An Algorithm for Set-to-Set Disjoint Paths Problem in a Möbius Cube

Keiichi Kaneko
Institute of Engineering
Tokyo University of Agriculture and Technology
Koganei-shi, Tokyo 184-8588, Japan
Email: k1kaneko@cc.tuat.ac.jp

***Abstract*—In this paper, we propose an algorithm that solves the set-to-set disjoint paths problem in n-möbius cubes in polynomial-order time of n. We also give a proof of its correctness as well as the estimates of time complexity $O(n^6)$ and the maximum path length $4n-7$.**

Keywords: hypercube, multicomputer, interconnection network, parallel processing

I. Introduction

Recently, researches on parallel processing, especially massively parallel systems are performed very actively. Because a massively parallel system connects many nodes, it is important to interconnect the nodes efficienlty. Therefore, many topologies for interconnection networks have been proposed [1], [2] and studied [3], [4], [5], [6], [7], [8], [9], [10] to replace simple interconnection networks such as a ring, a mesh, a torus and a hypercube [11]. A möbius cube [12] is one such new topology. It has attracted much attention because it can connect the same number of nodes as a hypercube while keeping its diameter about half of that of the hypercube [13], [14], [15], [16], [17], [18].

The unsolved problems in möbius cubes include the set-to-set disjoint paths problem: given a set of source nodes $S = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_k\}$ and a set of destination nodes $D = \{\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_k\}$ in a k-connected graph $G = (V, E)$, find k paths P_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{j_i}$ $(1 \le i \le n)$ such that $\{j_1, j_2, \ldots, j_k\} = \{1, 2, \ldots, k\}$ and the paths P_i's are node-disjoint. The set-to-set disjoint paths problem is an important issue in parallel and distributed computation [19], [20], [21], [22] as well as the node-to-node disjoint paths problem [23], [24], [25], [26], [27] and the node-to-set disjoint paths problem [28], [29], [30], [31], [15].

For a graph $G(V, E)$, by using the maximum flow algorithm, the set-to-set disjoint paths can be obtained in polynomial-order time of $|V|$ in general. However, the complexity of the algorithm is too large for an n-dimensional möbius cube or an M_n in short because the number of nodes in it is equal to 2^n. In this paper, we propose an algorithm called S2S (set-to-set) which has a polynomial-order time of n instead of 2^n. Algorithm S2S consists of three cases according to the relative positions of the source nodes and the destination nodes. The algorithm obtains n disjoint paths between the set of the source nodes S and the set of the destination nodes D where $|S| = |V| = n$ and n is equal to the connectivity of an M_n.

The rest of this paper is organized as follows. Section 2 introduces the definition of möbius cubes as well as other requisite definitions. Section 3 explains our algorithm S2S in detail. Section 4 describes a proof of correctness and the theoretical complexities of S2S. We conclude and give future works in Section 5.

II. Preliminaries

In this section, we first introduce a definition of a möbius cube followed by several lemmas.

Definition *1:* An n-dimensional möbius cube M_n has 2^n nodes. Each node has a unique n-bit address. For two nodes $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ and $\boldsymbol{y}$, they are connected if and only if one of the following conditions are satisfied:

$$\boldsymbol{y} = \begin{cases} (x_1, x_2, \ldots, x_{i-1}, \bar{x}_i, x_{i+1}, \ldots, x_n) & \text{if } x_{i-1} = 0, \\ (x_1, x_2, \ldots, x_{i-1}, \bar{x}_i, \bar{x}_{i+1}, \ldots, \bar{x}_n) & \text{if } x_{i-1} = 1. \end{cases}$$

where we can assume that $x_0 = 0$ or $x_0 = 1$. For the former case, we call it a 0-M_n while for the latter a 1-M_n.

If two nodes $\boldsymbol{x}$ and $\boldsymbol{y}$ are connected by one of the conditions in Definition 1, we say that $\boldsymbol{x}$ and $\boldsymbol{y}$ are connected by an edge of i-th dimension, and $\boldsymbol{y}$ is denoted by $\boldsymbol{x}^{(i)}$ or $\boldsymbol{x}$ is denoted by $\boldsymbol{y}^{(i)}$.

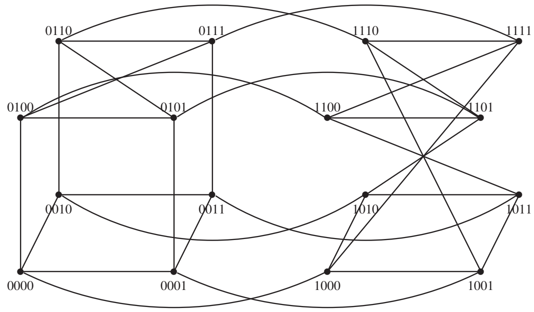
Figure 1 shows examples of a 0-M_4 and a 1-M_4. An M_n consists of two disjoint subgraphs M^0 and M^1 where M^i is derived from the set of nodes $\{\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \mid x_1 = i\}$. Note that an M^0 and an M^1 are isomorphic to a 0-M_{n-1} and 1-M_{n-1}, respectively.

Table I shows a comparison of characteristics of an n-dimensional 0-möbius cube, 0-M_n, and an n-dimensional 1-möbisu cube, 1-M_n, with an n-dimensional hypercube, H_n and an n-dimensional twisted hypercube, T_n. With respect to the diameter, a T_n is a slightly better than 0-M_n. However, a T_n is much inferior to a 0-M_n and a 1-M_n with respect to the average distance.

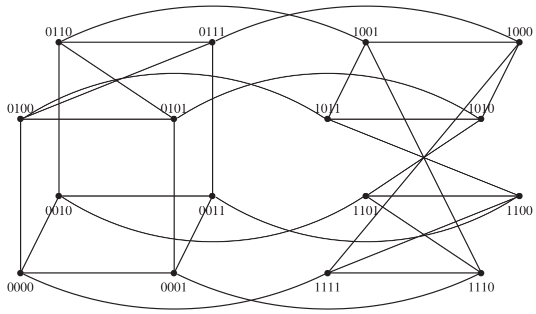
TABLE I. Comparison of a 0-M_n and a 1-M_n with other topologies.

