r\text{-}1}$} is the set of destination nodes. A *multicast list* for a multicast operation is an ordered list, which indicates the order of the destination nodes in the multicast path. We shall use $<s$: $d'_0$, $d'_1$, $d'_2$, …, $d'_{r\text{-}1}>$ to denote a multicast list for ($s$, {$d_0$, $d_1$, $d_2$, …, $d_{r\text{-}1}$}), where ($d'_0$, $d'_1$, $d'_2$, …, $d'_{r\text{-}1}$) is a permutation of ($d_0$, $d_1$, $d_2$, …, $d_{r\text{-}1}$). Fig. 2 shows a possible multicast path for the multicast list <1: 3, 6, 4> on a 3D hypercube.

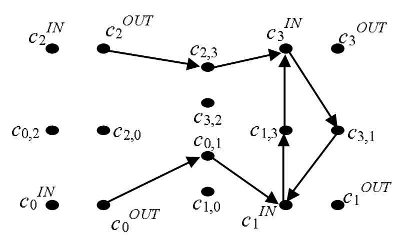
## 2.3 Deadlocks

Deadlock occurs when a message waits for an event that cannot happen. For example, a message may wait for a channel to be released by another message, which in turns is waiting for the first message to release some channel. When performing a multicast operation, if the header flit of the message arrives an intermediate destination node, the router will first reserve an internal input channel and then reserve an external output channel for the message. Therefore, in a path-based wormhole-routed network, all the internal and external channels could be involved in deadlocks. Fig. 3 shows two deadlock examples. The first one contains both external and internal input channels in the circular wait condition, while the other one contains only external channels. The one-port model is assumed in Fig. 3(a), and two multicast messages are waiting each other for internal input channels. One of these two multicast messages is from $P_0$ to $P_1$ and $P_3$, and the other one is from $P_2$ to $P_3$ and $P_1$. The channel-waiting graph for these two messages is shown in Fig. 3(b). The deadlock condition can be seen from the cycle in the graph. In Fig. 3(c), there are four multicast operations and the corresponding multicast lists are <0: 3, 7>, <2: 1, 5>, <3: 0, 4>, and <1: 2, 6>. Fig. 3(d) shows the cycle in the channel-waiting graph for these four multicasts. The channels of the nodes $P_4$, $P_5$, $P_6$, and $P_7$ are omitted in Fig. 3(d). It can be observed that the channels involved in the circular wait condition are all external channels. The following lemma gives an important property in a channel-waiting cycle.

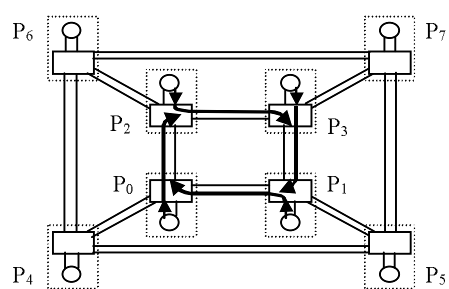
**Lemma 1**: Given an arbitrary circle in a hypercube, there must be an equal number of positive channels and negative channels of the same dimension in the circle [4].



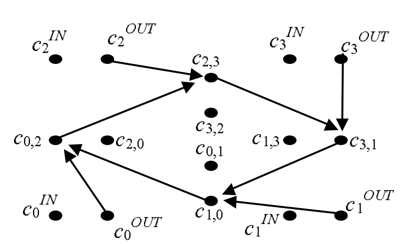

(a) Deadlock in a one-port system.




(b) Channel-waiting graph of (a).




(c) Deadlock in a 3D hypercube.




(d) Part of Channel-waiting graph of (c).

Fig. 3. Two deadlock examples.

