IEICE TRANSACTIONS

on Information and Systems

DOI:10.1587/transinf.2024EDP7076

Publicized:2024/09/05

This advance publication article will be replaced by the finalized version after proofreading.

A PUBLICATION OF THE INFORMATION AND SYSTEMS SOCIETY

The Institute of Electronics, Information and Communication Engineers

Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER

Fault-tolerant Routing in Bicubes*

Yitong WANG†, **Htoo Htoo Sandi KYAW**†, ***Nonmembers***, **Kunihiro FUJIYOSHI**†, ***and*** **Keiichi KANEKO**†a), ***Members***

SUMMARY The bicube is derived from the hypercube, and it provides a topology for interconnection networks of parallel systems. The bicube can interconnect the same number of nodes with the same degree as the hypercube while its diameter is almost half of that of the hypercube. In addition, the bicube preserves the property of node symmetry. Hence, the bicube attracts much attention. In this paper, we focus on the bicube with faulty nodes and propose three fault-tolerant routing methods to find a fault-free path between any pair of non-faulty nodes in it.

key words: *hypercube, interconnection network, massively parallel system, topology*

1. Introduction

The topology of the interconnection network of a parallel system defines the pattern to interconnect the processing elements by using links among them in the system, and the system performance is closely related to the topology. The topological structure of an interconnection network can be discussed in a framework of the graph theory by regarding its processing elements and links as nodes and edges, respectively.

Many topologies have been proposed for interconnection networks. The bicube proposed by Lim et al. [2] is one such topology. It is a variant of the hypercube [3]. The bicube can interconnect the same number of nodes with the same degree as the hypercube while its diameter is almost half of the hypercube. Additionally, it has the good property of node symmetry, where each node has the same view of the network. Thus, a single common algorithm can be executed on each node. Therefore, it can be easily applied to massively parallel systems, and it is recently studied eagerly [4]–[8]. Because a massively parallel system includes many processing elements, it is unrealistic to operate it while ignoring faulty processing elements. Hence, it is important to design algorithms so that they can tolerate faulty nodes. Thus, in this study, we focus on the bicube with faulty nodes and propose three methods to find a fault-free path between any pair of non-faulty nodes in it. Specifically, in a bicube with a set of permanently faulty nodes, for a non-faulty source node $\boldsymbol{s}$ and a non-faulty destination node $\boldsymbol{d}$, we propose an adaptive routing method that finds a non-faulty path from $\boldsymbol{s}$ to $\boldsymbol{d}$. In this situation, it is assumed that each non-faulty node can detect its neighboring faulty nodes in a constant time.

In a previous work [8], Okada and Kaneko proposed a shortest-path routing algorithm in the bicube. For a node $\boldsymbol{s}$ with a message in a bicube to a certain destination node $\boldsymbol{t}$ with the distance $d(\boldsymbol{s},\boldsymbol{t})$, their method divides the neighbor nodes of $\boldsymbol{s}$ into two disjoint subsets: the preferred neighbor node set $Pre(\boldsymbol{s},\boldsymbol{t})$ and the spare neighbor node set $Spr(\boldsymbol{s},\boldsymbol{t})$. The nodes in $Pre(\boldsymbol{s},\boldsymbol{t})$ are on the shortest paths from $\boldsymbol{s}$ to $\boldsymbol{t}$. On the other hand, $Spr(\boldsymbol{s},\boldsymbol{t})$ includes the neighbor nodes that are not included in $Pre(\boldsymbol{s},\boldsymbol{t})$.

In this paper, we first show that for any node $\boldsymbol{w}(\in Spr(\boldsymbol{s},\boldsymbol{t}))$, $d(\boldsymbol{w},\boldsymbol{t}) = d(\boldsymbol{s},\boldsymbol{t}) + 1$ holds. In other words, we show that $Spr(\boldsymbol{s},\boldsymbol{t})$ does not include any node $\boldsymbol{w}$ such that $d(\boldsymbol{w},\boldsymbol{t}) = d(\boldsymbol{s},\boldsymbol{t})$. Next, we propose three fault-tolerant routing methods in the bicube, and compare them with a baseline method by conducting a computer experiment.

The rest of this paper is structured as follows. In Section 2, we introduce the necessary definitions and Theorems. We describe our methods in details in Section 3. In Section 4, we explain the details of the computer experiment, and its results. In Section 5, we give a conclusion and a future work.

2. Preliminaries

In this section, we give relevant definitions and a theorem.

Definition 1: An n-dimensional hypercube, Q_n, is an undirected graph whose node set is $\{0,1\}^n$. Given two nodes $\boldsymbol{a}$ and $\boldsymbol{b}$ in Q_n, $\boldsymbol{a}$ and $\boldsymbol{b}$ are adjacent if and only if $H(\boldsymbol{a},\boldsymbol{b}) = 1$, where $H(\boldsymbol{a},\boldsymbol{b})$ represents the Hamming distance between $\boldsymbol{a}$ and $\boldsymbol{b}$. □

Figure 1 shows an example of a 4-dimensional hypercube, Q_4.

Next, we give a definition of the lp-relation regarding two n-dimensional bit sequences.

Definition 2: Given a bit sequence $\boldsymbol{a} = (a_{n-1}, a_{n-2}, \ldots, a_0)(\in \{0,1\}^n)$ where n is even, define a function $p(\boldsymbol{a})$ by $p(\boldsymbol{a}) = a_{n-1} \oplus a_{n-2} \oplus \cdots \oplus a_0$, where the operator $\oplus$ represents the exclusive-or operation. Then, given a pair of bit sequences $\boldsymbol{a}, \boldsymbol{b}(\in \{0,1\}^n)$, $\boldsymbol{a}$ and $\boldsymbol{b}$ are in lp-relation if

Final manuscript received January 1, 202X.

†The authors are with Graduate School of Engineering, Tokyo University of Agriculture and Technology, Koganei-shi, 184-8588 Japan.

*This paper is an extended version based on a conference paper [1], and it provides full descriptions of the lemma proofs and the computer experiment.

a) E-mail: k1kaneko@cc.tuat.ac.jp (Corresponding author).

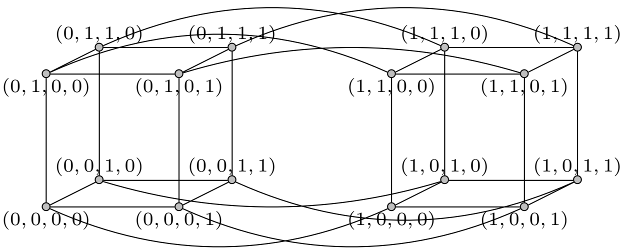

Fig. 1 Example of a 4-dimensional hypercube, Q_4.

and only if either '$\boldsymbol{a} = \boldsymbol{b}$ and $p(\boldsymbol{a}) = p(\boldsymbol{b}) = 0$' or '$\boldsymbol{a} = \bar{\boldsymbol{b}}$ and $p(\boldsymbol{a}) = p(\boldsymbol{b}) = 1$' holds. □

For example, for two bit sequences $(1, 0, 0, 0, 1, 0)$ and $(1, 0, 0, 0, 1, 0)$, they are in lp-relation because $(1, 0, 0, 0, 1, 0) = (1, 0, 0, 0, 1, 0)$ and $p(1, 0, 0, 0, 1, 0) = p(1, 0, 0, 0, 1, 0) = 0$. For another pair of bit sequences $(1, 1, 0, 0, 1, 0)$ and $(0, 0, 1, 1, 0, 1)$, they are in lp-relation because $(1, 1, 0, 0, 1, 0) = \overline{(0, 0, 1, 1, 0, 1)}$ and $p(1, 1, 0, 0, 1, 0) = p(0, 0, 1, 1, 0, 1) = 1$.

