プログラミング　11−10
(1997. 1. 23)

ビザンチン故障を考慮したトーラスにおける耐故障ブロードキャスト

梶原 由香[†], 岩崎 至宏[†], 小保方 幸次[†], 船生 豊[‡], 五十嵐 善英[†]

[†]群馬大学工学部情報工学科
[‡]横河電機 (株)
[†]〒 376 桐生市天神町 1-5-1
[†]Phone: +81-277-30-1829
[†]Email: igarashi@comp.cs.gunma-u.ac.jp

あらまし この論文は、トーラスにおけるビザンチン故障を考慮したブロードキャストプロトコルを設計し、その効率と耐故障性について論じる。n 次元トーラスの $2n$ 本の独立な全域木に沿ってソースノードから $2n$ 個のメッセージのコピーを送ることで、$2n-1$ 個までのクラッシュタイプの故障、$\lfloor (2n-1)/2 \rfloor$ 個までのビザンチンタイプの故障に耐えられる。ブロードキャストに要するステップ数は $2|V|-5n$ である。ここで、$|V|$ はトーラスのノード数であり、n はトーラスの次元数である。

キーワード　ビザンチンタイプ故障、クラッシュタイプ故障、耐故障ブロードキャスト、独立全域木、トーラス

Fault-Tolerance of Broadcasting in Tori with Byzantine Faults

Yuka Kajiwara[†], Yukihiro Iwasaki[†], Koji Obokata[†], Yutaka Funyu[‡], and Yoshihide Igarashi[†]

[†]Department of Computer Science,
Gunma University, Kiryu, 376 Japan
[‡]Advanced Technology Department,
Yokogawa Electric Corporation,
Nishi-Shinjuku, Shinjuku, Tokyo, 163-06 Japan
[†]Phone: +81-277-30-1829
[†]Email: igarashi@comp.cs.gunma-u.ac.jp

Abstract　We design a protocol for broadcasting on tori with Byzantine faults. Its efficiency and fault tolerance are analyzed. The protocol is to send $2n$ copies of the message from the source node through $2n$ independent spanning trees of an n-dimensional torus. It can tolerate up to $2n-1$ faults of the crash type and up to $\lfloor (2n-1)/2 \rfloor$ faults of the Byzantine type. The broadcasting time is $2|V|-5n$ steps, where $|V|$ and n are the order and the dimension of the torus, respectively.

key words　*Byzantine faults, crash faults, fault-tolerant broadcasting, independent spanning trees, tori*

1 Introduction

In this paper we consider two types of faults, the crash type and the Byzantine type. A fault of the crash type stops sending any message, and a fault of the Byzantine type can change arbitrarily the message. We design a protocol for broadcasting on tori with faults, and analyze its efficiency and fault tolerance. We assume that the source node is always faultless and that each node in the network does not know anything about faults in advance. In order to tolerate faults we send copies of the message from the source node through multiple channels. Each node decides the original message by the majority voting rule. This method was discussed on hypercubes and meshes in [1, 2, 4].

2 Definitions

Two paths connecting a pair of nodes are said to be internally disjoint if the two paths have no common nodes and no common edges excepting their extreme nodes. Two spanning trees of a graph $G = (V, E)$ are said to be independent if they are rooted at the same node, say s, and for each node v in V, the two paths from s to v, one path in each tree, are internally disjoint. A set of spanning trees of G are said to be independent if they are pairwise independent. A graph G is called an n-channel graph at node s if there are n independent spanning trees rooted at s of G. If G is an n-channel spanning trees rooted at every node, G is called an n-channel graph. For any $k \leq 3$, it is known that any k-connected graph is a k-channel graph [3, 5, 7]. However, it is open whether for $k \geq 4$, any k-connected graph is a k-channel graph.