### 2.4 The MIN routing strategy for unicast

The strategy proposed by Li [4] presents a minimum set of deadlock-free routing restrictions for unicasts on hypercubes. The routing restrictions mean the rules that specify which channels ending at a node can forward messages to certain channels, called *legal channels*, starting from the node. Every routing algorithm implies a set of restrictions. For example, the *e*-cube routing allows a channel to forward messages to only those channels that are at higher dimensions. It establishes only one shortest path between each pair of source-destination nodes. In order to take the advantage of the flexibility provided by hypercubes, various adaptive unicast routing algorithms have also been proposed [3], [9]. The following set of restrictions is the minimum set presented in [4].

***Restriction 1.*** Messages can be forwarded from $c^{l}_{x,y}$ to $c^{m}_{y,z}$ on the node $y$ if and only if one or both of the following conditions are true:

(1) $m > l$, and (2) $c^m$ is positive.

That is, messages can be forwarded from a lower dimensional channel to a higher dimensional channel (condition 1), or from a higher dimensional channel to a lower dimensional channel if the latter is positive (condition 2). The following theorems show that *Restriction 1* is a minimum set of restrictions and the unicast routing algorithms under *Restriction 1* are deadlock-free.

**Theorem 1**: The unicast routing algorithms under *Restriction 1* are deadlock-free [4].

**Theorem 2**: Any relaxation to *Restriction 1* will result in deadlock in the hypercube [4].

## 3. The Multicast Routing Strategy

The most fundamental requirement for an efficient path-based multicast routing algorithm is to avoid deadlock. Next, it is expected that the degenerate cases such as unicast and broadcast also use the same algorithm, thereby offering a comprehensive routing solution. Third, the multicast algorithm should always follow a shortest path between each pair of nodes in the multicast path. Thus, a unicast message routed according to the algorithm will always follow a shortest path. Finally, the multicast algorithm should take the flexibility of hypercubes to provide more paths between each pair of nodes in the multicast path.

Upon developing an efficient path-based multicast routing algorithm, two issues have to be considered to achieve the above requirements.

1. How to order the destination nodes, i.e. how to decide the multicast list for the multicast operation?
2. How to route the multicast message corresponding to the specified multicast list?

In this section, we will first propose the routing restrictions for both unicast and multicast. Then, we will show the method to construct a multicast list for each multicast operation so that no deadlock may be incurred.

### 3.1 Routing restrictions

In this subsection, we shall show an equivalent set of restrictions to *Restriction 1*, and take it as the base adaptive routing algorithm in the proposed multicast routing strategy.

***Restriction 2.*** Messages can be forwarded from $c^{l}_{x,y}$ to $c^{m}_{y,z}$ on the node $y$ if and only if one or both of the following conditions are true:

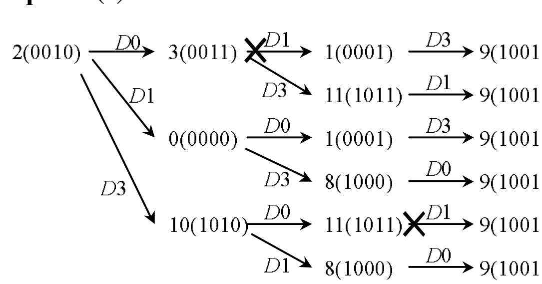
(1) $m < l$, and (2) $c^m$ is positive.

The only difference between *Restriction 1* and *Restriction 2* is that, in *Restriction 2*, messages cannot be forwarded from a lower dimensional channel to a negative higher dimensional channel; while, in *Restriction 1*, messages cannot be forwarded from a higher dimensional channel to a negative lower dimensional channel. The routing algorithms under *Restriction 2* can also be proved to be deadlock-free for unicasts similarly.

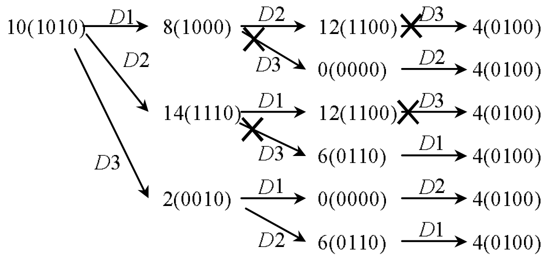
**Theorem 3**: The routing algorithms under *Restriction 2* are deadlock-free [4].