Now, we give a definition and a notation of the set of neighbor nodes, and a definition of the bicube based on them.

Definition 3: For a node $\boldsymbol{a}$ in a graph, let $N(\boldsymbol{a}) = \{\boldsymbol{n} \mid d(\boldsymbol{a}, \boldsymbol{n}) = 1\}$ represent the set of neighbor nodes of $\boldsymbol{a}$, where $d(\boldsymbol{a}, \boldsymbol{b})$ represents the distance between $\boldsymbol{a}$ and $\boldsymbol{b}$. □

Definition 4: An n-dimensional bicube, B_n, is an undirected graph, whose node set is $\{0, 1\}^n$. For a node $\boldsymbol{a} = (a_{n-1}, a_{n-2}, \ldots, a_0)$ in B_n, there are n neighbor nodes $N(\boldsymbol{a}) = \{\boldsymbol{a}^{(0)}, \boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(n-1)}\}$. The $(n-1)$ nodes $\boldsymbol{a}^{(i)}$ $(0 \leq i \leq n-2)$ are given by $\boldsymbol{a}^{(i)} = (a_{n-1}, a_{n-2}, \ldots, a_{i+1}, \overline{a_i}, a_{i-1}, \ldots, a_0)$ while the node $\boldsymbol{a}^{(n-1)}$ is given depending on the parity of n. That is, if n is odd, $\boldsymbol{a}^{(n-1)} = (\overline{a_{n-1}}, b_{n-2}, \ldots, b_0)$, where $(b_{n-2}, b_{n-3}, \ldots, b_0)$ is the bit sequence that is in lp-relation with $(a_{n-2}, a_{n-3}, \ldots, a_0)$. If n is even, $a^{(n-1)} = (\overline{a_{n-1}}, a_{n-2}, b_{n-3}, \ldots, b_0)$, where $(b_{n-3}, b_{n-4}, \ldots, b_0)$ is the bit sequence that is in lp-relation with $(a_{n-3}, a_{n-4}, \ldots, a_0)$. □

Figure 2 shows an example of a 4-dimensional bicube, B_4. For example, for the node $\boldsymbol{a} = (a_3, a_2, a_1, a_0) = (0, 1, 1, 0)$ in the figure, $\boldsymbol{a}^{(2)} = (0, 0, 1, 0)$, $\boldsymbol{a}^{(1)} = (0, 1, 0, 0)$, and $\boldsymbol{a}^{(0)} = (0, 1, 1, 1)$. In addition, $\boldsymbol{a}^{(n-1)} = (0, 1, 1, 0)^{(3)} = (\bar{0}, 1, 0, 1) = (1, 1, 0, 1) = (\overline{a_3}, a_2, b_1, b_0)$ because n is even and $(b_1, b_0) = (0, 1)$ is in lp-relation with $(a_1, a_0) = (1, 0)$.

The hypercube used to be the most popular topology and it has many variants, including the bicube. Almost all of them have the degree n and interconnect 2^n nodes when they are n-dimensional. We compare the properties of the bicube with other cube-based topologies in terms of the diameter and the symmetry, by summarizing them in Table 1.

Table 1 shows the comparison of an n-dimensional bicube B_n with an n-dimensional hypercube Q_n [3], an n-dimensional 0-Möbius cube 0-M_n [9], an n-dimensional 1-Möbius cube 1-M_n [9], an n-dimensional crossed cube C_n [10], an n-dimensional twisted cube T_n[11], an n-dimensional twisted crossed cube TC_n [12], an n-dimensional locally twisted cube LT_n [13], and an n-dimensional spined cube S_n [14].

The diameter is an important property of interconnection networks because it indicates the maximum number of routing hops to transfer a message. As shown in Table 1, the bicube and other variants of the hypercube have smaller diameters than that of the hypercube. Another important property of an interconnection network is the symmetry of the network. As shown in Table 1, only the bicube and the hypercube have the node symmetry. A topology with the node symmetry is more conducive to designing algorithms, because each node can use a single algorithm to process a certain task including the fault-tolerant routing. Therefore, the bicube provides a promising topology for the interconnection networks of the massively parallel systems.

Regarding the distance between two arbitrary nodes in B_n with odd n, Theorems 1 and 2 are provided by previous works.

Theorem 1: Given a source node $\boldsymbol{s} = (s_{n-1}, s_{n-2}, \ldots, s_0)$ and a destination node $\boldsymbol{t} = (t_{n-1}, t_{n-2}, \ldots, t_0)$ in B_n ($n(\geq 3)$: odd), let $\boldsymbol{u} = (u_{n-1}, u_{n-2}, \ldots, u_0) = \boldsymbol{s} \oplus \boldsymbol{t}$ and $h = \sum_{i=0}^{n-1} u_i (= H(\boldsymbol{s}, \boldsymbol{t}))$. If $\sum_{i=0}^{n-2} s_i$ is even, the distance between $\boldsymbol{s}$ and $\boldsymbol{t}$, $d(\boldsymbol{s}, \boldsymbol{t})$, is given by:

$$d(\boldsymbol{s}, \boldsymbol{t}) = \begin{cases} 3 & (h = n), \\ \min\{h, 4\} & (h = n-1, u_{n-1} = 0), \\ 2 & (h = n-1, u_{n-1} = 1), \\ \min\{h, n-h+1\} & (h \leq n-2). \end{cases}$$

(Proof) From [2]. □

Theorem 2: Given a source node $\boldsymbol{s} = (s_{n-1}, s_{n-2}, \ldots,$

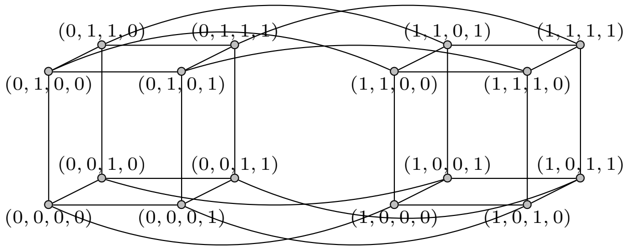

Fig. 2 Example of a 4-dimensional bicube, B_4.

Table 1 Comparison of cube-based topologies with order 2^n and degree n.