	#nodes	degree	diameter	average distance
0-M_n	2^n	n	$\lceil (n+2)/2 \rceil$	†
1-M_n	2^n	n	$\lceil (n+1)/2 \rceil$	†
H_n	2^n	n	n	$n/2$
T_n	2^n	n	$\lfloor (n+1)/2 \rfloor$	$\to 3n/8$ $(n \to \infty)$

†: $\le n/3 + [1 - (-1/2)^n]/9 + 1$ [12]



(a) 0-M_4



(b) 1-M_4

Fig. 1. Examples of a 0-M_4 and a 1-M_4.

There is a shortest-path routing algorithm for an M_n and it takes $O(n)$ time [12]. In the rest of this paper, we refer the algorithm `spr`.

We assume that each node is stored in a machine word, and construction of an edge by obtaining $\boldsymbol{a}^{(i)}$ for any node $\boldsymbol{a}$ requires $O(1)$ time. On the other hand, `spr` takes $O(n)$ execution time to construct a shortest path whose length is at most $\lceil (n+2)/2 \rceil$ [12].

Lemma *1:* There is no cycle whose length is 3 in an M_n.
(Proof) Assume that there is a cycle C of length 3. Let $\boldsymbol{x} \to \boldsymbol{x}^{(i)} \to \boldsymbol{x}^{(j)} \to \boldsymbol{x}$ $(i > j)$ be the cycle C. Then, the i-th bit of $\boldsymbol{x}^{(i)}$ is $\bar{x}_i$ while the i-th bit of $\boldsymbol{x}^{(j)}$ is x_i from $i > j$. Then, to revert the i-th bit of $\boldsymbol{x}^{(i)}$ without changing the 1st to $(i-1)$-th bits, it is necessary to perform the operation of $\boldsymbol{x}^{(i,i)}$. Therefore, $\boldsymbol{x}^{(j)} = \boldsymbol{x}^{(i,i)} = \boldsymbol{x}$ must hold, and it contradicts that the C is a cycle. Therefore, there is no cycle whose length is 3.

Lemma *2:* For an arbitrary node $\boldsymbol{x}$ in an M_n, there is exactly one edge between M^0 and M^1 including $\boldsymbol{x}$ as one of its terminal nodes.
(Proof) For a node $\boldsymbol{x}$, $\boldsymbol{x}^{(i)}$ $(2 \le i \le n)$ belongs to the sub graph that includes $\boldsymbol{x}$. On the other hand, $\boldsymbol{x}^{(1)}$ belongs to the sub graph that does not include $\boldsymbol{x}$. Therefore, there is exactly one edge between M^0 and M^1 including $\boldsymbol{x}$ as one of its terminal nodes.

Lemma *3:* For an arbitrary node $\boldsymbol{x}$ in an M_n, there are n disjoint paths except for $\boldsymbol{x}$ of length at most 2 between M^0 and M^1 including $\boldsymbol{x}$ as one of its terminal nodes.

(Proof) Let us consider n paths that include $\boldsymbol{x}$ as one of its terminal nodes and span between M^0 and M^1.

$$Q_i\colon \begin{cases} \boldsymbol{x} \to \boldsymbol{x}^{(i)} \to \boldsymbol{x}^{(i,1)} & (2 \le i \le n) \\ \boldsymbol{x} \to \boldsymbol{x}^{(i)} & (i = 1) \end{cases}$$

where from Lemma 2, Q_1 is disjoint with other paths Q_i $(2 \le i \le n)$ except for $\boldsymbol{x}$. Moreover, for two paths Q_i and Q_j $(2 \le i < j \le n)$, from $\boldsymbol{x}^{(i)} \ne \boldsymbol{x}^{(j)}$ and Lemma 1, these paths are also disjoint except for $\boldsymbol{x}$ (Figure 2). From above discussion, the n paths Q_i $(1 \le i \le n)$ of length at most 2 are disjoint except for $\boldsymbol{x}$.

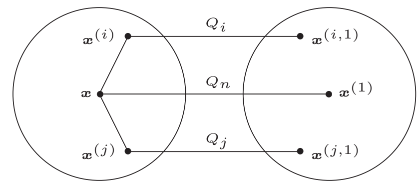


Fig. 2. Disjoint paths between M^0 and M^1.

III. Algorithm S2S

In this section, for a set of source nodes $S = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_n\}$ and a set of destination nodes $D = \{\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_n\}$ in an M_n, we show an algorithm S2S that finds n disjoint paths between S and D.

A. Case 1 ($S \cup D \subset M^j$ $(j \in \{0,1\})$)

In this case, we construct n disjoint paths R_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \le i \le n, \{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\})$ by the following Procedure 1.
Step 1 In the M^j, apply Algorithm S2S recursively to construct $(n-1)$ disjoint paths P_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \le i \le n-1, \{k_1, k_2, \ldots, k_{n-1}\} = \{1, 2, \ldots, n-1\})$.
Step 2 If $\boldsymbol{s}_n$ is included in one of the paths P_i $(1 \le i \le n-1)$, say P_l, discard the sub path $\boldsymbol{s}_l \rightsquigarrow \boldsymbol{s}_n$, let P_l be $\boldsymbol{s}_n \rightsquigarrow \boldsymbol{d}_{k_l}$, and exchange the indices of $\boldsymbol{s}_l$ and $\boldsymbol{s}_n$.
Step 3 If $\boldsymbol{d}_n$ is included in one of the paths P_i $(1 \le i \le n-1)$, say P_m, discard the sub path $\boldsymbol{d}_n \rightsquigarrow \boldsymbol{d}_{k_m}$, let P_m be $\boldsymbol{s}_m \rightsquigarrow \boldsymbol{d}_n$, and exchange the indices of $\boldsymbol{d}_{k_m}$ and $\boldsymbol{d}_n$. See Figure 3.

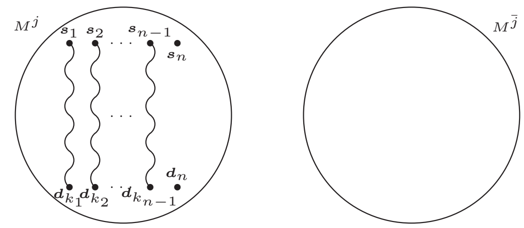


Fig. 3. After Step 3 in Procedure 1

Step 4 Select edges $\boldsymbol{s}_n \to \boldsymbol{s}_n^{(1)}$ and $\boldsymbol{d}_n \to \boldsymbol{d}_n^{(1)}$.
Step 5 In the $M^{\bar{j}}$, construct a path P_n: $\boldsymbol{s}_n^{(1)} \rightsquigarrow \boldsymbol{d}_n^{(1)}$ by using the algorithm `spr`. Finally, n disjoint paths R_i $(1 \le i \le n)$

are obtained:

$$R_i: \begin{cases} s_i \overset{P_i}{\rightsquigarrow} d_{k_i} & \text{if } 1 \le i \le n-1, \\ s_n \to s_n^{(1)} \overset{P_n}{\rightsquigarrow} d_n^{(1)} \to d_n & \text{if } i = n. \end{cases}$$

See Figure 4.