The following example shows the paths for two unicasts. Some of the paths are marked to be unavailable under *Restriction 2*. Note that, from *Restriction 2*, if the required channel at the highest dimension is negative then it has to be selected to forward the message first. Otherwise, the message will be blocked forever due to no legal channel exists. This can be verified in Example 1(b). From node 10 to node 4, the required channels are at dimensions $D_1$, $D_2$, and $D_3$, and the one at $D_3$ is negative. It can be observed that only the paths passing $D_3$ first are legal under *Restriction 2*.

**Example 1**. (a) Paths from node 2 to node 9.



(b) Paths from node 10 to node 4.




## 3.2 The multicast list

For a multicast operation, if the number of destination nodes is $r$, then there are $r!$ multicast lists. However, not every multicast list can be performed correctly under the base routing algorithm because the corresponding multicast path may not exist. Fig. 4 shows such an example based on $e$-cube routing. The multicast list is <0: 7, 6>. We can see that, following the order, the message is first sent from $P_0$ to $P_7$ through the path ($c_{0,1}$, $c_{1,3}$, $c_{3,7}$). Then, when trying to forward the message from $P_7$ to $P_6$, there is no legal channel because $c_{3,7}$ is at dimension 2 while $c_{7,6}$ is at dimension 0, and the messages can not be forwarded from $c_{3,7}$ to $c_{7,6}$ under $e$-cube routing. A multicast list is said to be *legal* if there exists at least one legal channel to forward the message at any time; otherwise it is *illegal*.

In this subsection, we shall give a method to construct a legal multicast list for every multicast operation, and show the proposed approach will not incur deadlock. Consider the multicast operation ($s$, $\{d_0, d_1, d_2, \ldots, d_{r\text{-}1}\}$). The set of destination nodes $\{d_0, d_1, d_2, \ldots, d_{r\text{-}1}\}$ can be sorted increasingly so that we can obtain a permutation ($d'_0$, $d'_1$, $d'_2$, …, $d'_{r\text{-}1}$) of $\{d_0, d_1, d_2, \ldots, d_{r\text{-}1}\}$, where $d'_i<d'_{i+1}$ and $0\le i<r$. A multicast list can be constructed to be $<s: d'_0, d'_1, d'_2, \ldots, d'_{r\text{-}1}>$ for ($s$, $\{d_0, d_1, d_2, \ldots, d_{r\text{-}1}\}$). Such a multicast list is called the *natural list*. Before showing that the natural list is a legal multicast list under *Restriction 2*, let's see the example illustrated in Fig. 5. In Fig. 5, the multicast (0, {3, 6, 7}) is performed by using the natural list under *Restriction 2*. Between the source node $P_0$ and the first destination node $P_3$, there are two available paths, ($c_{0,1}$, $c_{1,3}$) and ($c_{0,2}$, $c_{2,3}$). If ($c_{0,1}$, $c_{1,3}$) is selected to send the message, then there are two available paths, ($c_{3,2}$, $c_{2,6}$) and ($c_{3,7}$, $c_{7,6}$), for forwarding the message from $P_3$ to $P_6$. On the other hand, if ($c_{0,2}$, $c_{2,3}$) is selected to send the message, then there is only one available path, ($c_{3,7}$, $c_{7,6}$). The reason is that $c_{2,3}$ and $c_{3,2}$ are both at dimension 0, and $c_{3,2}$ is negative. Therefore, it is not allowed to forward the message from $c_{2,3}$ to $c_{3,2}$. The last channel in the multicast path is $c_{6,7}$. Since $c_{6,7}$ is a positive channel, it is always legal for forwarding the message from $P_6$ to $P_7$. In the following theorems, we will show that the natural list is legal under *Restriction 2*, and that the natural list approach is deadlock-free.