Topology	Diameter	Symmetry Node	Symmetry Edge
Hypercube	n	✓	✓
Bicube	$\lceil (n+1)/2 \rceil^{\star}$	✓	✗
0-Möbius Cube	$\lceil (n+2)/2 \rceil$	✗	✗
1-Möbius Cube	$\lceil (n+1)/2 \rceil$	✗	✗
Crossed Cube	$\lceil (n+1)/2 \rceil$	✗	✗
Twisted Cube	$\lceil (n+1)/2 \rceil$	✗	✗
Twisted Crossed Cube	$\lceil (n+1)/2 \rceil$	✗	✗
Locally Twisted Cube	$\lceil (n+3)/2 \rceil^{\dagger}$	✗	✗
Spined Cube	$\lceil n/3 \rceil + 3^{\ddagger}$	✗	✗

$\star$: $n \geq 7$, $\dagger$: $n \geq 5$, $\ddagger$: $n \geq 14$

s_0) and a destination node $\boldsymbol{t} = (t_{n-1}, t_{n-2}, \ldots, t_0)$ in B_n ($n(\geq 3)$: odd), let $\boldsymbol{u} = (u_{n-1}, u_{n-2}, \ldots, u_0) = \boldsymbol{s} \oplus \boldsymbol{t}$ and $h = \sum_{i=0}^{n-1} u_i (= H(\boldsymbol{s}, \boldsymbol{t}))$. If $\sum_{i=0}^{n-2} s_i$ is odd, the distance between $\boldsymbol{s}$ and $\boldsymbol{t}$, $d(\boldsymbol{s}, \boldsymbol{t})$, is given by:

$$d(\boldsymbol{s},\boldsymbol{t}) = \begin{cases} 1 & (h = n), \\ \min\{h, 4\} & (h = n-1, u_{n-1} = 0), \\ 2 & (h = n-1, u_{n-1} = 1), \\ 3 & (h = 1, u_{n-1} = 1), \\ \min\{h, n-h+1\} & (\text{otherwise}). \end{cases}$$

(Proof) From [8]. □

Finally, we give a definition of the preferred, spare, sideward, and backward neighbor node sets with a related theorem.

Definition 5: Given a source node $\boldsymbol{s}$ and a destination node $\boldsymbol{t}$, let $Pre(\boldsymbol{s},\boldsymbol{t})$, $Spr(\boldsymbol{s},\boldsymbol{t})$, $Swd(\boldsymbol{s},\boldsymbol{t})$, and $Bwd(\boldsymbol{s},\boldsymbol{t})$ represent subsets of the neighbor nodes of $\boldsymbol{s}$, $N(\boldsymbol{s})$, where

$Pre(\boldsymbol{s},\boldsymbol{t}) = \{\boldsymbol{c} \mid d(\boldsymbol{c},\boldsymbol{t}) = d(\boldsymbol{s},\boldsymbol{t}) - 1, \boldsymbol{c} \in N(\boldsymbol{s})\}$,
$Spr(\boldsymbol{s},\boldsymbol{t}) = N(\boldsymbol{s}) \setminus Pre(\boldsymbol{s},\boldsymbol{t})$,
$Swd(\boldsymbol{s},\boldsymbol{t}) = \{\boldsymbol{c} \mid d(\boldsymbol{c},\boldsymbol{t}) = d(\boldsymbol{s},\boldsymbol{t}), \boldsymbol{c} \in Spr(\boldsymbol{s},\boldsymbol{t})\}$, and
$Bwd(\boldsymbol{s},\boldsymbol{t}) = \{\boldsymbol{c} \mid d(\boldsymbol{c},\boldsymbol{t}) = d(\boldsymbol{s},\boldsymbol{t}) + 1, \boldsymbol{c} \in Spr(\boldsymbol{s},\boldsymbol{t})\}$.

$Pre(\boldsymbol{s},\boldsymbol{t})$, $Spr(\boldsymbol{s},\boldsymbol{t})$, $Swd(\boldsymbol{s},\boldsymbol{t})$, and $Bwd(\boldsymbol{s},\boldsymbol{t})$ are called the sets of preferred, spare, sideward, and backward neighbor nodes of $\boldsymbol{s}$ to $\boldsymbol{t}$, respectively. □

Figure 3 shows the relationship between the set of preferred neighbor nodes and the set of spare neighbor nodes.

Theorem 3 shows the preferred neighbor node set $Pre(\boldsymbol{s},\boldsymbol{t})$ and the spare neighbor node set $Spr(\boldsymbol{s},\boldsymbol{t})(= N(\boldsymbol{s}) \setminus Pre(\boldsymbol{s},\boldsymbol{t}))$ for a source node $\boldsymbol{s}$ and a destination node $\boldsymbol{d}$ in B_n with odd n.

Theorem 3: Given a source node $\boldsymbol{s} = (s_{n-1}, s_{n-2}, \ldots, s_0)$ and a destination node $\boldsymbol{t} = (t_{n-1}, t_{n-2}, \ldots, t_0)$ in B_n ($n(\geq 5)$: odd), let $\boldsymbol{u} = (u_{n-1}, u_{n-2}, \ldots, u_0) = \boldsymbol{s} \oplus \boldsymbol{t}$ and $h = \sum_{i=0}^{n-2} u_i$. Then, the preferred neighbor node set $Pre(\boldsymbol{s},\boldsymbol{t})$ and the spare neighbor node set $Spr(\boldsymbol{s},\boldsymbol{t})$ are given by Table 2.
(Proof) From [8]. □

Next, we prove that $Bwd(\boldsymbol{s},\boldsymbol{t}) = Spr(\boldsymbol{s},\boldsymbol{t})$, that is, $Swd(\boldsymbol{s},\boldsymbol{t}) = \emptyset$ in Fig. 3 for any pair of the source node $\boldsymbol{s}$ and the destination node $\boldsymbol{t}$ in the bicube.

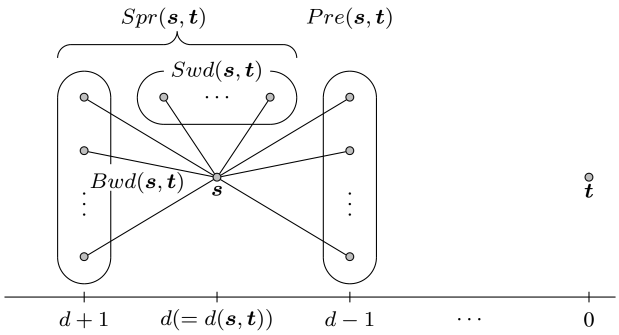

Fig. 3 Relationship among preferred, spare, sideward, and backward neighbor node sets of $\boldsymbol{s}$ to $\boldsymbol{t}$.

Lemma 1: B_n is bipartite.
(Proof) For a node $\boldsymbol{a}$ in B_n, let $\boldsymbol{a}^{(i)}$ ($0 \leq i \leq n-1$) be its neighbor node. If $0 \leq i \leq n-2$, $H(\boldsymbol{a}, \boldsymbol{a}^{(i)}) = 1$. Hence, $p(\boldsymbol{a}^{(i)}) = 1 - p(\boldsymbol{a})$. If $i = n-1$ and n is odd, $\boldsymbol{a}^{(i)} = (\overline{a_{n-1}}, b_{n-2}, \ldots, b_0)$, where $(b_{n-2}, b_{n-3}, \ldots, b_0)$ is the bit sequence that is in lp-relation with $(a_{n-2}, a_{n-3}, \ldots, a_0)$. That is, $p(b_{n-2}, b_{n-3}, \ldots, b_0) = p(a_{n-2}, a_{n-3}, \ldots, a_0)$. Hence, $p(\boldsymbol{a}^{(i)}) = 1 - p(\boldsymbol{a})$. If $i = n-1$ and n is even, $\boldsymbol{a}^{(i)} = (\overline{a_{n-1}}, a_{n-2}, b_{n-3}, \ldots, b_0)$, where $(b_{n-3}, b_{n-4}, \ldots, b_0)$ is the bit sequence that is in lp-relation with $(a_{n-3}, a_{n-4}, \ldots, a_0)$. That is, $p(b_{n-3}, b_{n-4}, \ldots, b_0) = p(a_{n-3}, a_{n-4}, \ldots, a_0)$. Hence, $p(\boldsymbol{a}^{(i)}) = 1 - p(\boldsymbol{a})$. Therefore, B_n is bipartite. □