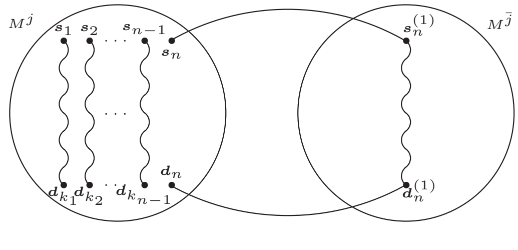


Fig. 4. After Step 5 in Procedure 1

B. Case 2 ($0 < |S \cap M^j| < n$ *or* $0 < |D \cap M^{\bar{j}}| < n$ ($j \in \{0,1\}$))

In this case, we construct n disjoint paths R_i: $s_i \rightsquigarrow d_{k_i}$ ($1 \le i \le n$, $\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}$) by the following Procedure 2. We can assume that $S \cap M^j = \{s_1, s_2, \ldots, s_g\}$, $D \cap M^j = \{d_1, d_2, \ldots, d_h\}$ and $h > g$ without loss of generality.

Step 1 For g source nodes $s_1, s_2, \ldots, s_g$ and g destination nodes $d_1, d_2, \ldots, d_g$ in the M^j, apply Algorithm S2S recursively to construct g disjoint paths P_i: $s_i \rightsquigarrow d_{k_i}$ ($1 \le i \le g$, $\{k_1, k_2, \ldots, k_g\} = \{1, 2, \ldots, g\}$).

Step 2 For each node d_i ($g+1 \le i \le h$), do the following. First, construct n paths $Q_{i,l}$: $d_i \rightsquigarrow d'_i (\in M^{\bar{j}})$ ($1 \le l \le n$) from Lemma 3. If among them exists a path Q_{i,l_i^*} ($1 \le l_i^* \le n$) that does not include any node of P_i ($1 \le i \le g$) nor any node of $Q_{i',l_{i'}^*}$ ($g+1 \le i' < i$), select it. Otherwise, find one of the paths P_i ($1 \le i \le g$), say P_x, that includes multiple nodes of one of $Q_{i,l}$ ($1 \le l \le n$). Then, find the node $d_i^{(y)}$ that is closest to s_x along P_x, discard the sub path $d_i^{(y)} \rightsquigarrow d_{k_x}$, let P_x be $s_x \rightsquigarrow d_i^{(y)} \to d_i$, and exchange the indices of d_i and d_{k_x}. Repeat this process until we can find $(h-g)$ paths Q_{i,l_i^*} ($g+1 \le i \le h$). See Figure 5.

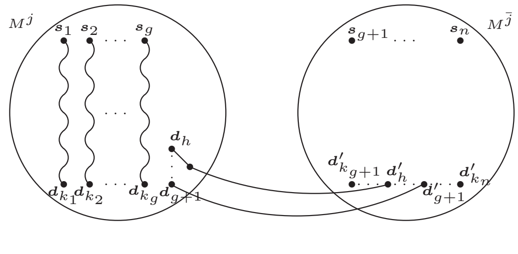


Fig. 5. After Step 2 in Procedure 2

Step 3 Let d'_i ($h+1 \le i \le n$) be d_i. Then, for $(n-g)$ source nodes $s_{g+1}, s_{g+2}, \ldots, s_n$ and $(n-g)$ destination nodes $d'_{g+1}, d'_{g+2}, \ldots, d'_n$ in $M^{\bar{j}}$, apply Algorithm S2S recursively to construct $(n-g)$ paths P_i: $s_i \rightsquigarrow d'_{k_i}$ ($g+1 \le i \le n$). Finally, n disjoint paths R_i ($1 \le i \le n$) are obtained:

$$R_i: \begin{cases} s_i \overset{P_i}{\rightsquigarrow} d_{k_i} & \text{if } 1 \le k_i \le g, \\ s_i \overset{P_i}{\rightsquigarrow} d'_{k_i} \overset{Q_{k_i, l_{k_i}^*}}{\rightsquigarrow} d_{k_i} & \text{if } g+1 \le k_i \le h, \\ s_i \overset{P_i}{\rightsquigarrow} d'_{k_i} (= d_{k_i}) & \text{if } h+1 \le k_i \le n. \end{cases}$$

See Figure 6.

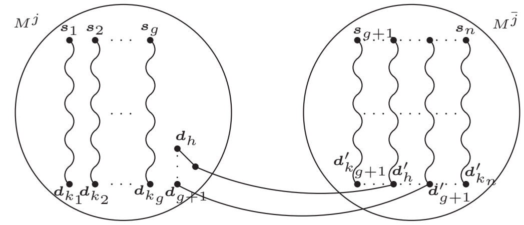


Fig. 6. After Step 3 in Procedure 2

C. Case 3 ($S \subset M^j$, $D \subset M^{\bar{j}}$)

In this case, we construct n disjoint paths P_i: $s_i \rightsquigarrow d_{k_i}$ ($1 \le i \le n$, $\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}$) by considering four sub Möbius cubes: $M^{j_1 j_2}$, $M^{j_1 \bar{j}_2}$, $M^{\bar{j}_1 j_2}$, and $M^{\bar{j}_1 \bar{j}_2}$ ($j_1, j_2 \in \{0,1\}$).

1) Case 3-1: First, we assume a 0-M_n and a case that $|S \cap M^{j_1 j_2}| \ge |S \cap M^{j_1 \bar{j}_2}|$ and $|D \cap M^{\bar{j}_1 j_2}| \ge |D \cap M^{\bar{j}_1, \bar{j}_2}|$. Without loss of generality, we can assume that $S \cap M^{j_1 j_2} = \{s_1, s_2, \ldots, s_{n_s}\}$ and $D \cap M^{\bar{j}_1 j_2} = \{d_1, d_2, \ldots, d_{n_d}\}$ where $n_s \ge n - n_s$ and $n_d \ge n - n_d$. Then, we construct n disjoint paths P_i: $s_i \rightsquigarrow d_{k_i}$ ($1 \le i \le n$, $\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}$) by the following Procedure 3.