**Theorem 4**: The natural list for ($s$, $\{d_0, d_1, d_2, \ldots, d_{r\text{-}1}\}$) is a legal multicast list under *Restriction 2.*

**Proof**: Suppose $<s: d'_0, d'_1, d'_2, \ldots, d'_{r\text{-}1}>$ is the natural list for ($s$, $\{d_0, d_1, d_2, \ldots, d_{r\text{-}1}\}$). For the following two reasons, there exists at least one legal channel under *Restriction 2* to forward the message at any time

(1) The multicast message can be sent from $s$ to $d'_0$ under *Restriction 2* as an unicast. Therefore, there exists at least one legal channel to forward the message from $s$ to $d'_0$ at any time.

(2) Consider any ($d'_i$, $d'_{i+1}$), $0\le i<r\text{-}1$, in the natural list. Because $d'_i< d'_{i+1}$, the highest dimensional channel of $d'_i$ that can be selected to forward the message to $d'_{i+1}$ must be positive. Let the channel be denoted by $c^h$. When the multicast message arrives $d'_i$, no matter which channel it comes from, $c^h$ can be selected to forward the message. Hence, there is at least one legal channel that can be selected as the first channel to forward the message from $d'_i$ to $d'_{i+1}$ under *Restriction 2*. After the first channel is selected, *Restriction 2* can be used continually so that the message can be sent to $d'_{i+1}$ correctly. Therefore, there exists at least one legal channel to forward the message from $d'_i$ to $d'_{i+1}$ at any time.

From the above reasons, The natural list for ($s$, $\{d_0, d_1, d_2, \ldots, d_{r\text{-}1}\}$) is a legal multicast list under *Restriction 2.*

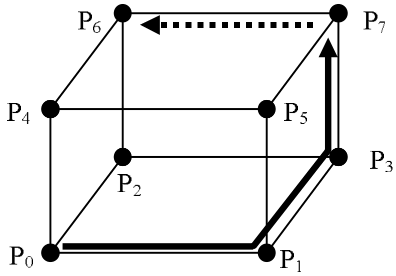



Fig. 4. An illegal multicast list <0: 7, 6> under *e*-cube routing.

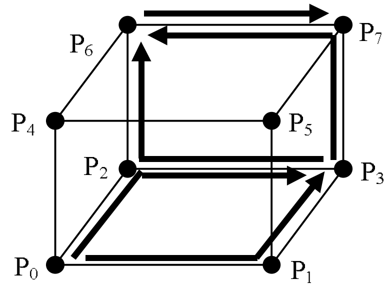



Fig. 5. The paths for the natural list <0: 3, 6, 7> under *Restriction 2.*

**Lemma 2**: In the channel-waiting graph of the natural list approach, there is no external channel waiting for a negative channel in the same or a higher dimension.

**Proof**: From Theorem 4, the natural list for a multicast is a legal multicast list under *Restriction 2*. Hence, all the channels in the multicast paths for the natural list conform to *Restriction 2*. Suppose $c^l$ and $c^m$ are two channels in the multicast paths. If $c^l$ waits for $c^m$ and $l \leq m$, then $c^m$ must be positive. Therefore, in the channel-waiting graph of the natural list approach, there is no external channel waiting for a negative channel in the same or a higher dimension.

**Theorem 5**: The natural list approach for multicast is deadlock-free under *Restriction 2*.

**Proof**: In the following proof, we assume that the target system is a one-port system. Since a multi-port system provides more internal channels than the one-port system, it can be directly derived that the following proof is also true for a multi-port system.

Let the external channels of the $n$-dimensional hypercubes be grouped into $n$ subsets, $C^h$, $0 \leq h < n$, where $C^h$ contains all the channels in dimension $h$. The proof is by induction on $h$ to show that there is no channel in $C^h$ involved in a deadlock.