Theorem 4: For a source node $\boldsymbol{s}$ and a destination node $\boldsymbol{t}$ in B_n, $Swd(\boldsymbol{s},\boldsymbol{t}) = \emptyset$.
(Proof) From Lemma 1, there is no cycle of odd length in B_n. Now, assume that there is a node $\boldsymbol{u} \in Swd(\boldsymbol{s},\boldsymbol{t})$. Then, there is a path P from $\boldsymbol{s}$ to $\boldsymbol{t}$, whose length is $d = d(\boldsymbol{s},\boldsymbol{t})$. Also, there is a path Q from $\boldsymbol{u}$ to $\boldsymbol{t}$, whose length is d. Let $\boldsymbol{v}$ be the common node of P and Q that is closest to $\boldsymbol{s}$ and $\boldsymbol{u}$. Note that $\boldsymbol{v}$ may be $\boldsymbol{t}$. Now, there is a cycle C that consists of the subpaths $\boldsymbol{s} \rightsquigarrow \boldsymbol{v}$, $\boldsymbol{v} \rightsquigarrow \boldsymbol{u}$, and the edge $\boldsymbol{u} \rightarrow \boldsymbol{s}$ (Fig. 4). Then, the length of C is $d(\boldsymbol{s},\boldsymbol{v}) + d(\boldsymbol{v},\boldsymbol{u}) + 1 = 2d(\boldsymbol{s},\boldsymbol{v}) + 1$ because $d(\boldsymbol{s},\boldsymbol{v}) = d(\boldsymbol{v},\boldsymbol{u}) = d - d(\boldsymbol{v},\boldsymbol{t})$, and the existence of C of odd length contradicts Lemma 1. Hence, $Swd(\boldsymbol{s},\boldsymbol{t}) = \emptyset$. □

3. Proposed Methods

In the rest of this paper, we assume that permanent faults may occur in multiple nodes. A faulty node loses all communication functions with its neighbor nodes. A non-faulty node can detect its neighbor faulty nodes in $O(1)$ time using the time-out mechanism. The occurrence of faulty nodes during operation is not considered.

In general, due to the existence of faulty nodes, message routings may fail. A routing failure occurs in two situations. In the first situation, the node that has the message cannot find any neighbor node to forward the message. In the second situation, the message is infinitely forwarded along a fixed cycle or between two adjacent nodes.

Okada and Kaneko proposed a shortest-path routing method in B_n [8]. For a source node $\boldsymbol{s}$ with a message and a destination node $\boldsymbol{t}$ in B_n without faulty nodes, their method

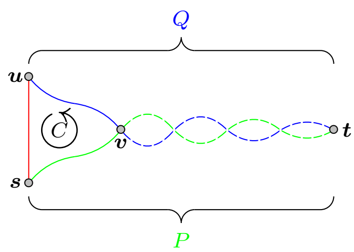

Fig. 4 Cycle C: $\boldsymbol{s} \rightsquigarrow \boldsymbol{v} \rightsquigarrow \boldsymbol{u} \rightarrow \boldsymbol{s}$ of odd length.

Table 2 $Pre(\boldsymbol{s},\boldsymbol{t})$ and $Spr(\boldsymbol{s},\boldsymbol{t})$ in B_n ($n(\geq 5)$: odd) [8].

Coverage			Results		
$\sum_{i=0}^{n-2} s_i$	h	Additional Conditions	$Pre(\boldsymbol{s},\boldsymbol{t})$	$Spr(\boldsymbol{s},\boldsymbol{t})$	$d(\boldsymbol{s},\boldsymbol{t})$
even	$h=n$	—	$\{\boldsymbol{s}^{(i)} \mid 0 \leq i \leq n-2\}$	$\{\boldsymbol{s}^{(n-1)}\}$	3
	$h=n-1$	$u_{n-1}=0$	$\{\boldsymbol{s}^{(i)} \mid 0 \leq i \leq n-1\}$	$\emptyset$	4
		$u_{n-1}=1$	$\{\boldsymbol{s}^{(q)} \mid u_q=0\}$	$\{\boldsymbol{s}^{(q)} \mid u_q=1\}$	2
	$\lceil n/2 \rceil < h \leq n-2$	$u_{n-1}=0, h=n-2$	$\{\boldsymbol{s}^{(n-1)}\}$	$\{\boldsymbol{s}^{(i)} \mid 0 \leq i \leq n-2\}$	3
		$u_{n-1}=1$ or $h \leq n-3$	$\{\boldsymbol{s}^{(q)} \mid u_q=0\}$	$\{\boldsymbol{s}^{(q)} \mid u_q=1\}$	$n-h+1$
	$h < \lceil n/2 \rceil$	$u_{n-1}=1, h \leq 2$	$\{\boldsymbol{s}^{(n-1)}\}$	$\{\boldsymbol{s}^{(i)} \mid 0 \leq i \leq n-2\}$	h
		$u_{n-1}=0$ or $3 \leq h$	$\{\boldsymbol{s}^{(q)} \mid u_q=1\}$	$\{\boldsymbol{s}^{(q)} \mid u_q=0\}$	h
	$h=\lceil n/2 \rceil$	—	$\{\boldsymbol{s}^{(i)} \mid 0 \leq i \leq n-1\}$	$\emptyset$	$\lceil n/2 \rceil$
odd	$h=n$	—	$\{\boldsymbol{s}^{(n-1)}\}$	$\{\boldsymbol{s}^{(i)} \mid 0 \leq i \leq n-2\}$	1
	$h=n-1$	$u_{n-1}=0$	$\{\boldsymbol{s}^{(i)} \mid 0 \leq i \leq n-1\}$	$\emptyset$	4
		$u_{n-1}=1$	$\{\boldsymbol{s}^{(n-1)}\}$	$\{\boldsymbol{s}^{(i)} \mid 0 \leq i \leq n-2\}$	2
	$\lceil n/2 \rceil < h \leq n-2$	$u_{n-1}=0, h=n-2$	$\{\boldsymbol{s}^{(n-1)}\}$	$\{\boldsymbol{s}^{(i)} \mid 0 \leq i \leq n-2\}$	3
		$u_{n-1}=0, h \leq n-3$	$\{\boldsymbol{s}^{(q)} \mid u_q=0\}$	$\{\boldsymbol{s}^{(q)} \mid u_q=1\}$	$n-h+1$
		$u_{n-1}=1$	$\{\boldsymbol{s}^{(n-1)}, \boldsymbol{s}^{(q)} \mid u_q=0\}$	$\{\boldsymbol{s}^{(q)} \mid u_q=1, q \neq n-1\}$	$n-h+1$
	$h < \lceil n/2 \rceil$	$u_{n-1}=0$	$\{\boldsymbol{s}^{(q)} \mid u_q=1\}$	$\{\boldsymbol{s}^{(q)} \mid u_q=0\}$	h
		$u_{n-1}=1, h=1$	$\{\boldsymbol{s}^{(i)} \mid 0 \leq i \leq n-2\}$	$\{\boldsymbol{s}^{(n-1)}\}$	3
		$u_{n-1}=1, 2 \leq h$	$\{\boldsymbol{s}^{(q)} \mid u_q=1, q \neq n-1\}$	$\{\boldsymbol{s}^{(n-1)}, \boldsymbol{s}^{(q)} \mid u_q=0\}$	h
	$h=\lceil n/2 \rceil$	—	$\{\boldsymbol{s}^{(i)} \mid 0 \leq i \leq n-1\}$	$\emptyset$	$\lceil n/2 \rceil$

obtains $Pre(\boldsymbol{s},\boldsymbol{t})$ in $O(n)$ time, forwards the message to a node in it, and repeats the process regarding the node as a new source node until the message arrives at the destination node.