Step 1 Find $(n_s - \lceil n/2 \rceil)$ nodes s_i in $S \cap M^{j_1 j_2}$ such that $s_i^{(2)} \notin S$. We can assume that $\{s^{(2)}_{\lceil n/2 \rceil + 1}, s^{(2)}_{\lceil n/2 \rceil + 2}, \ldots, s^{(2)}_{n_s}\} \cap S = \emptyset$.

Step 2 Select $(n_s - \lceil n/2 \rceil)$ paths $s_i \to s_i^{(2)} \to s_i^{(2,1)} = s'_i$ ($\lceil n/2 \rceil + 1 \le i \le n_s$) and $(n - n_s)$ edges $s_i \to s_i^{(1)} = s'_i$ ($n_s + 1 \le i \le n$).

Step 3 Select edges $s_i \to s_i^{(1)}$ ($1 \le i \le \lceil n/2 \rceil$). See Figure 7.

Step 4 Apply S2S in $M^{\bar{j}_1 j_2}$ to obtain $\lceil n/2 \rceil$ disjoint paths P_i: $s_i^{(1)} \rightsquigarrow d_{k_i}$ ($1 \le i \le \lceil n/2 \rceil$, $k_{i_1} \ne k_{i_2}$ ($i_1 \ne i_2$)). Without loss of generality, we can assume that $d_{k_i} = d_i$ ($1 \le i \le \lceil n/2 \rceil$).

Step 5 For d_i ($\lceil n/2 \rceil + 1 \le i \le n_d$), construct disjoint paths $d_i \rightsquigarrow d'_i (\in M^{\bar{j}_1 \bar{j}_2})$ of lengths at most 2 as in Case 1. See Figure 8.

Step 6 Apply S2S in $M^{\bar{j}_1 \bar{j}_2}$ to obtain $\lfloor n/2 \rfloor$ disjoint paths P_i: $s'_i \rightsquigarrow d_{k_i}$ ($\lceil n/2 \rceil + 1 \le i \le n$, $\{k_{\lceil n/2 \rceil + 1}, k_{\lceil n/2 \rceil + 2}, \ldots, k_n\} = \{\lceil n/2 \rceil + 1, \lceil n/2 \rceil + 2, \ldots, n\}$).

2) Case 3-2: We assume that a 0-M_n and a case that $|S \cap M^{j_1 j_2}| \ge |S \cap M^{j_1 \bar{j}_2}|$ and $|D \cap M^{\bar{j}_1 j_2}| < |D \cap M^{\bar{j}_1 \bar{j}_2}|$. Without loss of generality, we can assume that $S \cap M^{j_1 j_2} = \{s_1, s_2, \ldots, s_{n_s}\}$ and $D \cap M^{\bar{j}_1 j_2} = \{d_1, d_2, \ldots, d_{n_d}\}$ where $n_s \ge n - n_s$ and $n_d < n - n_d$. Then, we construct n disjoint paths P_i: $s_i \rightsquigarrow d_{k_i}$ ($1 \le i \le n$, $\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}$ by following Procedure 4.

Step 1 Find $(n_s - \lceil n/2 \rceil)$ nodes s_i in $S \cap M^{j_1 j_2}$ such that $s_i^{(2)} \notin S$. We can assume that $\{s^{(2)}_{\lceil n/2 \rceil + 1}, s^{(2)}_{\lceil n/2 \rceil + 2}, \ldots, s^{(2)}_{n_s}\} \cap S = \emptyset$.