(1) Consider $h=n$-1. If there is any channel in $C^{n-1}$ involved in a deadlock, a cycle can be found in the channel-waiting graph. From Lemma 1, there must exists a negative channel $c^{(n-1)-}$ involved in the channel-waiting cycle. For the following reasons, such a cycle does not exist. Hence, there is no channel in $C^{n-1}$ involved in a deadlock.
   (a) From *Restriction 2* and Lemma 2, there is no channel $c^l$, $l \leq n$-1, waiting for $c^{(n-1)-}$. Therefore, there is no external channel waiting for $c^{(n-1)-}$.
   (b) Because of the natural list approach, the address of the next destination must be larger than that of the current node. Hence, there is no $c^{(n-1)-}$ required for forwarding the message. Therefore, there is no internal input channel waiting for $c^{(n-1)-}$.
   Since no channel will wait for $c^{(n-1)-}$, $c^{(n-1)-}$ must not be involved in any cycle. Thus, no channel in $C^{n-1}$ is involved in a deadlock.

(2) Suppose there is no channel in $C^h$, $k \leq h \leq n$-1, involved in a deadlock.

(3) We shall prove that there is no channel in $C^h$, $h=k$-1, involved in a deadlock. Suppose there is a deadlock, from (1) and (2), all the channels in the channel-waiting cycle corresponding to the deadlock are in the dimensions less than $k$. From Lemma 1, if there is any channel in $C^{k-1}$ involved, there must exists a negative channel $c^{(k-1)-}$ involved in the channel-waiting cycle. From the same reason shown in (1)(a), there is no channel $c^l$, $l \leq k$-1, waiting for $c^{(k-1)-}$. Thus, the only possible condition is that some internal input channel $c_j^{IN}$ waiting for $c^{(k-1)-}$ in the cycle. Because of the natural list approach, the address of the next destination must be larger than that of the current node. Therefore, if $c_j^{IN}$ waits for $c^{(k-1)-}$, there must exist a positive channel $c^{h'+}$, $h' \geq k$, that could be used to forward the message. Since $c^{h'+}$ is not involved in any deadlock, the message can be forwarded through $c^{h'+}$ eventually, and $c_j^{IN}$ can stop waiting for $c^{(k-1)-}$. For this reason, no channel will wait for $c^{(k-1)-}$ forever, and no cycle containing $c^{(k-1)-}$ will cause deadlock.

From the above discussion, it can be proved that the natural list approach for multicast operation is deadlock-free under *Restriction 2*.

## 4. Performance Analysis

In order to show the advantage of the proposed approach, we analyze its potential adaptivity and compare the result with that of the UD-path method [6]. The potential adaptivity is measured by the average numbers of paths from the source node to the first destination node, and from a destination node to the next destination node. The average number of paths from the source node to the first destination node is equivalent to the average number of paths for all unicasts. That will be discussed in subsection 4.1. In subsection 4.2, we analyze the number of paths from a destination node to the next destination node in a natural list.

### 4.1 The average number of paths for unicasts

Suppose, in a unicast ($s$, $d$), the distance between $s$ and $d$ is $k$. To compute the average number of paths between $s$ and $d$, we may assume that ($s$, $d$) is a pair of antipodal nodes in a $k$-dimensional hypercube. The number of paths for each pair ($s$, $d$) is depends on the addresses of them. There are $2^k$ possible combinations of addresses for $s$ and $d$. The set of all these combinations can be denoted by

$S_k$ = {($s$, $d$)| ($s$, $d$) is a pair of antipodal nodes in a $k$-dimensional hypercube}.

$S_k$ can be divided into two subsets $S_k^+$ and $S_k^-$ by the sign of the highest dimensional channel $c^{k-1}$ required for sending message from $s$ to $d$, i.e.,

$S_k^+$ = {($s$, $d$)| ( $s$, $d$ )∈ $S_k$ and $c^{k-1}$ is positive}, and

$S_k^-$ = {($s$, $d$)| ( $s$, $d$ )∈ $S_k$ and $c^{k-1}$ is negative}.

Corresponding to $S_k$, $S_k^+$, and $S_k^-$, three sets of paths are defined as follows.

