An Optimal $d-1$-Fault-Tolerant Broadcasting Scheme for d-Dimensional Hypercubes

Siu-Cheung Chau[1]
Dept. of Mathematics and Computer Science, University of Lethbridge,
Lethbridge, Alberta, Canada, T1K 3M4

Abstract

Broadcasting is the process of information dissemination in a communication network in which a message, originated by one member, is transmitted to all members of the network. We propose a k-fault-tolerant broadcasting scheme for a faulty d-dimensional hypercube where $0 \le k < d$. For the whispering model, the new scheme requires $2d$ time units that are optimal for faulty-hypercubes. Moreover, it only requires $2d-1$ time units for non-faulty hypercubes. For the shouting model, the new scheme requires $d+1$ time units that are optimal for faulty-hypercubes and d time units for non-faulty hypercubes. The new algorithm is simpler, faster, and requires $n-d-1$ fewer message transmissions than previously proposed optimal k-fault-tolerant broadcast schemes.

keywords: Hypercube, Fault-tolerant, Broadcasting

---

[1]supported by the National Sciences and Engineering Research Council of Canada

Proc. ICCI'94, 493-508 © 1994 Int. Conf. on Computing and Information

1 Introduction

Broadcasting is the process of information dissemination in a communication network in which a message, originated by one member, is transmitted to all members of the network. A k-fault-tolerant broadcast is to broadcast with enough redundancy so that the broadcast can be completed even if k links or nodes fail. The broadcast is carried out by a series of calls between nodes.

Two different models of communication, shouting and whispering [3] are considered. In the shouting model, a node can communicate simultaneously with all the adjacent nodes. In the whispering model, a node can only communicate with one adjacent node at any given time.

In this paper, we assume that the communications are based on a message passing procedure and are sent in the store and forward mode. We only consider permanent node and link faults and the originator has no knowledge of the location of these faults. Furthermore, a node or a link is faulty if it cannot transmit any messages. It cannot corrupt messages.

Two important parameters in broadcasting are the time units, and the number of calls required to complete the broadcast. Given a communication network with n nodes, for the whispering model, 0-fault-tolerant broadcasting requires at least $\log_2 n$ time units. K-fault-tolerant broadcasting requires at least $\log_2 n+k$ or $\log_2 n+k+1$ time units depending on n and the topology of the network [6, 4].

In this paper, we will concentrate on fault-tolerant broadcasting in multi-computer networks connected in the form of hypercubes. The hypercube network is already commercially available and is being used for a variety of applications. Ramanathan and Shin [9] considered broadcasting in faulty hypercubes. They proposed a $(d-1)$-fault-tolerant broadcasting scheme for a d-dimensional hypercube. In their scheme, a node can send a message to an

adjacent node and receive messages from other adjacent nodes at the same time unit. Their scheme requires $d+1$ time units for a $(d-1)$-fault-tolerant broadcast for the shouting model. The number of messages required is nd. Their scheme is optimal in the number of time units required for the shouting model. However, for the whispering model, their scheme cannot be applied directly due to their assumptions.

Carlsson et. al. [2] generalized Ramanathan and Shin's broadcast algorithm. A node can send t messages and receive any number of messages at the same time unit. Their algorithm can complete a k-fault-tolerant broadcast in $d+\lceil (k+1)/t \rceil$ time units. Their algorithm is optimal for the shouting model but it cannot be applied directly to the whispering model due to their assumptions.

Bruck [1] considered fault-tolerant broadcast in hypercubes that can tolerate up to $\lceil d/2 \rceil - 1$ edge faults. If the originator knows the location of the edge faults, his algorithm can broadcast in d time units that are the same as a non-fault-tolerant broadcast. However, the number of time units increases if the originator does not know the link fault locations. Moreover, his method can only deal with link faults.