First, we extend their method to be applied to B_n with a faulty node set F. We call this extended method 'Simple'. In Simple, if $Pre(\boldsymbol{s},\boldsymbol{t}) \not\subset F$, it selects a non-faulty node in $Pre(\boldsymbol{s},\boldsymbol{t})$. Otherwise, that is, if $Pre(\boldsymbol{s},\boldsymbol{t}) \subset F$, Simple selects a non-faulty node in $Bwd(\boldsymbol{s},\boldsymbol{t})$. The pseudo code of Simple is shown in Fig. 5.

The existence of faulty nodes causes detours in the message routing. In the situation, a non-faulty node from $Bwd(\boldsymbol{s},\boldsymbol{t}) \setminus F$ can be arbitrarily selected because $Swd(\boldsymbol{s}, \boldsymbol{t}) = \emptyset$ is guaranteed by Theorem 4.

```
procedure Simple(s, t)
/*
** s: node that has the message
** t: destination node
*/
while s <> t do begin
  Fwd := Pre(s,t) \ F; Bwd := (N(s) \ Pre(s,t)) \ F;
  if Fwd <> ∅ then select s from Fwd
  else if Bwd <> ∅ then select s from Bwd
  else error ('message delivery failed')
end
```

Fig. 5 The baseline algorithm Simple.

In the subsequent subsections, we propose three methods that tolerate the node fault to find a fault-free path from a non-faulty source node $\boldsymbol{s}$ to a non-faulty destination node $\boldsymbol{t}$ in B_n with a set of faulty nodes F.

3.1 Method1

When we applied Simple to faulty bicubes, we observed many routing failures. Hence, we analyzed how these failures have occurred in B_{11}, B_{12}, and B_{13}, and found that 40 to 50 percent of them were caused when the messages are at 3 hops to the destination nodes (Fig. 6).

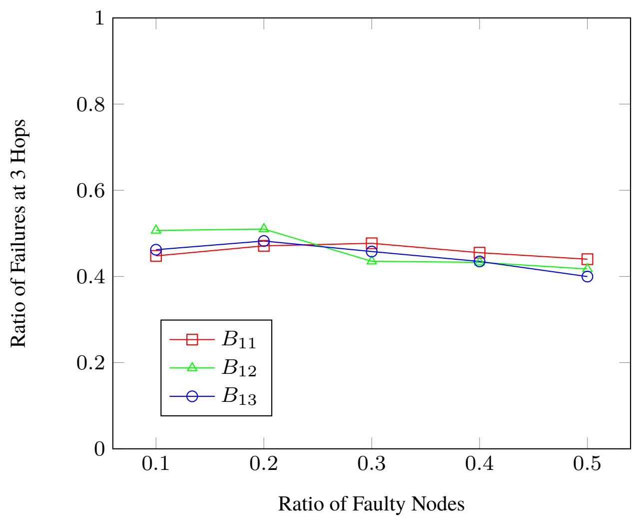

Fig. 6 Ratios of failure cases where messages are at distance 3 to destination nodes in B_{11}, B_{12}, and B_{13}.

To address this problem, we have devised a method, Method1, in which the depth-first search is adopted to ensure the message delivery when the message is at 3 hops to the destination node. In Fig. 7, the node $\boldsymbol{s}$, which has the message, first sends a probe signal including the information of the destination node $\boldsymbol{t}$ to the node $\boldsymbol{u}$ in $Pre(\boldsymbol{s}, \boldsymbol{t})$. However, $\boldsymbol{u}$ returns the reject signal because there is no non-faulty node in $Pre(\boldsymbol{u}, \boldsymbol{t})$. Because the node $\boldsymbol{v}$ is faulty, it is skipped. Then, $\boldsymbol{s}$ sends another probe signal to the node $\boldsymbol{w}$. Because there are non-faulty nodes in $Pre(\boldsymbol{w}, \boldsymbol{t})$, $\boldsymbol{w}$ returns the acknowledge signal. Hence, $\boldsymbol{s}$ forwards the message to $\boldsymbol{w}$. Then, $\boldsymbol{w}$ forwards the message to its non-faulty neighbor node in $Pre(\boldsymbol{w}, \boldsymbol{t})$, and it is delivered to the destination node $\boldsymbol{t}$. Note that signal exchange is only performed between the node $\boldsymbol{s}$ and the non-faulty preferred neighbor nodes in $Pre(\boldsymbol{s}, \boldsymbol{t})$. $|Pre(\boldsymbol{s}, \boldsymbol{t}) \setminus F| = O(n)$ and it takes $O(n)$ time to obtain $Pre(\boldsymbol{u}, \boldsymbol{t}) \setminus F$ at each node $\boldsymbol{u}(\in (Pre(\boldsymbol{s}, \boldsymbol{t}) \setminus F))$. Hence, a call of the depth-first search at a node, say $\boldsymbol{s}$ in Fig. 7, takes $O(n^2)$ time.

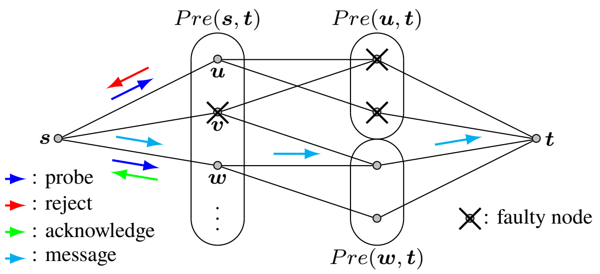

Fig. 7 Depth-first search of Method1 with $d(\boldsymbol{s}, \boldsymbol{t}) = 3$.

This process is repeated until the message is delivered to the destination node or the message delivery fails by an infinite loop. We assume that the infinite loop can be detected by the timeout mechanism.

Figure 8 shows the pseudo code of Method1 in B_n. For a source node $\boldsymbol{s}$ and a destination node $\boldsymbol{t}$, it is invoked by `Method1(s, t)`.

```
procedure Method1(s, t)
/*
** s: node that has the message
** t: destination node
*/
while s <> t do begin
  Fwd := Pre(s,t) \ F; Bwd := (N(s) \ Pre(s,t)) \ F;
  if d(s,t) = 3 then begin
    execute DFS to find w ∈ Fwd
      that ensures message delivery;
    if w exists then s := w
    else if Bwd <> ∅ then select s from Bwd
    else error ('message delivery failed') end
  else if Fwd <> ∅ then select s from Fwd
  else if Bwd <> ∅ then select s from Bwd
  else error ('message delivery failed')
end
```

Fig. 8 Fault-tolerant routing algorithm Method1.