$P_k$= {all the paths for all ($s$, $d$)∈$S_k$},

$P_k^+$= {all the paths for all ($s$, $d$)∈$S_k^+$}, and

$P_k^-$= {all the paths for all ($s$, $d$)∈$S_k^-$}.

Let $t_k$ be the cardinality of $P_k$, $t_k^+$ be the cardinality of $P_k^+$, and $t_k^-$ be the cardinality of $P_k^-$. Obviously, $t_k = t_k^+ + t_k^-$.

For each path in $P_k^+$, since $c^{k-1}$ is positive, it is a legal channel and can be selected at any time. Therefore, we can derive that $t_k^+ = k \times t_{k-1}$. For each path in $P_k^-$, since $c^{k-1}$ is negative, it must be the first channel to be passed. Hence, $t_k^- = t_{k-1}$. From the above reasons,

$$t_k = t_k^+ + t_k^- = k \times t_{k-1} + t_{k-1} = (k+1) \times t_{k-1} \text{ and } t_1 = 2. \Rightarrow$$

$$t_k = (k+1)!. \quad (4.1)$$

Therefore, the average number, $e_k$, of paths from $s$ to $d$ with distance $k$ is as follows.

$$e_k = \frac{(k+1)!}{2^k}. \quad (4.2)$$

## 4.2 Paths from a destination node to the next destination node

Let $<s: d'_0, d'_1, d'_2, \ldots, d'_{r-1}>$ be the natural list for $(s, \{d_0, d_1, d_2, \ldots, d_{r-1}\})$. Now we shall measure the average number of paths from a destination node $d'_i$ to the next destination node $d'_{i+1}$ in the natural list. The number of paths from $d'_i$ to $d'_{i+1}$ not only depends on the addresses of them but also on the external input channel $c^e$ of $d'_i$, which the message is forwarded through to arrive $d'_i$. Without considering the impact of $c^e$, the number of paths from $d'_i$ to $d'_{i+1}$ can be analyzed as in the above subsection. Suppose the distance between $d'_i$ and $d'_{i+1}$ is $k$. Similarly, we may assume that $(d'_i, d'_{i+1})$ is a pair of antipodal nodes in a $k$-dimensional hypercube. Note that the highest dimensional channel required for sending message from $d'_i$ to $d'_{i+1}$ must be positive because $d'_i < d'_{i+1}$ in the natural list approach. Hence, there are $2^{k-1}$ possible combinations for $d'_i$ and $d'_{i+1}$. Let

$S'_k = \{(d'_i, d'_{i+1}) \mid (d'_i, d'_{i+1})$ is a pair of antipodal nodes in a $k$-dimensional hypercube and $d'_i < d'_{i+1}\}$,

$P'_k = \{$all the paths for all $(d'_i, d'_{i+1}) \in S'_k\}$, and

$t'_k =$ the cardinality of $P'_k$.

We can derive that

$$t'_k = k \times t_{k-1} \Rightarrow t'_k = k \times k! \quad (4.3)$$

Therefore, the average number, $e'_k$, of paths from $d'_i$ to $d'_{i+1}$ with distance $k$ must be less than or equal to $t'_k/2^{k-1}$, i.e.,

$$e'_k \le t'_k/2^{k-1} = \frac{k \times k!}{2^{k-1}}. \quad (4.4)$$

The impact of $c^e$ will be discussed in the following paragraphs. First, consider the $P_k$ defined in subsection 4.1. $P_k$ can be divided into two subsets $P_k^P$ and $P_k^N$ by the sign of the first channel of each path in $P_k$, i.e.,

$P_k^P = \{p \mid p \in P_k$ and the first channel of $p$ is positive$\}$, and

$P_k^N = \{p \mid p \in P_k$ and the first channel of $p$ is negative$\}$.