Gargano, Rescigno, and Vaccaro [5] also considered broadcasting in hypercubes with the presence of edge faults. Instead of sending out $k+1$ copies of the message, their scheme uses Rabin's [8] information dispersal algorithm to send out the message in $m+k$ pieces. A node can reconstruct the message if it receives m pieces of the message. They compared their scheme with schemes that send out $k+1$ copies of the message. They concluded that their scheme required less time.

Fraigniaud [3] proposed optimal $d-1$-fault-tolerant broadcasting schemes for d-dimensional hypercubes. His algorithm is based on Johnsson and Ho's [7] arc-

disjoint spanning tree. His scheme is optimal for short messages for both the shouting and whispering model. Moreover, his scheme is also sub-optimal for long messages for both models.

In this paper, we propose a new broadcasting scheme for faulty d-dimensional hypercubes that can tolerate up to $d-1$ node or link faults. Our scheme is applicable to both the whispering and shouting model. The new scheme is optimal in terms of the number of time units required to achieve k-fault-tolerant broadcast for faulty hypercubes. For non-faulty hypercubes, it requires even less time. It is also simpler, faster, and requires $n-d-1$ fewer message transmissions than Fraigniaud's scheme. The rest of the paper is organized as follows. In Section 2, we give the algorithm for $(d-1)$-fault-tolerant broadcasting in d-dimensional hypercubes. In Section 3, we compare the new scheme with previously proposed scheme.

2 An optimal $(d-1)$-fault-tolerant broadcasting scheme

A d-dimensional hypercube is a network with $n=2^d$ nodes. Each node can be coded by a binary sequence of length d. Two nodes are connected if the binary sequences differ in exactly one position. Each node $v=x_1x_2...x_d$ is connected to d nodes. We call the link that connects $v=x_1x_2...x_d$ to the node $u=y_1y_2...y_d$, the link of dimension i if $x_i \neq y_i$ and $x_j=y_j$ for $j \in \{1..d\}$ and $j \neq i$.

First, we assume that communication is carried out in the whispering model. The algorithm has two phases, the broadcast phase and the extended broadcast phase. The broadcast phase consists of d time units. In the first time unit of the broadcast phase, the originator o sends the broadcast message to an adjacent node through o_1, the link of dimension 1. In the 2^{nd} time unit, the originator and the node that got the message in time unit 1, sends the message through the

links of dimension 2. In the i^{th} time unit, the originator and all the nodes that already have the message send the message through the links of dimension i. The first phase requires d time units. After the first phase, every node will receive the message if the hypercube is non-faulty.

The second phase also consists of d time units. In the first time unit, every node sends a message through the links of dimension 1. In the second time unit, every node sends a message through the links of dimension 2. In the i^{th} time unit, every node sends a message through the links of dimension i. After the second phase, every node will have received d copies of the message if the hypercube is non-faulty.

It is obvious that every node in a non-faulty hypercube will receive the message after phase one of the new scheme. We will show that every node gets d node disjoint calling paths after both phases of the broadcast.

Consider a 4-dimensional hypercube with 16 nodes. Without loss of generality, let us assume that node 0(0000) is the originator. In time unit 1 of phase one, 0 sends to 8(1000). In time unit 2, 0 sends to 4, and 8 sends to 12. In time unit 3, 0 sends to 2, 4 sends to 6, 8 sends to 10, and 12 sends to 14. In the last time unit of phase one, 0 sends to 1, 2 sends to 3, 4 sends to 5, 6 sends to 7, 8 sends to 9, 10 sends to 11, 12 sends to 13, and 14 sends to 15.

For node 14, it gets a calling path from 0 to 8, 8 to 12, and 12 to 14 in the first phase. In the first time unit of phase two, node 14 gets another calling path from 0 to 4, 4 to 6, and 6 to 14. The calls from 0 to 4 and 4 to 6 are made in the first phase. The call from 6 to 14 is made in the first time unit of phase two. In the second time unit, another calling path is obtained from 0 to 2 in the first phase, 2 to 10 in the first call in the second phase, and from 10 to 14 in the second call of the second phase. The final calling path of node 14 is from 0 to 1 in the first

phase, 1 to 9 in the first call of the second phase, 9 to 13 in the second call of the second phase, 13 to 15 in the third call of the second phase, and 15 to 14 in the fourth call of the second phase. The calling paths for node 14 are:

1. 0(0000) -> 8(1000) -> 12(1100) -> 14(1110).
2. 0(0000) -> 4(0100) -> 6(0110) -> 14(1110)^1.
3. 0(0000) -> 2(0010) -> 10(1010)^1 -> 14(1110)^2.
4. 0(0000) -> 1(0001) -> 9(1001)^1 -> 13(1101)^2 -> 15(1111)^3 -> 14(1110)^4.

The calls with an ^{i} indicates they are calls made in the in the i^{th} time units of the second phase. The four calling paths are node disjoint.

Theorem 1: Every node in the hypercube will have d node disjoint calling paths from the originator after the two phases of the broadcast.

Proof: Without loss of generality, let u=00..0 be the originator. The d-dimensional hypercube is made up of two $(d-1)$-dimensional hypercubes. The first $(d-1)$-dimensional hypercube consists of all the nodes with a zero in their leftmost bit. The second $(d-1)$-dimensional hypercube consists all the nodes with a one in their leftmost bit. In time unit 1 of phase one, the originator that resides in one of the $(d-1)$-dimensional hypercubes calls a node in the other $(d-1)$-dimensional hypercube. Subsequent calls of phase one are made within the two $d-1$-dimensional hypercubes. Thus, each node in the second hypercube has one calling path originated from 00..0 to 10..0 and the calling path also consists of calls within its own subcube.

We call this group of nodes M_1. Similarly, the $(d-1)$-dimensional hypercube that the originator is in, is also made up of two $(d-2)$-dimensional hypercubes. In time unit 2 of phase one, the originator calls a node in the other $(d-2)$-dimensional hypercube. Each node in the other $(d-2)$-dimensional hypercube has one calling path originated from 00..0 to 010..0 and the calling path also consists of calls within its own subcube. We call this group of nodes M_2. Hence, after d time units, the nodes are divided into M_i groups where $i=1..d$ and the originator o. The group M_i is a $(d-i)$-dimensional hypercube.

Consider a node v in M_i. In phase two, the 1st calling path of v is from the originator to the node 1000..00 in M_1, followed by broadcasting within M_1, and a call between v and a node in M_1 through the link in the 1st dimension, v_1 in time unit 1 of phase two. The 2nd calling path of v is from the originator to the node 0100..00 in M_2, followed by broadcasting within M_2, and a call between v and a node in M_2 through the link in the 2nd dimension, v_2 in time unit 2. Similarly, The $i-1^{th}$ calling path of v is from the originator to the node with a 1 in the $i-1^{th}$ position 00..010..00 in M_{i-1}, followed by broadcasting within M_{i-1}, and a call between v and a node in M_{i-1} through the link in the $i-1^{th}$ dimension, v_{i-1} in time unit $i-1$ of phase 2. None of the links between v_1 and v_{i-1} are used in phase one of the broadcast. All the $i-1$ calling paths for v are disjoint.

If $i=d$, the node v will have $d-1$ edge and node disjoint calling path

after $d-1$ time units. The d^{th} calling path of v is obtained from a call from the originator to v in phase one. Hence, v gets d disjoint calling paths after both phases if $i=d$.

If $i \neq d$, the i^{th} calling path of v is from the originator to a node in M_j, followed by broadcasting within M_j, where $i<j \leq d$, and a call between v and a node in M_j in time unit i. All the nodes in M_i get one more edge disjoint calling path in time unit i except node m where m is the node that received the call from the originator in time unit i of phase one. Node m does not get another node disjoint calling path for it is called by the originator again in time unit i.

Similar to the original d-dimensional hypercube, M_i can be divided into $d-i$ groups, $M_{i,i+1}$ to $M_{i,d}$ and the node m. The group $M_{i,j}$ is a $(d-j)$-dimensional hypercube. The i^{th} calling path of nodes in $M_{i,j}$ is from the originator to a node in M_j, followed by broadcasting within M_j, and calls between M_j and $M_{i,j}$.