The n-dimensional torus of $(r_0 \times \cdots \times r_{n-1})$ is a graph $Q_n = (V, E)$, where $V = \{(x_0, \ldots, x_{n-1}) \mid \text{for each } i\ (0 \leq i \leq n-1),\ 0 \leq x_i \leq r_i - 1\}$ and $E = \{(x_0, \ldots, x_{n-1})(x'_0, \ldots, x'_{n-1}) \mid \text{for some } i,\ x'_i = [x_i + 1]_{r_i}, \text{ and for any } j\ (j \neq i),\ x_j = x'_j\}$. Note that $[t]_n$ is t modulo n. It is known that any n-dimensional torus of $(r_0 \times \cdots \times r_{n-1})$ is a $2n$ channel graph if for any i $(0 \leq i \leq n-1)$, $r_i \geq 3$. If we send the message through $2n$ independent spanning trees rooted at the source node, the broadcasting can tolerate up to $2n-1$ faults of the crash type and up to $\lfloor (2n-1)/2 \rfloor$ faults of the Byzantine type.

A torus is a regular graph, and it is symmetric with respect to any node, without loss of generality we may discuss our broadcasting scheme with the fixed source node, $(0, \ldots, 0)$.

3 Independent Spanning Trees of a Torus

We describe how to construct $2n$ independent spanning trees rooted at $(0, \ldots, 0)$ of an n-dimensional torus Q_n of $(r_0 \times \cdots \times r_{n-1})$. Our construction consists of 2 ways. We can construct n spanning trees in each way. We therefore have $2n$ spanning trees altogether. Let the n spanning trees constructed in the first be denoted by $T = \{T_0, \ldots, T_{n-1}\}$, and let the n spanning trees constructed in the second way be denoted by $\bar{T} = \{\bar{T}_0, \ldots, \bar{T}_{n-1}\}$.

A general construction way of independent spanning trees of a product graph was given in [6]. However, our method can construct a set of independent spanning trees that are more efficient communication channels than the independent spanning trees constructed by the method in [6]. Our construction can be described by showing how the parent of each node is chosen.

For node $x = (x_0, \ldots, x_{n-1})$, x_i denotes the ith component of x. $neigh(x, i, +)$ and $neigh(x, i, -)$ denote neighbor nodes of x such that $x'_i = [x_i + 1]_{r_i}$ and $x''_i = [x_i - 1]_{r_i}$ respectively. $A \rightarrow B$ denotes the path of length 1 from node A to node B. The transitive and reflexive closure of $\rightarrow$ is denoted by $\overset{*}{\rightarrow}$.

Construction 1: the construction of T_i $(0 \leq i < n)$

1. If $x_i = 0$ then $neigh(x, i, +)$ is the parent of x.

2. If $x_i = r_i - 1$ then $neigh(x, i, -)$ is the parent of x.

3. When $0 < x_i < r_i - 1$, let j be the smallest positive integer such that $x_{[i-j]_n} \neq 0$ and let $k = [i - j]_n$. If $x_k = r_k - 1$ then $neigh(x, k, +)$ is the parent of x, and if $x_k < r_k - 1$ then $neigh(x, k, -)$ is the parent of x.

We show an example of our construction in Fig.1, where the dark node denotes the source node. Since the parent of x specified by Construction 1 is a neighbor node of x, T_i is a subgraph of Q_n. By the next lemma T_i is a spanning tree of Q_n.

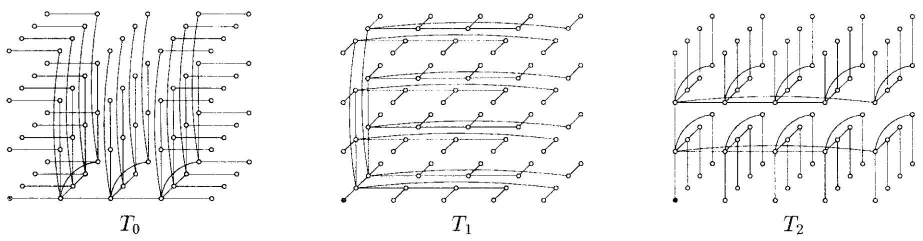

Fig. 1: Spanning trees of Q_n by Construction 1.