Let $t_k^P$ be the cardinality of $P_k^P$ and $t_k^N$ be the cardinality of $P_k^N$, then

$$t_k = t_k^P + t_k^N. \quad (4.5)$$

Since $P_k = P_k^+ \cup P_k^- = P_k^P \cup P_k^N$, we can derive

$$P_k^N = (P_k^N \cap P_k^+) \cup (P_k^N \cap P_k^-), \text{ and}$$
$$((P_k^N \cap P_k^+) \cap (P_k^N \cap P_k^-)) = \phi. \quad (4.6)$$

From the definitions of $P_k^N$, $P_k^-$, and $P_k^+$, it can be derived that $P_k^- \subset P_k^N$, and the cardinality of $(P_k^N \cap P_k^+)$ is $(k-1) \times t_{k-1}^N$. Therefore, from (4.1) and (4.6),

$$t_k^N = (k-1) \times t_{k-1}^N + t_k^- = (k-1) \times t_{k-1}^N + t_{k-1} = (k-1) \times t_{k-1}^N + k!$$
and $t_1^N = 1. \Rightarrow$

$$t_k^N = (k+1)!/2. \quad (4.7)$$

Similar to the definitions of $P_k^P$ and $P_k^N$, we defined $P'^P_k$ and $P'^N_k$ as follows.

$P'^P_k = \{p \mid p \in P'_k$ and the first channel of $p$ is positive$\}$, and

$P'^N_k = \{p \mid p \in P'_k$ and the first channel of $p$ is negative$\}$.

Let $t'^P_k$ be the cardinality of $P'^P_k$ and $t'^N_k$ be the cardinality of $P'^N_k$. Then

$$t'_k = t'^P_k + t'^N_k. \quad (4.8)$$

From the definitions of $P'^N_k$ and (4.7), it can be derived that

$$t'^N_k = (k-1) \times t_{k-1}^N \Rightarrow t'^N_k = (k-1)k!/2. \quad (4.9)$$

Therefore, from (4.3), (4.8) and (4.9),

$$t'^P_k = t'_k - t'^N_k = k \times k! - (k-1)k!/2 = (k+1)!/2. \quad (4.10)$$

No matter which $c^e$ is, all the path in $P'^P_k$ is available because the first channel of each path in $P'^P_k$ is positive. Therefore, the average number, $e'_k$, of paths from $d'_i$ to $d'_{i+1}$ with distance $k$ must be larger than or equal to $t'^P_k/2^{k-1}$, i.e.,

$$e'_k \ge t'^P_k/2^{k-1} = \frac{(k+1)!}{2^k}. \quad (4.11)$$

Table 1. Average number of paths between nodes.

| Distance | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| UD-Path | 1.00 | 1.00 | 1.50 | 3.00 | 7.50 | 22.50 | 78.75 | 315.00 | 1417.50 | 7087.50 |
| Natural List (Lower Bound) | 1.00 | 1.50 | 3.00 | 7.50 | 22.50 | 78.75 | 315.00 | 1417.50 | 7087.50 | 38981.25 |
| Natural List (Upper Bound) | 1.00 | 2.00 | 4.50 | 12.00 | 37.50 | 135.00 | 551.25 | 2520.00 | 12757.50 | 70875.00 |

From (4.2), (4.4), and (4.11), the average number of paths, $E_k$, between two nodes with distance $k$ is bounded by the following formula.

$$\frac{(k+1)!}{2^k} \le E_k \le \frac{k \times k!}{2^{k-1}} \qquad (4.12)$$

This result is compared with that of the UD-path method, and shown in Table 1. The first row of Table 1 gives the distance between two nodes. The average numbers of paths for the UD-path method is given in the second row. In the third and fourth rows, the analysis result for the proposed approach is presented. It can be observed from the table that the average number of paths for the proposed approach is at least $(k+1)/2$ times than that of the UD-path method. It is of great improvement on the potential adaptivity.