For the node m, it will get an additional $d-i$ edge disjoint calling paths from $M_{i,j}$ to m in time unit j where $i+1 \leq j \leq d$, through the link in dimension j, m_j. Although the link m_j has already been used in phase one of the broadcast, it is used to send messages from m to nodes in $M_{i,j}$. In phase two, we are using it in the other direction. Hence, node m will have d node disjoint paths from the originator after both phases.

For the node v, let v be in $M_{i,k}$. As described above, v will get its

$i+1^{th}$ to $i+k-1$ node disjoint calling paths from calls between v and a node in $M_{i,j}$ where $i+1 \leq j < k$ in time units $i+1^{th}$ to $i+k-1$.

In time unit $i+k$, again $M_{i,k}$ can be divided into $d-i-k$ groups, $M_{i,k,k+1}$ to $M_{i,k,d}$ and a single node m'. This process is repeated until v becomes a node in $M_{i,k,..,d}$ or v becomes the single node after a division. In both cases, the node v will have $d-1$ node disjoint calling paths after d time units of phase two. Adding the calling path of v obtained in the first phase, the node v has d node disjoint calling paths after both phases.

In phase one, $n-1$ messages are sent. In each time unit in phase two, n messages are sent. The total number of messages is $n(d+1)-1$ which is more than necessary to give every node d node disjoint calling paths. However, if we are a bit more careful in the extended broadcast, only $(n-1)d$ messages are sufficient. This can be accomplished by not sending messages from a node v to a node u if node v has already sent a message to node u in phase one. $n-1$ messages are used in phase one. The total number can then be reduced by $n-1$. Moreover, nodes do not have to send to the originator in the second phase. A further d messages are saved. The number of messages after two phases becomes $(n-1)d$. If some nodes or links have already failed in the hypercube, the number of messages sent is even less.

If the hypercube is non-faulty, the messages sent through links of dimension d in phase two are not necessary. These messages are sent from $n/2$ nodes that first received the message in time unit d of phase one through links of dimension d. Hence, these nodes knew that the nodes that are connected to them through links of dimension d already got the message. It is not necessary for these

nodes to send the message out in time unit d of phase two. Similarly, an additional $n/2-d-1$ messages can be reduced by not sending in phase two through the links where nodes got their messages in phase one. The total number of messages required for a non-faulty hypercube becomes $nd-n+1$.

The reduction described above also applies if the hypercube is faulty. If we assume that a node knew whether its links are faulty, or if we only consider node faults, the links used in phase one will only be used at most once. If a link is used in phase one to send the broadcast message from a node v to a node u, the node u will know that v has already received the message. Node u does not have to send to v in phase two. If u does not receive the message from v in phase one, u must send a message to v in phase two. In both cases, links used in phase one will only be used once resulting in a reduction of $n-d-1$ messages. Hence, the total number of messages required for the new scheme is less than $nd-n+1$ for faulty hypercubes.

Fraigniaud [3] proved that for the whispering model, at least $d+k+1$ time units are required to achieve k-fault-tolerant broadcast in a d-dimensional hypercube for $1 \le k \le d-1$. The new scheme requires $2d$ time units to achieve $d-1$-fault-tolerant broadcast in a d-dimensional hypercube. Thus, it is optimal in terms of the number of time units required for the whispering model. Furthermore, for a non-faulty d-dimensional hypercube, only $2d-1$ time units are sufficient for the last time unit in phase two can be omitted entirely.