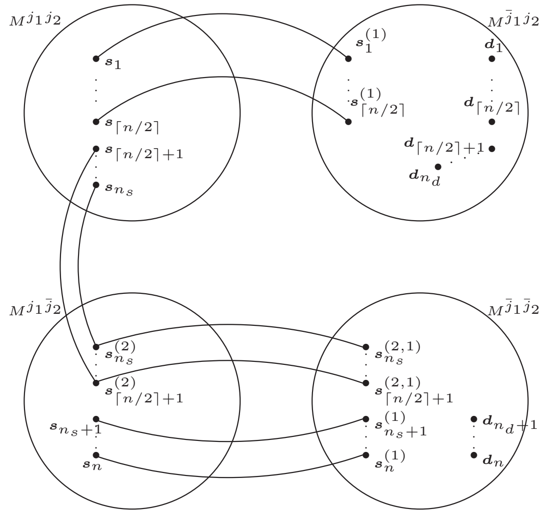


Fig. 7. After Step 3 in Procedure 3

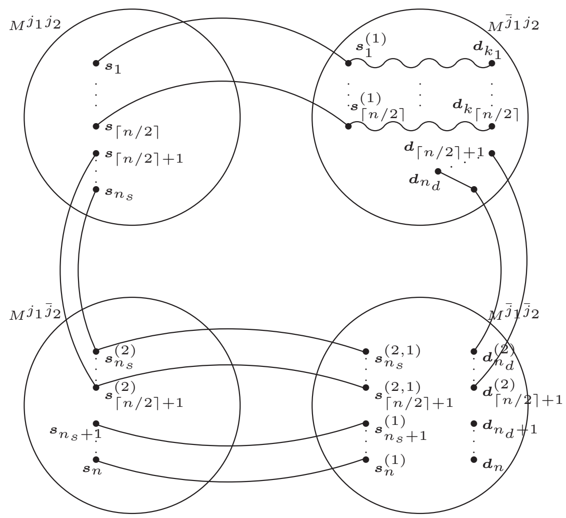


Fig. 8. After Step 5 in Procedure 3

Step 2 Select $\lceil n/2 \rceil$ edges $\boldsymbol{s}_i \to \boldsymbol{s}_i^{(1)}$ $(1 \le i \le \lceil n/2 \rceil)$.
Step 3 Select $(n_s - \lceil n/2 \rceil)$ edges $\boldsymbol{s}_i \to \boldsymbol{s}_i^{(1)}$ $(\lceil n/2 \rceil + 1 \le i \le n_s)$. Let $\boldsymbol{s}'_i$ be $\boldsymbol{s}_i^{(2)}$ if $\lceil n/2 \rceil + 1 \le i \le n$ or $\boldsymbol{s}_i$ if $n_s + 1 \le i \le n$.
Step 4 Select $(\lceil n/2 \rceil - n_d)$ nodes $\boldsymbol{d}_i$ in $D \cap M^{\bar{j}_1 \bar{j}_2}$ such that $\boldsymbol{d}_i^{(2)} \notin D$. We can assume that $\{\boldsymbol{d}_{n_d+1}^{(2)}, \boldsymbol{d}_{n_d+2}^{(2)}, \ldots, \boldsymbol{d}_{\lceil n/2 \rceil}^{(2)}\} \cap D = \emptyset$.
Step 5 Select $\lfloor n/2 \rfloor$ edges $\boldsymbol{d}_i \to \boldsymbol{d}_i^{(1)}$ $(\lceil n/2 \rceil + 1 \le i \le n)$.
Step 6 Select $(\lceil n/2 \rceil - n_d)$ edges $\boldsymbol{d}_i \to \boldsymbol{d}_i^{(2)}$ $(n_d + 1 \le i \le \lceil n/2 \rceil)$. Let $\boldsymbol{d}'_i$ be $\boldsymbol{d}_i$ if $1 \le i \le n_d$ or $\boldsymbol{d}_i^{(2)}$ if $n_d + 1 \le i \le \lceil n/2 \rceil$. See Figure 9.
Step 7 Apply S2S in $M^{\bar{j}_1 j_2}$ to obtain $\lceil n/2 \rceil$ disjoint paths P_i: $\boldsymbol{s}_i^{(1)} \rightsquigarrow \boldsymbol{d}'_{k_i}$ $(1 \le i \le \lceil n/2 \rceil$, $\{k_1, k_2, \ldots, k_{\lceil n/2 \rceil}\} = \{1, 2, \ldots, \lceil n/2 \rceil\})$.
Step 8 Apply S2S in $M^{j_1 \bar{j}_2}$ to obtain $\lfloor n/2 \rfloor$ disjoint paths P_i: $\boldsymbol{s}'_i \rightsquigarrow \boldsymbol{d}'_{k_i}$ $(\lceil n/2 \rceil + 1 \le i \le n, \{k_{\lceil n/2 \rceil+1}, k_{\lceil n/2 \rceil+2}, \ldots, k_n\} = \{\lceil n/2 \rceil + 1, \lceil n/2 \rceil + 2, \ldots, n\})$.

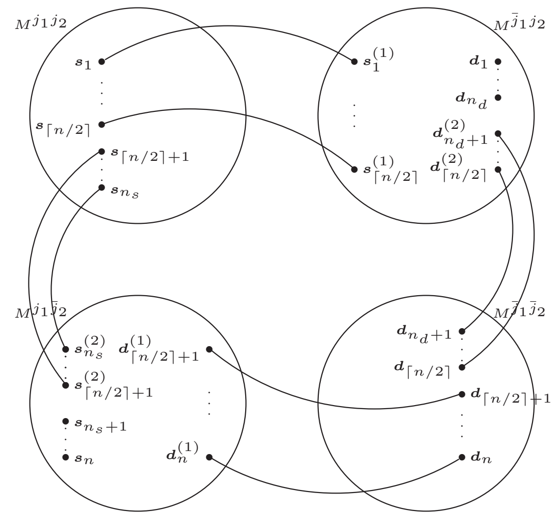


Fig. 9. After Step 6 in Procedure 4

3) Case 3-3: We assume that a 0-M_n and a case that $|S \cap M^{j_1 j_2}| < |S \cap M^{j_1 \bar{j}_2}|$ and $|D \cap M^{\bar{j}_1 \bar{j}_2}| \ge |D \cap M^{\bar{j}_1 \bar{j}_2}|$. We construct n disjoint paths P_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \le i \le n$, $\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}$ by the Procedure 5 that is similar to Procedure 4.

4) Case 3-4: We assume that a 0-M_n and a case that $|S \cap M^{j_1 j_2}| < |S \cap M^{j_1 \bar{j}_2}|$ and $|D \cap M^{\bar{j}_1 \bar{j}_2}| < |D \cap M^{\bar{j}_1 \bar{j}_2}|$. We construct n disjoint paths P_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \le i \le n$, $\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}$ by the Procedure 6 that is similar to Procedure 3.

5) Case 3-5: We assume that a 1-M_n and a case that $|S \cap M^{j_1 j_2}| \ge |S \cap M^{j_1 \bar{j}_2}|$ and $|D \cap M^{\bar{j}_1 \bar{j}_2}| \ge |D \cap M^{\bar{j}_1 \bar{j}_2}|$. We construct n disjoint paths P_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \le i \le n$, $\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}$ by the Procedure 7 that is similar to Procedure 3.

6) Case 3-6: We assume that a 1-M_n and a case that $|S \cap M^{j_1 j_2}| \ge |S \cap M^{j_1 \bar{j}_2}|$ and $|D \cap M^{\bar{j}_1 \bar{j}_2}| < |D \cap M^{\bar{j}_1 \bar{j}_2}|$. We construct n disjoint paths P_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \le i \le n$, $\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}$ by the Procedure 8 that is similar to Procedure 4.

7) Case 3-7: We assume that a 1-M_n and a case that $|S \cap M^{j_1 j_2}| < |S \cap M^{j_1 \bar{j}_2}|$ and $|D \cap M^{\bar{j}_1 \bar{j}_2}| \ge |D \cap M^{\bar{j}_1 \bar{j}_2}|$. We construct n disjoint paths P_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \le i \le n$, $\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}$ by the Procedure 9 that is similar to Procedure 4.

8) Case 3-8: We assume that a 1-M_n and a case that $|S \cap M^{j_1 j_2}| < |S \cap M^{j_1 \bar{j}_2}|$ and $|D \cap M^{\bar{j}_1 \bar{j}_2}| < |D \cap M^{\bar{j}_1 \bar{j}_2}|$. We construct n disjoint paths P_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \le i \le n$, $\{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}$ by the Procedure 10 that is similar to Procedure 3.

IV. Proof of Correctness and Estimation of Complexities

In this section, we prove the correctness of our algorithm and we give the estimates of time complexity $T(n)$ and maximum path length $L(n)$ for an n-dimensional möbius cube. Proof is based on induction on n.