Lemma 1 *For any i $(0 \leq i < n)$, the subgraph T_i of Q_n, constructed by Construction 1, is a spanning tree of Q_n.*

Proof: Let N be the number of nodes of Q_n. For any node that is not the source node, just one edge is added to connect the node to its parent, the number of edges of T_i is exactly $N-1$. We next show that for any node v, there is a path from the source node to v in T_i.

Let $s = (0,\dots,0)$ be the source node, and let $x = (x_0,\dots,x_{n-1})$ be an arbitrary node of Q_n. If $x_i = 0$, there is a path $s \to neigh(s,i,+) \xrightarrow{*} (0,\dots,0,1,x_{i+1},0,\dots,0) \xrightarrow{*} (x_0,\dots,x_{i-1},1,x_{i+1},\dots,x_{n-1}) \to x$. If $x_i = r_i - 1$, there is a path $s \to neigh(s,i,+) \xrightarrow{*} (0,\dots,0,x_i-2,0,\dots,0) \xrightarrow{*} (0,\dots,0,x_i-2,x_{i+1},0,\dots,0) \xrightarrow{*} (x_0,\dots,x_{i-1},x_i-2,x_{i+1},\dots,x_{n-1}) \to x$. If $0 < x_i < r_i - 1$, there is a path $s \to neigh(s,i,+) \xrightarrow{*} (0,\dots,0,x_i,0,\dots,0) \xrightarrow{*} (x_0,\dots,x_{i-2},0,x_i,\dots,x_{n-1}) \xrightarrow{*} x$. Hence, there is a path in T_i from s to any node. T_i is therefore a spanning tree of Q_n. $\square$

Lemma 2 *The set of spanning trees $T = \{T_0,\dots,T_{n-1}\}$ constructed by Construction 1 is independent.*

Proof: Let T_i and T_j $(i < j)$ be a pair of distinct spanning trees arbitrarily chosen from T. We show that for any node x, two paths P_i and P_j from root s to x, one through T_i and the other through T_j are internally disjoint. Let $u = (u_0,\dots,u_{n-1})$ and $v = (v_0,\dots,v_{n-1})$ be internal nodes of P_i and P_j, respectively. If $x_i = 0$, $u_i = 1$ and $v_i = 0$. Hence, in this case P_i and P_j are internally disjoint. Symmetrically we can show that P_i and P_j are internally disjoint in the case $x_j = 0$. If $x_i = r_i - 1$, then $0 < u_i < r_i - 1$, and $v_i = 0$ or $r_i - 1$. Hence, in this case P_i and P_j are also internally disjoint. Symmetrically we can show that P_i and P_j are internally disjoint in the case $x_j = r_j - 1$.

Suppose that $0 < x_i < r_i - 1$ and $0 < x_j < r_j - 1$. Let k be the largest integer such that $i \leq k < j$ and $x_k \neq 0$. Let $u' = (u'_0,\dots,u'_{n-1})$ be the first node on P_i such that $u'_k = x_k$. Then the kth component of every node on P_i before u' is 0, and the kth component of every node on P_i after u' is x_k. The jth component of every internal node on P_j is $v_j > 0$. When the kth component of a node on P_j has changed to x_k, the obtained node is the last one on P_j. Hence, in this case P_i and P_j are also internally disjoint. $\square$

Construction 2: the construction of $\bar{T}_i$ $(0 \leq i < n)$

1. If $x_i = 0$ then $neigh(x,i,-)$ is the parent of x.
2. If $0 < x_i < r_i - 1$ then $neigh(x,i,+)$ is the parent of x.
3. When $x_i = r_i - 1$, let j be the smallest positive integer such that $x_{[i-j]_n} \neq 0$, and let $k = [i-j]_n$. If $x_k = r_k - 1$ then $neigh(x,k,+)$ is the parent of x, and if $x_k < r_k - 1$ then $neigh(x,k,-)$ is the parent of x.