## 5. Conclusions

In this paper, we consider the problem of path-based multicasting on wormhole-routed hypercubes. A minimum set of routing restrictions based on the strategy proposed by Li [4] is given for both unicast and multicast. These restrictions prevent a message to be forwarded from a lower dimensional channel to a negative higher dimensional channel. Hence, no deadlock will be incurred for unicasts in the routing algorithms. To perform the multicast operations correctly, we propose the natural list approach to order the destination nodes in each multicast operation. The proposed approach can be proved to be deadlock-free for both one-port and multi-port systems. Furthermore, it creates only one worm for each multicast operation, and provides adaptive shortest paths between each pair of nodes in the multicast path. The potential adaptivity of the proposed algorithm is compared with that of the UD-path algorithm. The result clearly shows the significant improvement provided by the proposed approach. Between each pair of nodes in the multicast path with distance $k$, the average number of paths, $E_k$, is bounded to be $\frac{(k+1)!}{2^k} \le E_k \le \frac{k \times k!}{2^{k-1}}$. Unicast and broadcast can be treated as degenerated cases and use the same algorithm. Therefore, the proposed algorithm offers a comprehensive routing solution for communication on hypercubes.

## References


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

[2] R. Boppana, S. Chalansani, and C. S. Raghavendra, “On multicast wormhole routing in multicomputer networks,” *Proceedings of the 6th IEEE Symposium on Parallel and Distributed Processing*, pp. 722-729, Apr. 1994.

[3] C. J. Glass and L. M. Ni, “The Turn Model for Adaptive Routing,” *Journal of the Association for Computing Machinery*, vol. 41, no. 5, pp. 874-902, Sep. 1994.

[4] Q. Li, “Minimum Deadlock-Free Message Routing Restrictions in Binary Hypercubes,” *Journal of Parallel and Distributed Computing*, vol. 15, no. 2, pp. 153-159, 1992.

[5] K. Li and R. Schaefer, “A hypercube shared virtual memory,” *Proc. 1989 International Conference on Parallel Processing*, vol. 1, pp. 125-132, 1989.

[6] X. Lin, “Adaptive Wormhole Routing in Hypercube Multicomputers,” *Journal of Parallel and Distributed Computing*, vol. 48, no. 2, pp. 165-174, 1998.

[7] X. Lin, P. K. McKinley and L. M. Ni, “Deadlock-Free Multicast Wormhole Routing in 2-D Mesh Multicomputers,” *IEEE Transaction on Parallel and Distributed Systems*, Vol. 5, No. 8, pp. 793-804, 1994.

[8] P. K. McKinley, H. Xu, A-H. Esfahanian and L. M. Ni, “Unicast-Based Multicast Communication in Wormhole-Routed Networks,” *IEEE Transaction on Parallel and Distributed Systems*, Vol. 5, No. 12, pp. 1252-1265, Dec. 1994.

[9] L. M. Ni and L. M. McKinley, “A Survey of Wormhole Routing Techniques in Direct Networks,” *IEEE Computer*, vol. 26, pp. 62-76, Feb. 1993.

[10] D. K. Panda, S. Singal, and P. Prabhakaran, “Multidestination Message Passing Mechanism Conforming to Base Wormhole Routing Scheme,” *Parallel Computer Routing and Communication: First International Workshop, PCRCW’94*, Lecture Notes in Computer Science 853, Springer-Verlag, pp. 131-145, 1994.

[11] D. K. Panda, S. Singal, and R. Kesavan, “Multidestination Message Passing in Wormhole *k*-ary *n*-cube Networks with Base Routing Conformed Paths,” *IEEE Transaction on Parallel and Distributed Systems*, vol. 10, no. 1, pp. 76-96, Jan. 1999.

[12] David F. Robinson, Philip K. McKinley and Betty H.C. Cheng, “Optimal Multicast Communication in Wormhole-Routed Torus Networks,” *IEEE Transaction on Parallel and Distributed Systems*, Vol. 5, No. 12, Dec. 1994, pp. 1252-1265.

[13] H. Xu, P. K. Mckinley, and L. M. Ni, “Efficient implementation of barrier synchronization in wormhole-routed hypercube multicaticomputers,” *Journal of Parallel and Distributed Computing*, vol. 16, no. 2, pp. 172-184, 1992.