Lemma *4:* In an M_n, the paths constructed by Procedure 1 are disjoint. The time complexity of Procedure 1 is $T(n-1) + O(nL(n-1))$ and the maximum length of the paths constructed is $\max\{L(n-1), \lfloor n/2 \rfloor + 3\}$.
(Proof) The paths constructed in Steps 1, 2, and 3 are disjoint by hypothesis of induction. The path constructed in Steps 4 and 5 is outside of M^j except for $\boldsymbol{s}_n$ and $\boldsymbol{d}_n$. Hence, the path cannot share any common node with the paths constructed in Steps 1, 2, and 3, that is, it is disjoint with other paths constructed in Steps 1, 2, and 3.

Step 1 takes $T(n-1)$ time to construct $(n-1)$ paths and the maximum length of them is $L(n-1)$ by hypothesis of induction. Step 2 takes $O(nL(n-1))$ time to check if $\boldsymbol{s}_n$ is included in one of P_i $(1 \leq i \leq n-1)$. Step 3 takes $O(nL(n-1))$ time, too. The path constructed in Steps 4 and 5 consists of two edges and a sub path by `spr`. Therefore, Steps 4 and 5 take $O(n)$ time to construct a path whose length is at most $2 + \lceil (n+1)/2 \rceil = \lfloor n/2 \rfloor + 3$. Hence, the time complexity of Procedure 1 is $T(n-1) + O(nL(n-1))$ and the maximum path length is $\max\{L(n-1), \lfloor n/2 \rfloor + 3\}$. □

Lemma *5:* In Step 2 of Case 2, for each node $\boldsymbol{d}_i$ $(g+1 \leq i \leq h)$, we can find a path Q_{i,l_i^*}.
(Proof) There are n candidate paths $Q_{i,l}$: $\boldsymbol{d}_i \rightsquigarrow \boldsymbol{d}'_i (\in M^{\bar{j}})$ $(1 \leq l \leq n)$. The path $Q_{i',l_{i'}^*}$ for a node $\boldsymbol{d}_{i'}$ can block at most one of the n candidate paths $Q_{i,l}$'s from Lemma 1. In addition, each of the nodes $\boldsymbol{d}_{h+1}, \boldsymbol{d}_{h+2}, \ldots, \boldsymbol{d}_n$ can block at most one of the n candidate paths $Q_{i,l}$. Therefore, if all of the n candidate paths are blocked, at least one of the g disjoint paths P_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \leq i \leq g)$, say P_x, must block multiple candidate paths. Assume that the paths Q_{i,l_1}: $\boldsymbol{d}_i \to \boldsymbol{d}_i^{(l_1)} \to \boldsymbol{d}_i^{(l_1,1)}$ and Q_{i,l_2}: $\boldsymbol{d}_i \to \boldsymbol{d}^{(l_2)} \to \boldsymbol{d}_i^{(l_2,1)}$ are blocked by P_x (Figure 10). Then, from Lemma 1, it never happens that P_x: $\boldsymbol{s}_x \rightsquigarrow \boldsymbol{d}_i^{(l_1)} \to \boldsymbol{d}_{k_x} (= \boldsymbol{d}_i^{(l_2)})$. Hence, the path $\boldsymbol{s}_x \rightsquigarrow \boldsymbol{d}_i^{(l_1)} \to \boldsymbol{d}_i$ is strictly shorter than P_x. Therefore, if we replace P_x by the former path and repeat this process, we can find a path Q_{i,l_i^*}. The total number of replacement is restricted by the total lengths of paths P_i's.

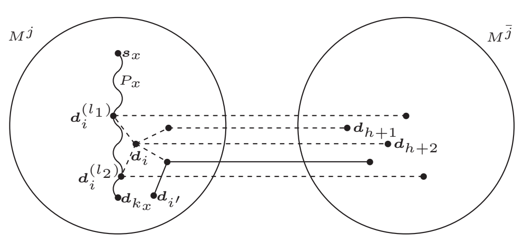


Fig. 10. Replacement of P_x in Lemma 5

Lemma *6:* In an M_n, the paths constructed by Procedure 2 are disjoint. The time complexity of Procedure 2 is $O(n^3L^2(n-1))$ and the maximum length of the paths constructed is $L(n-1)+2$.
(Proof) The g paths constructed in Step 1 are disjoint by hypothesis of induction. The $(h-g)$ paths constructed in Step 2 are disjoint with other paths constructed in Steps 1 and 2. The $(n-h)$ paths constructed in Step 3 are disjoint each other. Also, they are disjoint with the paths constructed in Step 2 except for nodes $\boldsymbol{d}'_{g+1}, \boldsymbol{d}'_{g+2}, \ldots, \boldsymbol{d}'_h$ and disjoint with the paths constructed in Step 1 since they are included in $M^{\bar{j}}$.

The time complexity for constructing the g paths of lengths at most $L(n-1)$ in Step 1 is $T(g)$. In Step 2, it takes $O(n^2L(n-1))$ time complexity to check if a path of $Q_{i,l}$ is available or not. This check is repeated at most $O(nL(n-1))$ times from Lemma 5. Hence, it takes $O(n^3L^2(n-1))$ time in Step 2 to find the $(h-g)$ paths Q_{i,l_i^*} $(g+1 \leq i \leq h)$ of lengths at most 2. Step 3 takes $T(n-g)$ time and the maximum path length is $L(n-1)$.

Hence, in total, it takes $T(g)+T(n-g)+O(n^3L^2(n-1))$ time to construct the n paths R_i $(1 \leq i \leq n)$ whose lengths are at most $L(n-1)+2$. □

Lemma *7:* In a 0-M_n, the paths constructed by Procedure 3 are disjoint. The time complexity of Procedure 3 is $T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + O(n^3L^2(n-1))$ and the maximum length of the paths constructed is $L(n-1)+4$.
(Proof) The paths constructed and the edges selected in Step 2 are disjoint each other. The $\lceil n/2 \rceil$ edges selected in Step 3 are disjoint with other constructed paths and selected edges in Step 2 since their opposite teminal nodes are all included in $M^{\bar{j}_1 j_2}$. The $\lceil n/2 \rceil$ paths constructed in Step 4 are disjoint with other paths and edges in Steps 2 and 3 by hypothesis of induction. The paths constructed in Step 5 are also disjoint with other paths and edges. The $\lfloor n/2 \rfloor$ paths constructed in Step 5 are disjoint with other paths and edges in Steps 2 to 4 since the paths are all included in $M^{\bar{j}_1 j_2}$. Therefore, the paths R_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \leq i \leq n)$ are disjoint each other.