We show an example of our construction in Fig.2, where the dark point denotes the source node. Since the parent of x specified by Construction 2 is a neighbor node of x in the torus, $\bar{T}_i$ is a subgraph of Q_n. By the next lemma $\bar{T}_i$ is a spanning tree of Q_n.

Lemma 3 *For any i $(0 \leq i < n)$, the subgraph $\bar{T}_i$ of Q_n, constructed by Construction 2 is a spanning tree of Q_n.*

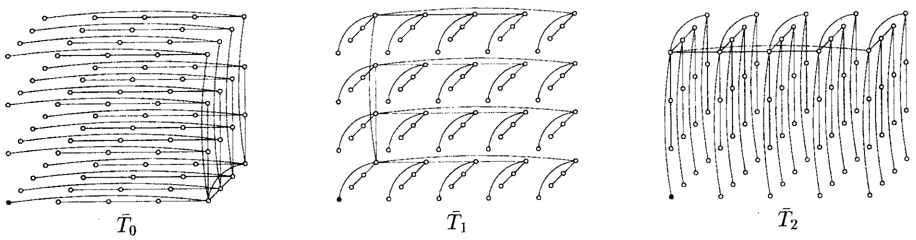

Fig. 2: Spanning trees of Q_n by Construction 2.

Proof: Let N be the number of nodes of Q_n. For any node that is not the source node, just one edge is added to connect the node to its parent, the number of edges of $\bar{T}_i$ is exactly $N-1$. We next show that for any node v, there is a path from the source node to v in $\bar{T}_i$.

Let $s = (0, \ldots, 0)$ be the source node, and let $x = (x_0, \ldots, x_{n-1})$ be an arbitrary node of Q_n. If $x_i = 0$, then there exists a path $s \to neigh(s, i, -) \overset{*}{\to} (0, \ldots, 0, r_i - 1, x_{i+1}, 0, \ldots, 0) \overset{*}{\to} (x_0, \ldots, x_{i-1}, r_i - 1, x_{i+1}, \ldots, x_{n-1}) \to x$. If $0 < x_i < r_i - 1$, then there exists a path $s \to neigh(s, i, -) \overset{*}{\to} (0, \ldots, 0, r_i - 1, r_{i+1}, 0, \ldots, 0) \overset{*}{\to} (x_0, \ldots, x_{i-1}, r_i - 1, x_{i+1}, \ldots, x_{n-1}) \overset{*}{\to} x$. If $x_i = r_i - 1$ then there exists a path $s \to neigh(s, i, -) \overset{*}{\to} (0, \ldots, 0, x_i, x_{i+1}, 0, \ldots, 0) \overset{*}{\to} (x_0, \ldots, x_{i-2}, 0, x_i, \ldots, x_{n-1}) \overset{*}{\to} x$. Hence, there exists a path in $\bar{T}_i$ from s to any node in Q_n. $\bar{T}_i$ is therefore a spanning tree of Q_n. □

Lemma 4 *The set of spanning trees $\bar{T} = \{\bar{T}_0, \ldots, \bar{T}_{n-1}\}$ constructed by Construction 2 is independent.*

Proof: Let $\bar{T}_i$ and $\bar{T}_j$ $(i < j)$ be a pair of distinct spanning trees arbitrarily chosen from $\bar{T}$. We show that for any node x, two paths P_i and P_j from root s to x, one through T_i and the other through T_j are internally disjoint. Let $u = (u_0, \ldots, u_{n-1})$ and $v = (v_0, \ldots, v_{n-1})$ be internal nodes of P_i and P_j, respectively. If $x_i = 0$, then $u_i = r_i - 1$ and $v_i = 0$. Hence, in this case P_i and P_j are internally disjoint. Symmetrically we can show that P_i and P_j are internally disjoint in the case $x_j = 0$. If $0 < x_i < r_i - 1$, then $u_i > x_i$ and $v_i \leq x_i$. Hence, in this case P_i and P_j are internally disjoint. Symmetrically P_i and P_j are internally disjoint in the case where $0 < x_j < r_j - 1$.