In B_5 with a faulty node set $F = \{(0,1,1,0,1), (1,0,1,1,0), (0,1,0,1,0), (1,1,1,1,0), (1,1,1,0,0), (1,1,0,0,1)\}$, let the source node $\boldsymbol{s}$ and the destination node $\boldsymbol{t}$ be $(1,0,0,1,0)$ and $(1,1,1,0,1)$, respectively. Then, $Fwd(\boldsymbol{s}, \boldsymbol{t}) = \{(0,1,1,0,1), (1,1,0,1,0), (1,0,1,1,0), (1,0,0,0,0), (1,0,0,1,1)\}$, where $(1,1,0,1,0), (1,0,0,0,0), (1,0,0,1,1) \notin F$. If there are multiple non-faulty nodes in $Fwd(\boldsymbol{s}, \boldsymbol{t})$, the node with a different bit in the highest dimension is selected. Hence, Method1 selects its neighbor node $(1,1,0,1,0)$ and forwards the message to it. Now, the distance between the node with the message and the destination node is 3. Hence, to select the neighbor node to forward the message, the routing switches to the depth-first search. First, a probe signal is sent to a non-faulty neighbor node $(1,1,0,0,0)(\in Fwd((1,1,0,1,0), \boldsymbol{t}) \setminus F)$. Then, $Fwd((1,1,0,0,0), \boldsymbol{t}) = \{(1,1,1,0,0), (1,1,0,0,1)\} \subset F$. Thus, a reject signal is sent back to $(1,1,0,1,0)$. Hence, another non-faulity node $(1,1,0,1,1)(\in Fwd((1,1,0,1,0), \boldsymbol{t}) \setminus F)$ is selected, and the probe signal is sent to it. Here, $(1,1,1,1,1)(\in Fwd((1,1,0,1,1), \boldsymbol{t}))$ is not faulty, and an acknowledge signal is sent back. Finally, the message is delivered to the destination node $\boldsymbol{t}$ via $(1,1,0,1,1)$ and $(1,1,1,1,1)$ because $\boldsymbol{t} = (1,1,1,0,1) \in Fwd((1,1,1,1,1), \boldsymbol{t}) \setminus F$.

3.2 Method2

In the actual routing process, we observed that many routing failures are caused by infinite loops between two adjacent nodes. To solve this problem, we propose Method2, which inhibits forwarding the message to the previous node. When a node receives a message from the previous node, it can detect the previous node easily by checking the input channel buffer of its router that contains the message. Hence, additional information to the message is not required at all. In Method2, each node never forwards the message to its previous node. This process is repeated until the message is delivered to the destination node.

Figure 9 shows the pseudo code of Method2 in B_n. For a source node $\boldsymbol{s}$ and a destination node $\boldsymbol{t}$, it is invoked by `Method2(s, t)`. In the pseudo code, $\boldsymbol{p}$ represents the previous node, and it is initialized by $\boldsymbol{s}$ as a dummy node because there is not any previous node when the message is initially injected from the source node.

In B_5 with a set of faulty node $F = \{(0,0,0,1,1), (0,1,0,1,0), (1,1,1,1,1), (1,1,0,1,0), (1,0,0,1,0), (0,0,1,$

```
procedure Method2(s, t)
/*
** s: node that has the message
** t: destination node
*/
p := s; /* p: previous node */
while s <> t do begin
  Fwd := (Pre(s,t) \ {p}) \ F;
  Bwd := ((N(s) \ Pre(s,t)) \ {p}) \ F;
  p := s;
  if Fwd <> ∅ then select s from Fwd
  else if Bwd <> ∅ then select s from Bwd
  else error ('message delivery failed')
end
```

Fig. 9 Fault-tolerant routing algorithm Method2.

$1, 1)\}$, let the source node $\boldsymbol{s}$ and the destination node $\boldsymbol{t}$ be $(0, 1, 0, 1, 1)$ and $(1, 1, 0, 1, 1)$, respectively. Then, $Fwd(\boldsymbol{s}, \boldsymbol{t}) = \{(0, 0, 0, 1, 1), (0, 1, 1, 1, 1), (0, 1, 0, 0, 1), (0, 1, 0, 1, 0)\}$, where $(0, 1, 1, 1, 1), (0, 1, 0, 0, 1) \notin F$. If there are multiple non-faulty nodes in $Fwd(\boldsymbol{s}, \boldsymbol{t})$, the node with a different bit in the highest dimension is selected. Hence, Method2 selects its neighbor node $(0, 1, 1, 1, 1)$ and forwards the message to it. Then, $Fwd((0, 1, 1, 1, 1), \boldsymbol{t}) = \{(1, 1, 1, 1, 1)\} \subset F$, Because there is not any non-faulty node in $Fwd((0, 1, 1, 1, 1), \boldsymbol{t})$, a non-faulty node in $Bwd((0, 1, 1, 1, 1), \boldsymbol{t})$ must be selected. If there are multiple non-faulty nodes in $Bwd((0, 1, 1, 1, 1), \boldsymbol{t})$, the node with a different bit in the highest dimension is selected. Now, $Bwd((0, 1, 1, 1, 1), \boldsymbol{t}) \setminus F = \{(0, 1, 0, 1, 1), (0, 1, 1, 0, 1), (0, 1, 1, 1, 0)\}$. However, Method2 never forwards the message to the previous node $(0, 1, 0, 1, 1)$. Hence, Method2 selects the node $(0, 1, 1, 0, 1)$, and forwards the message to it. Next, the message is forwarded to the node $(0, 0, 1, 0, 1) (\in Fwd((0, 1, 1, 0, 1), \boldsymbol{t}) \setminus F)$, and then to the node $(0, 0, 1, 0, 0) (\in Fwd(0, 0, 1, 0, 1), \boldsymbol{t}) \setminus F)$. Because $(1, 1, 0, 1, 1) (= \boldsymbol{t}) \in (Fwd((0, 0, 1, 0, 0), \boldsymbol{t}) \setminus F)$, the message is forwarded to $\boldsymbol{t}$, and the delivery is finished successfully.

3.3 Method3

Because the improvements introduced in Method1 and Method2 are compatible, we have devised another method, Method3, which includes these improvements simultaneously. In Method3, the node that has the message never selects the previous node to forward the message as in Method2. In addition, if the destination node is 3 hops from the node, the routing is switched to the depth-first search as in Method1. The routing process is repeated until the message is delivered to the target node.

Figure 10 shows the pseudo code of Method3 in B_n. For a source node $\boldsymbol{s}$ and a destination node $\boldsymbol{t}$, it is invoked by `Method3(s, t)`. In the pseudo code, $\boldsymbol{p}$ represents the previous node, and it is initialized by $\boldsymbol{s}$ as a dummy node because there is not any previous node when the message is initially injected from the source node.

4. Computer Experiment

To evaluate the proposed methods, we conducted a computer experiment by the following steps. As a baseline algorithm, we adopted a method, Simple, which is described in Fig. 5.

(1) In B_n ($n = 11, 12, 13$), for each of the ratio of faulty nodes, $\alpha = 0.1, 0.2, \ldots, 0.5$, conduct the following Steps (2) to (5) 10,000 times.
(2) Select $\lfloor \alpha 2^n \rfloor$ faulty nodes randomly.
(3) Select the source node $\boldsymbol{s}$ and the destination node $\boldsymbol{t}$ randomly among the non-faulty nodes.
(4) If there is no fault-free path between $\boldsymbol{s}$ and $\boldsymbol{t}$, go back to Step (2) and start over the trial.
(5) Apply Method1, Method2, Method3, and Simple, and measure the number of successful routings and the path lengths in the successful routings.

Because the connectivity between the source node and the destination node is guaranteed in Step (4), the least upper bound of the ratio of successful routings is 1.

Figures 11, 12, and 13 show the ratios of the successful routings in B_{11}, B_{12}, and B_{13}, respectively. As shown in the figures, our proposed methods outperform Simple in any dimension and in any proportion of faulty nodes. In addition, Method2 and Method3 have a greater performance improvement than Method1. Moreover, Method3 is slightly better than Method2. Method1 showed better ratios of successful routings than Simple by at most 0.0612 in B_{11}, 0.0623 in B_{12}, and 0.0615 in B_{13}, respectively. Method2 showed better ratios than a baseline method Simple by at most 0.347 in B_{11}, 0.3342 in B_{12}, and 0.3755 in B_{13}, respectively. Method3 showed better ratios than Simple by at most 0.3637 in B_{11}, 0.3769 in B_{12}, and 0.3965 in B_{13}, respectively. Especially, for Method3, when the ratio of faulty nodes is less than 0.2, the minimum ratio of the successful routings can reach 0.9989 in B_{11}, 0.9990 in B_{12}, and 0.9998 in B_{13}.