Let the time to send out a message be $T=\beta+F\tau$ [7]. β is the start up time. τ is the time to send out one bit. F is the length of the broadcast message. The time requirement for the new scheme is optimal only if $\beta >> \tau$ or the message is short. For long messages, the same technique proposed by Johnsson and Ho [7], and Fraigniaud [3] can be applied. The message can be split into parts. Let the size of each part be p. In the extended broadcast of the first partition of the message,

the links used in the broadcast phase are all idle. We can use these idle links for the broadcast phase for the second partition. Similarly, we can use the idle links of the extended broadcast phase of the i^{th} partition for the broadcast phase of the $i+1^{th}$ partition. For the whispering model, the total time required is $d(F/p+1)(\beta+p\tau)$. This can be minimized by choosing $p=\sqrt{F\beta/\tau}$. The total time becomes $d(\sqrt{F\tau}+\sqrt{\beta})^2$ for faulty hypercubes. For non-faulty hypercubes, the total time is even smaller.

For the shouting model, after a node has received the message for the broadcast phase, it can immediately start sending out messages for the broadcast phase and the extended broadcast phase. The broadcast phase requires d time units and one more unit is required for nodes that received the message in the d^{th} time unit to send their messages for the extended broadcast phase. Hence, $d+1$ time units are required which is optimal for short messages.

$d+1$ time units are sufficient because a node v which receives the message in time unit i in the broadcast phase must have received messages from the extended broadcast phase through its links of dimension 1 to $i-1$ before time unit $i+1$. Since v receives the message in time unit i, the Hamming distance between v and the originator is equal to i. Bit $i+1$ to bit d of the originator and v are the same. Consider a node u that sends a message to v in the extended broadcast phase through one of the links of dimension 1 to $i-1$. The Hamming distance between bit 1 to bit i of u and v must be equal to 1. Bit $i+1$ to bit d of u and v are the same. The Hamming distance between u and the originator is at most $i-1$. u must have started its extended broadcast before time unit $i+1$. Hence, v must have received all the messages in the extended broadcasting from links of dimension 1 to $i-1$ before it sends out its messages for the extended broadcast through links of dimension 1 to $i-1$. Furthermore, using the same argument as in the whispering model, d time units are sufficient for non-faulty hypercubes.

For long messages, the same partitioning technique used in the whispering model can be applied. The total time required is $(\sqrt{F\tau}+\sqrt{d\beta})^2$ for faulty hypercubes. For non-faulty hypercubes, the total time is even smaller.

The new $d-1$-fault-tolerant broadcasting scheme can be modified easily to become a k-fault-tolerant broadcasting scheme where $k<d-1$. Instead of requiring d time units in the extended broadcast phase, $k+1$ time units are sufficient. The algorithm proceeds the same way as the $d-1$-fault-tolerant broadcasting scheme from time unit 1 to $k+1$. It is obvious that each node can receive k node disjoint calling paths in the extended phase.

3 Comparison

The number of time units required for the new scheme and Ramanathan and Shin's scheme are the same for the shouting model. However, Ramanathan and Shin's scheme does not work with the whispering model as their scheme requires communication between more than one adjacent node at the same time. Furthermore, the new scheme requires $n-1$ fewer message transmissions.

The new scheme and Fraigniaud's scheme have the same time and requirement for faulty hypercubes. For non-faulty hypercubes, the new scheme requires one less time unit or $\beta+p\tau$ less time. The new scheme also requires at least $n-d-1$ fewer message transmissions compared to Fraigniaud's scheme. This is because we do not send messages through links that already been used in phase one. Moreover, the new scheme is much simpler. Only one additional integer is required in the message to indicate the time unit the message is being sent. The receiver, on receiving the message, only has to check the integer to determine which adjacent node the message should be send to, and in which time units.

Consider a d-dimensional hypercube. The originator o sends the broadcast message and the integer $k=2d-2$ in the first time unit of phase one. The node that receives the message checks whether k is greater than d. If the answer is yes, subtract k from $2d$. Otherwise, subtract k from d. The result obtained is the dimension that the message has to be sent in the next time unit. After the message is sent, subtract 1 from k. The above process is repeated until $k<0$. Hence, only one comparison and one subtraction are necessary for each node in each time unit. The total number of operations for all the n nodes are $4nd$.