Assume that $x_i = r_i - 1$ and $x_j = r_j - 1$. Let k be the largest integer such that $i \leq k < j$ and $x_k \neq 0$. Let $u' = (u'_0, \ldots, u'_{n-1})$ be the first node on P_i such that $u'_k = x_k$. Then the kth component of every node on P_i before u' is 0, and the kth component of every node on P_i after u' is x_k. The jth component of every internal node on P_j is $v_j > 0$. When the kth component of a node on P_j has changed to x_k, the obtained node is the last node on P_j. Hence, in this case P_i and P_j are internally disjoint. □

Lemma 5 *For any T_i of T and $\bar{T}_j$ of $\bar{T}$ $(0 \leq i, j < n)$, these two trees are independent spanning trees of Q_n.*

Proof: We show that for any node x of Q_n two paths P_i and P_j, one through T_i and the other through $\bar{T}_j$ are internally disjoint. Let $u = (u_0, \ldots, u_{n-1})$ and $v = (v_0, \ldots, v_{n-1})$ be internal nodes of P_i and P_j, respectively.

Assume $x_i = 0$. If $i = j$, then $u_i = 1$ and $v_i = r_i - 1$. If $i \neq j$ then $u_i = 1$ and $v_i = 0$. Hence, in this case P_i and P_j are internally disjoint. If $x_j = 0$ and $i \neq j$, then $u_j = 0$ and $v_j = r_j - 1$. Hence, in this case P_i and P_j are internally disjoint.

We next assume that $0 < x_i < r_i - 1$ and $0 < x_j < r_j - 1$. If $i = j$ then $u_j \leq x_j$ and $v_j > x_j$. Hence, in this case P_i and P_j are internally disjoint. Similarly we can show that P_i and P_j are also internally disjoint in the case $i \neq j$.

Finally we assume that $0 < x_i < r_i - 1$ and $x_j = r_j - 1$. We first assume $i < j$. Let k be the largest integer such that $i \leq k < j$ and $x_k \neq 0$. Let $u' = (u'_0, \ldots, u'_{n-1})$ be the first node on P_i such that $u'_k = x_k$. If u locates before u' on P_i, $u_k < x_k$ and $u_j = 0$. If u locates not before u' on P_i, $u_k = x_k$. The jth component of every internal node on P_j is $v_j > 0$. When the kth component of a node on P_j has changed to x_k, the obtained node is the last node of P_j. Hence, in this case P_i and P_j are also internally disjoint. □

From Lemma 2, Lemma 4, and Lemma 5, the following theorem is immediate.

Theorem 1 *The set of $2n$ spanning trees of Q_n, $T \cup \bar{T}$ constructed by Construction 1 and Construction 2 is independent.*

4 A Broadcasting Protocol

We describe a protocol for broadcasting along independent spanning trees in $T \cup \bar{T}$ from the $source = (0, \ldots, 0)$. We assume that each node has no information about faults in advance and that the source node is always faultless.

The broadcasting consists of two stages, First Stage and Second Stage. We denote First Stage by F and Second Stage by S. Each stage consists of n rounds. In each round each node can send a message and/or can receive a message in a direction (dimension).

Let V_{dim} denote a message value sent along spanning tree, T_{dim}, and let $\bar{V}_{dim}$ denote a received message value that comes along spanning tree $\bar{T}_{dim}$. We denote the number of iterations in each stage by $w(stage, round)$.

$$w(stage, round) = \begin{cases} 1 & \textbf{if } stage = F \textbf{ and } r_{round} = 3 \\ r_i - 3 & \textbf{elseif } stage = F \textbf{ and } r_{round} > 3 \\ 2 & \textbf{elseif } stage = S \textbf{ and } r_{round} = 3 \\ r_i - 2 & \textbf{elseif } stage = S \textbf{ and } r_{round} \text{ is odd} \\ r_i - 3 & \textbf{elseif } stage = S \textbf{ and } r_{round} \text{ is even} \end{cases}$$