Figures 14, 15, and 16 show the average path lengths in the successful routings in B_{11}, B_{12}, and B_{13}, respectively. As shown in the figures, if the ratio of faulty nodes is less than or equal to 0.3, the performance of our proposed methods is almost same as Simple. However, if the ratio of faulty nodes is greater than 0.3, the average path lengths of Method2 and Method3 are longer than those of Simple. This is because Method2 and Method3 can find a fault-free path even if the ratio of faulty nodes becomes higher. Hence, these methods find longer paths on average. In addition, Method3 can construct a slightly shorter fault-free path than Method2 because Method3 incorporates the depth-first search. Compared to Method2, Method3 finds shorter paths on average by at most 0.214 in B_{11}, 0.141 in B_{12}, and 0.235 in B_{13}, respectively.

```
procedure Method3(s, t)
/*
** s: node that has the message
** t: destination node
*/
p := s; /* p: previous node */
while s <> t do
  Fwd := (Pre(s,t) \ {p}) \ F;
  Bwd := ((N(s) \ Pre(s,t)) \ {p}) \ F;
  p := s;
  if d(s,t) = 3 then begin
    execute DFS to find w ∈ Fwd
      that ensures message delivery;
    if w exists then s := w
    else if Bwd <> ∅ then select s from Bwd
    else error ('message delivery failed') end
  else if Fwd <> ∅ then select s from Fwd
  else if Bwd <> ∅ then select s from Bwd
  else error ('message delivery failed')
end
```

Fig. 10 Fault-tolerant routing algorithm Method3.

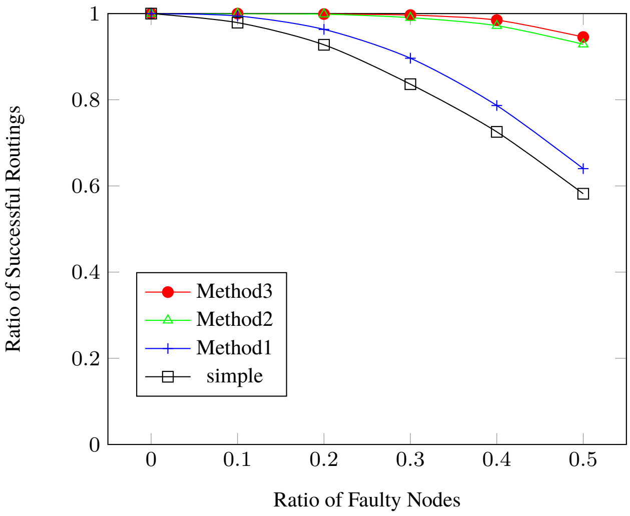

Fig. 11 Ratio of successful routings in BQ_{11}.

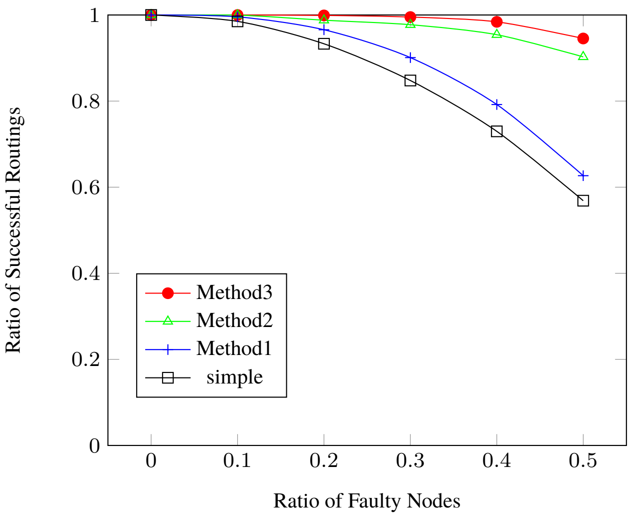

Fig. 12 Ratio of successful routings in BQ_{12}.

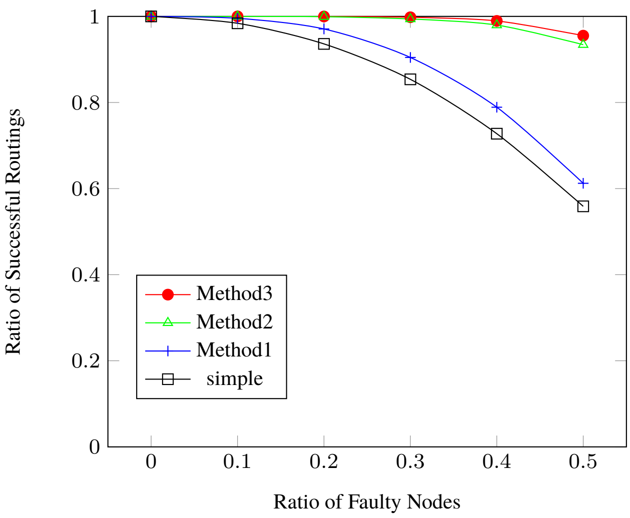

Fig. 13 Ratio of successful routings in BQ_{13}.

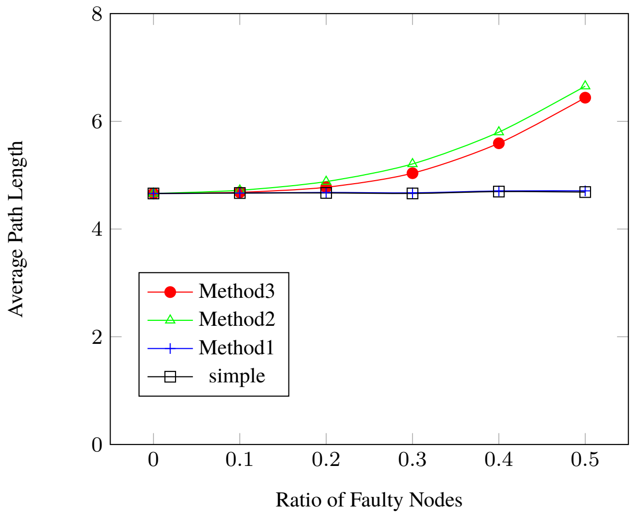

Fig. 14 Average path lengths in BQ_{11}.

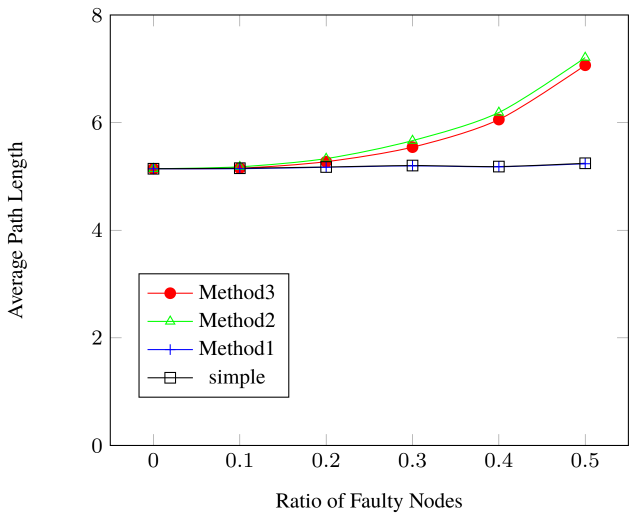

Fig. 15 Average path lengths in BQ_{12}.