It takes $O(n^2)$ time to find $(n_s - \lceil n/2 \rceil)$ nodes $\boldsymbol{s}_i (\in S \cap M^{j_1 j_2})$ such that $\boldsymbol{s}_i^{(2)} \not\in S$ in Step 1. Step 2 takes $O(n)$ time to construct $(n_s - \lceil n/2 \rceil)$ paths of lengths at most 2 and to select $(n - n_s)$ edges. It takes $O(n)$ time to select $(\lceil n/2 \rceil)$ edges in Step 3. Step 4 takes $T(\lceil n/2 \rceil)$ time to construct $\lceil n/2 \rceil$ disjoint paths of lengths at most $L(n-1)$. From the proof of Lemma 6, Step 5 takes $O(n^3L^2(n-1))$ time to construct $(n_d - \lceil n/2 \rceil)$ disjoint paths of lengths at most 2. In Step 6, it takes to construct $(\lfloor n/2 \rfloor)$ disjoint paths of lengths at most $L(n-1)$ in $T(\lfloor n/2 \rfloor)$ time complexity. Hence, in total, it takes $T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + O(n^3L^2(n-1))$ time to construct n disjoint paths $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \leq i \leq n,\ \{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\})$ whose lengths are at most $L(n-1)+4$. □

Lemma *8:* In a 0-M_n, the paths constructed by Procedure 4 are disjoint. The time complexity of Procedure 4 is $T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + O(n^2)$ and the maximum length of the paths constructed is $L(n-1)+2$.
(Proof) The n_s edges selected in Steps 2 and 3 are disjoint. Also, the n_d edges selected in Steps 5 and 6 are disjoint. The $\lceil n/2 \rceil$ paths constructed in Step 7 are disjoint by hypothesis of induction. They are disjoint with other edges selected in Step 3. Also, the $\lfloor n/2 \rfloor$ paths constructed in Step 8 are disjoint by hypothesis of induction. They are disjoint with other paths constructed in Step 7 and the edges selected in Step 6.

It takes $O(n^2)$ time to find $(n_s - \lceil n/2 \rceil)$ nodes $\boldsymbol{s}_i (\in S \cap M^{j_1 j_2})$ such that $\boldsymbol{s}_i^{(2)} \notin S$ in Step 1. Steps 2 and 3 takes $O(n)$ time to select edges. Similarly, it takes $O(n^2)$ time to find $(\lceil n/2 \rceil - n_d)$ nodes $\boldsymbol{d}_i (\in D \cap M^{\bar{j}_1 \bar{j}_2})$ such that $\boldsymbol{d}_i^{(2)} \notin D$ in Step 4. Steps 5 and 6 takes $O(n)$ time to select edges. Step 7 takes $T(\lceil n/2 \rceil)$ time to construct $\lceil n/2 \rceil$ paths whose lengths are at most $L(n-1)$. Step 8 takes $T(\lfloor n/2 \rfloor)$ time to construct $\lfloor n/2 \rfloor$ paths whose length are at most $L(n-1)$, too. Therefore, the time complexity of Procedure 4 is $T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + O(n^2)$ and the maximum length of the constructed paths is $L(n-1) + 2$. □

Similarly to Lemmas 7 and 8, the following lemmas are deduced.

Lemma *9:* In a 0-M_n, the paths constructed by Procedure 5 are disjoint. The time complexity of Procedure 5 is $T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + O(n^2)$ and the maximum length of the paths constructed is $L(n-1) + 2$.

Lemma *10:* In a 0-M_n, the paths constructed by Procedure 6 are disjoint. The time complexity of Procedure 6 is $T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + O(n^3 L^2(n-1))$ and the maximum length of the paths constructed is $L(n-1) + 4$.

Lemma *11:* In a 1-M_n, the paths constructed by Procedure 7 are disjoint. The time complexity of Procedure 7 is $T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + O(n^3 L^2(n-1))$ and the maximum length of the paths constructed is $L(n-1) + 4$.

Lemma *12:* In a 1-M_n, the paths constructed by Procedure 8 are disjoint. The time complexity of Procedure 8 is $T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + O(n^2)$ and the maximum length of the paths constructed is $L(n-1) + 2$.

Lemma *13:* In a 1-M_n, the paths constructed by Procedure 9 are disjoint. The time complexity of Procedure 9 is $T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + O(n^2)$ and the maximum length of the paths constructed is $L(n-1) + 2$.

Lemma *14:* In a 1-M_n, the paths constructed by Procedure 10 are disjoint. The time complexity of Procedure 10 is $T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + O(n^3 L^2(n-1))$ and the maximum length of the paths constructed is $L(n-1) + 4$.

Theorem *1:* For a set of n nodes $S = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_n\}$ and a set of n nodes $D = \{\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_n\}$ in M_n, Algorithm S2S finds n disjoint paths R_i: $\boldsymbol{s}_i \rightsquigarrow \boldsymbol{d}_{k_i}$ $(1 \le i \le n, \{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\})$. The time complexity $T(n)$ of S2S is $O(n^6)$, and the maximum path length $L(n)$ is $4n-7$.
(Proof) From Lemmas 4 to 14, the constructed paths are disjoint except. Also, $L(n) = 4n-7$ from $L(n) = L(n-1)+4$ and $L(2) = 1$. Then, $T(n) = O(n^6)$ from $T(n) = T(g) + T(n-g) + O(n^3 L^2(n-1))$ and $T(1) = O(n)$. □

V. Conclusions

In this paper, we proposed a polynomial-order time algorithm for the set-to-set disjoint paths problem in n-möbius cubes. Its time complexity is $O(n^6)$ and the maximum path length is $4n - 7$.

Future works include theoretical analysis of the maximum path length of the algorithm as well as its average performance. Also, improvement of the algorithm to construct shorter paths in smaller execution time is also interesting for us.

Acknowledgments

This study is partly supported by a Grant-in-Aid for Scientific Research (C) of the Japan Society for the Promotion of Science (JSPS) under Grant No. 17K00093.

References

[1] S. B. Akers and B. Krishnamurthy, “A group-theoretic model for symmetric interconnection networks,” *IEEE Transactions on Computers*, vol. 38, no. 4, pp. 555–566, Apr. 1989.

[2] P. F. Corbett, “Rotator graphs: An efficient topology for point-to-point multiprocessor networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 3, no. 5, pp. 622–626, May 1992.