A message value that is sent from node x or has been received in a round of a stage is denoted by $val(stage, x, round)$.

$$val(stage, x, round) = \begin{cases} \bar{V}_j & \textbf{if } stage = F \textbf{ and } j < round \textbf{ and } x_j = r_j - 1, \\ & \text{where } j \text{ is the smallest integer such that } 0 \le j \le round \text{ and } x_j \ne 0. \\ V_j & \textbf{elseif } stage = F \textbf{ and } j \le round \\ \bar{V}_k & \textbf{elseif } stage = S \textbf{ and } x_k = r_k - 1, \\ & \text{where } k \text{ is the smallest integer such that } round < k < n \text{ and } x_k \ne 0. \\ V_k & \textbf{elseif } stage = S \textbf{ and } x_k < r_k - 1 \\ \text{undefine} & \textbf{otherwise} \end{cases}$$

We are now ready to describe our protocol using the symbols defined above.
/* First Stage */

```
begin
for $round := 0$ to $n-1$ do
  for $time := 0$ to $w(F, round)$ do
    if $^{\forall}index\ (round < index < n),\ x_{index} = 0$ then begin
      /* send */
      if $x = source$ then
        if $time = 0$ then
          $send(V_{round}, neigh(x, round, +))$
        elseif $time = 1$ then
          $send(\bar{V}_{round}, neigh(x, round, -))$
      elseif $x_{round} = time$ then
        $send(val(F, x, round), neigh(x, round, +))$
      elseif $time = 1$ and $x_{round} = 0$ then
        $send(val(F, x, round), neigh(x, round, -))$;
      /* receive */
      if $x = neigh(source, round, +)$ or $x = neigh(source, round, -)$ then
        if $x = neigh(source, round, +)$ and $time = 0$ then
          $receive(V_{round}, source)$                                        $(F1)$
        elseif $x = neigh(source, round, -)$ and $time = 1$ then
          $receive(\bar{V}_{round}, source)$                                        $(F2)$
      elseif $x_{round} = time + 1$ then
        $receive(val(F, neigh(x, round, -), round), neigh(x, round, -))$        $(F3)$
      elseif $time = 1$ and $x_{round} = r_{round} - 1$ then
        $receive(val(F, neigh(x, round, +), round), neigh(x, round, +))$        (F4)
    end
end.

/* Second Stage */

begin
for $round := 0$ to $n - 1$ do
  for $time := 0$ to $w(S, round)$ do begin
      /* send */
    if $round < n - 1$ and $^{\exists}index(round < index < n)$ such that $x_{index} \neq 0$ and
          $x_{round} \leq w(F, round)$ and $x_{round} = time$ then
      $send(val(S, x, round), neigh(x, round, +))$
    elseif $round < n - 1$ and $^{\exists}index(round < index < n)$ such that $x_{index} \neq 0$ and
          $time = 1$ and $x_{round} = 0$ then
      $send(val(S, x, round), neigh(x, round, -))$
    else
      if $neigh(x, round, -) \neq source$ and $time = 0$ and $x_{round} = 1$ then
        $send(V_{round}, neigh(x, round, -))$
      elseif $time = 1$ and $x_{round} = r_{round} - 1$ then
        $send(\bar{V}_{round}, neigh(x, round, +))$
      else