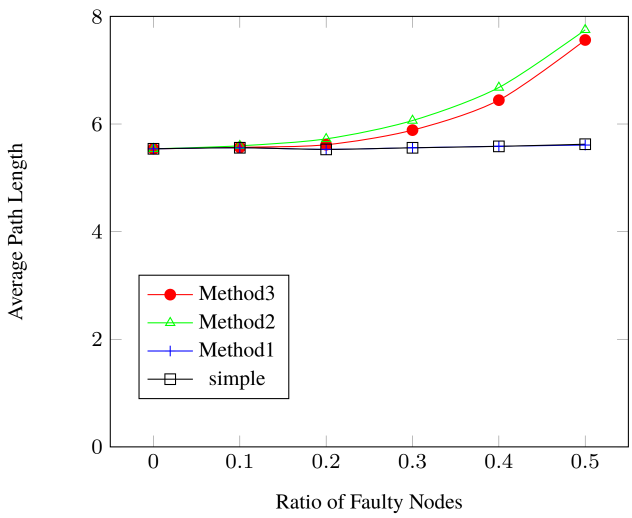

Fig. 16 Average path lengths in BQ_{13}.

5. Conclusions

In this paper, we proposed three methods for fault-tolerant routing in the bicube.

The first method, Method1, performs the depth-first search when the message is stored in the node that is 3 hops away from the destination node. The second method, Method2, refrains from forwarding the message stored in a node to its previous node. The third method, Method3, is obtained by combining Method1 and Method2.

Also, we have adopted the simple fault-tolerant routing method, Simple, as the baseline and carried out a computer experiment in B_{11}, B_{12}, and B_{13} with the ratios of faulty nodes $\alpha = 0.1, 0.2, \ldots, 0.5$.

As a result, Method1 showed better ratios of successful routings than the baseline method, Simple, by at most 0.0612 in B_{11}, 0.0623 in B_{12}, and 0.0615 in B_{13}, respectively. Method2 showed better ratios than Simple by at most 0.347 in B_{11}, 0.3342 in B_{12}, and 0.3755 in B_{13}, respectively. Also, Method3 showed better ratios than Simple by at most 0.3637 in B_{11}, 0.3769 in B_{12}, and 0.3965 in B_{13}, respectively. Among the three methods, Method2 and Method3 showed almost the same performances, which are rather better than that of Method1.

As a future work, we should investigate the cases in which the message deliveries fail even if Method3 is used, and propose another method by which they can be avoided.

Acknowledgment

The authors appreciate the reviewers for their proceleusmatic comments and hortative suggestions. This study was partly supported by a Grant-in-Aid for Scientific Research (C) of the Japan Society for the Promotion of Science under Grant No. 23K11029.

References

[1] Y. Wang, H. H. S. Kyaw, and K. Kaneko, "Fault-tolerant routing methods in bicubes," in *Proceedings of the 2023 International Conference on Parallel and Distributed Processing Techniques and Applications*, pp. 2093–2100, July 2023.
[2] H.-S. Lim, J.-H. Park, and H.-C. Kim, "The bicube: An interconnection of two hypercubes," *International Journal of Computer Mathematics*, vol. 92, pp. 29–40, Jan. 2015.
[3] C. L. Seitz, "The cosmic cube," *Communications of the ACM*, vol. 28, pp. 22–33, Jan. 1985.
[4] Y.-H. Chen, S.-M. Tang, K.-J. Pai, and J.-M. Chang, "Constructing dual-cists with short diameters using a generic adjustment scheme on bicubes," *Theoretical Computer Science*, vol. 878-879, pp. 102–112, 2021.
[5] J. Liu, S. Zhou, Z. Gu, Q. Zhou, and D. Wang, "Fault diagnosability of bicube networks under the PMC diagnostic model," *Theoretical Computer Science*, vol. 851, pp. 14–23, 2021.
[6] H. Zhuang, W. Guo, X.-Y. Li, X. Liu, and C.-K. Lin, "The component connectivity, component diagnosability, and t/k-diagnosability of bicube networks," *Theoretical Computer Science*, vol. 896, pp. 145–157, 2021.
[7] J. Liu, S. Zhou, E. Cheng, G. Chen, and M. Li, "Reliability evaluation of bicube-based multiprocessor system under the g-good-neighbor restriction," *Parallel Processing Letters*, vol. 31, no. 04, p. 2150018, 2021.
[8] M. Okada and K. Kaneko, "Minimal paths in a bicube," *IEICE Transactions on Information and Systems*, vol. E105-D, pp. 1383–1392, Aug. 2022.
[9] P. Cull and S. M. Larson, "The Möbius cubes," *IEEE Transactions on Computers*, vol. 44, pp. 647–659, May 1995.
[10] K. Efe, "A variation on the hypercube with lower diameter," *IEEE Transactions on Computers*, vol. 40, pp. 1312–1316, Nov. 1991.
[11] P. A. J. Hilbers, M. R. Koopman, and J. L. A. van de Snepscheut, "The twisted cube," in *Volume I: Parallel Architectures on PARLE: Parallel Architectures and Languages Europe*, (London, UK), pp. 152–159, Springer-Verlag, 1987.
[12] X. Wang, J. Liang, D. Qi, and W. Lin, "The twisted crossed cube," *Concurrency and Computation: Practice and Experience*, vol. 28, pp. 1507–1526, 2016.
[13] X. Yang, D. J. Evans, and G. M. Megson, "The locally twisted cubes," *International Journal of Computer Mathematics*, vol. 82, pp. 401–413, Apr. 2005.
[14] W. J. Zhou, J. X. Fan, X. H. Jia, and S. K. Zhang, "The spined cube: a new hypercube variant with smaller diameter," *Information Processing Letters*, vol. 111, pp. 561–567, June 2011.

Yitong Wang is a Ph.D. program student at Tokyo University of Agriculture and Technology in Japan. Her main research areas are interconnection networks and fault-tolerant systems based on graph theory and network theory. She received the B.E. degree from Dalian University of Foreign Languages in China in 2019 and the M.E. degree from Tokyo University of Agriculture and Technology in 2023.

Htoo Htoo Sandi Kyaw is an Assistant Professor at Tokyo University of Agriculture and Technology in Japan. Her main research areas are educational technology, web application systems, and graph theory. She received the B.E. and M.E. degrees from University of Technology (Yatanarpon Cyber City) in Myanmar in 2015 and 2018, respectively, and the Ph.D. degree from Okayama University in Japan in 2021.

Kunihiro Fujiyoshi is an Associate Professor at Tokyo University of Agriculture and Technology in Japan. His main research interests are in combinatorial algorithms and VLSI layout design. He received the B.E., M.E., and D.E. degrees from Tokyo Institute of Technology in 1987, 1989, and 1994, respectively. He is a member of IEEE and IPSJ.

Keiichi Kaneko is a Professor at Tokyo University of Agriculture and Technology in Japan. His main research areas are functional programming, parallel and distributed computation, partial evaluation and fault-tolerant systems. He received the B.E., M.E., and Ph.D. degrees from the University of Tokyo in 1985, 1987, and 1994, respectively. He is a member of ACM, IEEE CS, IPSJ and JSSST.