[3] S. G. Akl and K. Qiu, “Parallel minimum spanning forest algorithms on the star and pancake interconnection networks,” in *Proceedings of Joint Conference on Vector and Parallel Processing*, 1992, pp. 565–570.

[4] ——, “A novel routing scheme on the star and pancake interconnection networks and its applications,” *Parallel Computing*, vol. 19, no. 1, pp. 95–101, Jan. 1993.

[5] S. G. Akl, K. Qiu, and I. Stojmenović, “Fundamental algorithms for the star and pancake interconnection networks with applications to computational geometry,” *Networks*, vol. 23, no. 4, pp. 215–226, Jul. 1993.

[6] P. Berthomé, A. Ferreira, and S. Perennes, “Optimal information dissemination in star and pancake networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 7, no. 12, pp. 1292–1300, Dec. 1996.

[7] L. Gardner, Z. Millter, D. Pritikin, and I. H. Sudborough, “Embedding hypercubes into pancake, cycle prefix and substring reversal networks,” in *Proceedings of the 28th Annual Hawaii International Conference on System Sciences*, Jan. 1995, pp. 537–545.

[8] L. Gargano, U. Vaccaro, and A. Vozella:, “Fault tolerant routing in the star and pancake interconnection networks,” *Information Processing Letters*, vol. 45, no. 6, pp. 315–320, 1993.

[9] W. H. Gates and C. H. Papadimitriou, “Bounds for sorting by prefix reversal,” *Discrete Mathematics*, vol. 27, pp. 47–57, 1979.

[10] K. Qiu, H. Meijer, and S. G. Akl, “Parallel routing and sorting on the pancake network,” in *Proceedings of International Conference on Computing and Information*, ser. Lecture Notes in Computer Science, vol. 497. Springer Verlag, 1991, pp. 360–371.

[11] C. L. Seitz, “The cosmic cube,” *Communications of the ACM*, vol. 28, no. 1, pp. 22–33, Jan. 1985.

[12] P. Cull and S. M. Larson, “The möbius cubes,” *IEEE Transactions on Computers*, vol. 44, no. 5, pp. 647–659, May 1995.

[13] J. Fan, “Hamilton-connectivity and cycle-embedding of the möbius cubes,” *Information Processing Letters*, vol. 82, no. 2, pp. 113–117, 2002.

[14] S.-Y. Hsieh and C.-H. Chen, “Pancyclicity on möbius cubes with maximal edge faults,” *Parallel Computing*, vol. 30, no. 3, pp. 407–421, 2004.

[15] D. Kocík, Y. Hirai, and K. Kaneko, “Node-to-set disjoint paths problem in a möbius cube,” *IEICE Transactions on Information and Systems*, vol. E99-D, no. 3, pp. 708–713, Mar. 2016.

[16] C.-H. Tsai, “Embedding of meshes in möbius cubes,” *Theoretical Computer Science*, vol. 401, no. 1-3, pp. 181–190, 2008.

[17] J.-M. Xu, M. Ma, and M. Lü, “Paths in möbius cubes and crossed cubes,” *Information Processing Letters*, vol. 97, no. 3, pp. 94–97, Feb. 2006.

[18] X. Yang, G. M. Megson, and D. J. Evans, “Pancyclicity of möbius cubes with faulty nodes,” *Microprocessors and Microsystems*, vol. 30, no. 3, pp. 165–172, May 2006.

[19] A. Bossard, “A set-to-set disjoint paths routing algorithm in hyper-star graphs,” *ISCA International Journal of Computers and Their Applications*, vol. 21, no. 1, pp. 76–82, Mar. 2014.

[20] A. Bossard and K. Kaneko, “The set-to-set disjoint-path problem in perfect hierarchical hypercubes,” *The Computer Journal*, vol. 55, no. 6, pp. 769–775, Jun. 2012.

[21] Q.-P. Gu and S. Peng, “Set-to-set fault tolerant routing in star graphs,” *IEICE Transactions on Information and Systems*, vol. E79-D, no. 4, pp. 282–289, Apr. 1996.

[22] ——, “Node-to-set and set-to-set cluster fault tolerant routing in hypercubes,” *Parallel Computing*, vol. 24, no. 8, pp. 1245–1261, 1998.

[23] M. Dietzfelbinger, S. Madhavapeddy, and I. H. Sudborough, “Three disjoint path paradigms in star networks,” in *Proceedings of the IEEE Symposium on Parallel and Distributed Processing*, Dec. 1991, pp. 400–406.

[24] Y. Hamada, F. Bao, A. Mei, and Y. Igarashi, “Nonadaptive fault-tolerant file transmission in rotator graphs,” *IEICE Transactions on Fundamentals*, vol. E79-A, no. 4, pp. 477–482, Apr. 1996.

[25] D. Kocík, Y. Hirai, and K. Kaneko, “An algorithm for node-to-node disjoint paths problem in a möbius cube,” in *Proceedings of the 2015 International Conference on Parallel and Distributed Processing Techniques and Applications*, 2015, pp. 149–155.

[26] S. Madhavapeddy and I. H. Sudborough, “A topological property of hypercubes — node disjoint paths,” in *Proceedings of the Second IEEE Symposium on Parallel and Distributed Processing*, Dec. 1990, pp. 532–539.

[27] Y. Suzuki and K. Kaneko, “An algorithm for node-disjoint paths in pancake graphs,” *IEICE Transactions on Information & Systems*, vol. E86-D, no. 3, pp. 610–615, Mar. 2003.

[28] A. Bossard and K. Kaneko, “Torus-connected cycles: an implementation-friendly topology for interconnection networks of massively parallel systems,” in *submitted to the 13th International Conference on Algorithms and Architectures for Parallel Processing*, Dec. 2013.

[29] Q.-P. Gu and S. Peng, “Node-to-set disjoint paths problem in star graphs,” *Information Processing Letters*, vol. 62, no. 4, pp. 201–207, Apr. 1997.

[30] K. Kaneko and Y. Suzuki, “An algorithm for node-to-set disjoint paths problem in rotator graphs,” *IEICE Transactions on Information and Systems*, vol. E84-D, no. 9, pp. 1155–1163, Sep. 2001.

[31] K. Kaneko, “An algorithm for node-to-set disjoint paths problem in burnt pancake graphs,” *IEICE Transactions on Information and Systems*, vol. E86-D, no. 12, pp. 2588–2594, Dec. 2003.