        case $r_{round} = 3$
          if $time = 2$ then
            if $x_{round} = r_{round} - 1$ then
              $send(\bar{V}_{round}, neigh(x, round, -))$
            elseif $x_{round} = r_{round} - 2$ then
              $send(V_{round}, neigh(x, round, +))$
        case $r_{round}$ is even
          if $time = 0$ and $x_{round} = r_{round} - 2$ then
            $send(V_{round}, neigh(x, round, +))$
          elseif $x_{round} = r_{round} - 1 - time$ then
            $send(\bar{V}_{round}, neigh(x, round, -))$
        case $r_{round}$ is odd
          if $time = 0$ and $x_{round} = r_{round} - 2$ then
            $send(V_{round}, neigh(x, round, +))$
          elseif $time = 0$ and $x_{round} = r_{round} - 1$ then
            $send(\bar{V}_{round}, neigh(x, round, -))$
          elseif $time \geq 2$ and $x_{round} = r_{round} - time$
            $send(\bar{V}_{round}, neigh(x, round, -))$;
        /* receive */
    if $round < n - 1$ and $^{\exists}index(round < index < n)$ such that $x_{index} \neq 0$ and
          $x_{round} \leq w(F, round)$ and $x_{round} = time + 1$ then
      $receive(val(S, neigh(x, round, -), round), neigh(x, round, -))$        (S1)
    elseif $round < n - 1$ and $^{\exists}index(round < index < n)$ such that $x_{index} \neq 0$ and
          $time = 1$ and $x_{round} = r_{round} - 1$ then
      $receive(val(S, neigh(x, round, +), round), neigh(x, round, +))$        (S2)
    else
      if $x \neq source$ and $time = 0$ and $x_{round} = 0$ then
        $receive(V_{round}, neigh(x, round, +))$        (S3)
      elseif $time = 1$ and $x_{round} = 0$ then
```

```
        $receive(\bar{V}_{round}, neigh(x, round, -))$                    $(S4)$
      else
        case $r_{round} = 3$
          if $time = 2$ then
            if $x_{round} = r_{round} - 2$ then
              $receive(\bar{V}_{round}, neigh(x, round, +))$              $(S5)$
            elseif $x_{round} = r_{round} - 1$ then
              $receive(V_{round}, neigh(x, round, -))$                    $(S6)$
        case $r_{round}$ is even
          if $time = 0$ and $x_{round} = r_{round} - 1$ then
            $receive(V_{round}, neigh(x, round, -))$                      $(S7)$
          elseif $x_{round} = r_{round} - 2 - time$ then
            $receive(\bar{V}_{round}, neigh(x, round, +))$                $(S8)$
        case $r_{round}$ is odd
          if $time = 0$ and $x_{round} = r_{round} - 1$ then
            $receive(V_{round}, neigh(x, round, -))$                      $(S9)$
          elseif $time = 0$ and $x_{round} = r_{round} - 2$ then
            $receive(\bar{V}_{round}, neigh(x, round, +))$                $(S10)$
          elseif $time \geq 2$ and $x_{round} = r_{round} - 1 - time$
            $receive(\bar{V}_{round}, neigh(x, round, +))$                $(S11)$
  end
end.
```