Fraigniaud's scheme also requires an integer j to be sent with the message to indicate which tree the message is being sent along. So, the amount of extra information being sent in the new scheme and Fraigniaud's scheme is roughly the same. However, in Fraigniaud's scheme, many more operations are required to determine what links should be used to send out the message. First, a node $v=x_{d-1}...x_1x_0$ has to find its children in the j^{th} broadcast tree. This is done by finding k. k is the bit position such that x_k is the first bit to the right of the bit at position j (cyclically). $k=-1$ if the $v=00...00$. $nd^2/2$ comparisons and nd assignments are required to find k for all the nodes in d edge disjoint broadcast trees.

After the value of k is known, the child u of v can be found by the following rules:

1. $u=x_{d-1}x_{d-2}...\neg x_j...x_0$, if $k=-1$.
2. $u=x_{d-1}x_{d-2}...\neg x_i...x_0$, $\forall i\in\{k+1,...,j-1,j\}$, if $x_j=1$ and $k\neq j$.
3. $u=x_{d-1}x_{d-2}...\neg x_i...x_0$, $\forall i\in\{k+1,...,j-1\}$, if $x_j=1$ and $k=j$.
4. $\varnothing$, if $x_j=0$ and $k\neq -1$.

Since, there are d edge disjoint broadcast trees and each tree has $n-1$ links, $3d(n-1)$ comparisons are required for all the nodes.

For the shouting model, Fraigniaud's scheme requires $4nd+nd^2/2-3d$ operations. The new scheme only requires $4nd$. For the whispering model, the new scheme requires the same number of operations. For Fraigniaud's scheme another comparison and addition operations are required to label the children to determine which time unit the link should be using. This adds an additional $2d(n-1)$ operations. Hence, the new scheme is a lot simpler. It also requires a lot less operations in each node to achieve the same optimal k-fault-tolerant broadcast in both models where $0 \le k < d$.

4 summary

A k-fault-tolerant broadcasting scheme for d-dimensional hypercubes where $0 \le k < d$ is proposed. For the whispering model, the new scheme requires $2d$ time units that are optimal for faulty-hypercubes and it only requires $2d-1$ time units for non-faulty hypercubes. For the shouting model, the new scheme requires $d+1$ time units that are optimal for faulty-hypercubes and d time units for non-faulty hypercubes. The new algorithm is simpler, faster, and requires $n-d-1$ fewer message transmissions than previously proposed optimal k-fault-tolerant broadcast schemes.

5 References

[1] Bruck, J.
Optimal Broadcasting in Faulty Hypercubes via Edge-Disjoint Embedding.
Networks :, To Appear.

[2] Carlsson, S., Y. Igarashi, K. Kanai, A. Lingas , K. Miura, and O. Peterson.
Information Disseminating Schemes for Fault Tolerance in Hypercubes.
IEICE Trans. Fund. E75:255-260, 1992.

[3] Fraigniaud, P.
Asymptotically Optimal Broadcasting and Gossiping in Faulty Hypercube Multicomputers.
IEEE Transaction on Computers 41(12):1410-1419, Dec, 1992.

[4] Gargano, L.
Tighter Time Bounds on Fault Tolerant Broadcasting and Gossiping.
Networks 22:469-486, 1992.

[5] Gargano, L., A.A. Rescigno, U. Vaccaro.
Fault Tolerant Hypercube Broadcasting via Information Dispersal.
Networks :, To Appear.

[6] Hedetniemi, S.M., S.T. Hedetniemi, and A.L. Liestman.
A Survey of Gossiping and Broadcasting in Communication Networks.
Networks 18:319-349, 1988.

[7] Johnsson, S., C.-T. Ho.
Optimal Broadcasting and Personalized Communication in Hypercubes.
IEEE Transaction on Computers 38(9):1249-1268, Sept, 1989.

[8] Rabin, M.O.
Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance.
JACM 36(2):335-348, 1989.

[9] Ramanathan, P., and K. G. Shin.
Reliable Broadcast in Hypercube Multicomputers.
IEEE Transactions on Computers 37(12):1654-1657, Dec, 1988.