Using First Stage and Second Stage given above we can describe our broadcasting protocol is describe as follows:

```
broadcast
begin
  First Stage;
  Second Stage;
  $V =$ the majority of $\{V_0, \ldots, V_{n-1}, \bar{V}_0, \ldots, \bar{V}_{n-1}\}$
end.
```

Hereafter, $p(x, T_i)$ denotes the parent of x in T_i, and $R(round, time, A)$ denotes a statement specified by A in First Stage and Second Stage of the protocol, where A is one of $F1, \ldots, F4$ (First Stage) or one of $S1, \ldots, S11$ (Second Stage) at the timing specified by round and time. For the broadcasting protocol the following lemma holds true.

Lemma 6 *For any node x, by the broadcasting protocol defined above x sends at most one message in each unit time interval (step) along a direction, and the message sent by x can be received by the neighbor node of x in the direction in the same unit time interval.*

Proof: From the definition of the protocol it is immediate that any node x sends at most one message and receives at most one message in each step. We show that a sending message from a node x can be received by the corresponding neighbor node of x in the same step. In order to this fact it is sufficient to show that for any node x and during any round, $neigh(x, round, +)$ and $neigh(x, round, -)$ never sends messages to x in the same step.

For First Stage, during any round if $0 < x_{round} < r_{round} - 1$ then only $neigh(x, round, -)$ can send a message to x, and if $x_{round} = r_{round} - 1$ then only $neigh(x, round, +)$ can send a message to x. Hence, for First Stage two nodes never send message to the same node in the same step.

For Second Stage, during a round message may be sent to a node along two directions. However, these two messages are sent in different steps. For x satisfying $x_{round} = 0$, $neigh(x, round, +)$ can send a message to x only when $time = 0$, and $neigh(x, round, -)$ can send a message to x only when $time = 1$. For x satisfying $x_{round} = r_i - 1$, $neigh(x, round, +)$ can send a message to x only when $time = 1$, and $neigh(x, round, -)$ can send a message to x only when $time = 0$ (for $r_{round} = 3$, only when $time = 2$). For x satisfying $0 < x_{round} < r_{round} - 1$,

$neigh(x, round, +)$ can send a message to x only when $time = r_{round} - x_{round} - 2$ if r_{round} is even and only when $time = r_{round} - x_{round} - 1$ if r_{round} is odd (for $r_{round} = 3$, only when $time = 2$), and $neigh(x, round, -)$ can send a message to x only when $time = x_{round} - 1$. Hence, for Second Stage two nodes never send to the same node in the same step. This lemma therefore holds true. □

We can prove the following four lemmas. Due to the page limit we omit the proofs of these lemmas.

Lemma 7 *For any node x of Q_n, x receives a message sent along each T_i.*

Lemma 8 *For any node x of Q_n, x receives a message sent along each $\bar{T}_i$.*

Lemma 9 *The broadcasting protocol defined in this section can be implemented in the one-port model of Q_n.*

Lemma 10 *The broadcasting time by the protocol on Q_n of $(r_0 \times \cdots \times r_{n-1})$ is $2|V| - 5n$, where $|V|$ is the number of nodes of Q_n and $n \geq 3$.*

From these lemmas we can obtain the next theorem.

Theorem 2 *The broadcasting by the protocol can be implemented on the 1-port model of Q_n, and its broadcasting time is $2|V| - 5n$, where $|V|$ is the number of nodes of Q_n. It can tolerate up to $2n - 1$ faults of the crash type and up to $\lfloor (2n-1)/2 \rfloor$ faults of the Byzantine type.*

5 Concluding Remarks

We show a fault-tolerant broadcasting protocol on tori. There are many ways of constructing $2n$ independent spanning trees of a n-dimensional torus. It is interesting to investigate how we can choose $2n$ independent spanning trees with nice properties for broadcasting. It is also worthy to study efficient and reliable broadcasting protocols for other network families.

References

[1] F. Bao and Y. Igarashi, "Reliable broadcasting in product networks with Byzantine Faults", 26th Annual International Symposium on Fault-Tolerant Computing, Sendai, pp.262–271, 1996.

[2] F. Bao, K. Katano, Y. Funyu and Y. Igarashi, "Fault tolerance of broadcasting in hypercubes, meshes and tori", Technical Report of IEICE, Fault-tolerant Systems, FTS95–79, pp.31–38, 1996.

[3] A. Itai and M. Rodeh, "The multi-tree approach to reliability in distributed networks", Information and Computation, vol.79, pp.43–59, 1988.

[4] Y. Kajiwara and Y. Igarashi, "Fault-Tolerant Broadcasting in Mesh-Connected Networks", Technical Report of IEICE, Computation, COMP96–26, pp.37–46, 1996.

[5] S. Khuller and B. Schieber, "On independent spanning trees", Information Processing Letters, vol.42, pp.321–323, 1992.

[6] K. Obokata, Y. Iwasaki, F. Bao and Y. Igarashi, "Independent spanning trees of product graphs", 22nd International Workshop on Graph-Theoretic Concepts in Computer Science, Como, Italy, 1996.

[7] A. Zehavi and A. Itai, "Three tree-paths", J. Graph Theory, vol.13, pp.175–188, 1989.