An Algorithm for \( k \)-pairwise Cluster-fault-tolerant Disjoint Paths in a Burnt Pancake Graph

Masato Tokuda, Yuki Hirai, and Keiichi Kaneko

Graduate School of Engineering, Tokyo University of Agriculture and Technology, Koganei-shi, Tokyo, Japan
{s149510s@st.go.yhirai@cc,k1kaneko@cc}.tuat.ac.jp

Abstract—In this paper, we focus on the pairwise cluster-fault-tolerant disjoint paths routing problem in a burnt pancake graph, and propose an algorithm that solves the problem in a polynomial time of the degree of the graph. That is, in a \( n \)-burnt pancake graph with \( (n-2k+1) \) faulty clusters whose diameters are at most 3, the algorithm can construct fault-free disjoint paths between \( k \) pairs of nodes. The time complexity of the algorithm is \( O(kn^3) \) and the maximum path length is \( 2n+13 \). We have conducted a computer experiment and its results showed that there was not any path that attained the theoretical maximum path length and the average time complexity of the algorithm is \( O(n^{2.2}) \).

Keywords: Cayley graph, multicomputer, interconnection network, parallel processing

1. Introduction

In the near future, processing performance of a sequential computer is expected to reach a ceiling because of limitations in technology. With this expectation, the field of parallel and distributed computation is taking on increasing importance, and studies on massively parallel computers are eagerly conducted recently. An interconnection network provides a topology to construct a massively parallel computer, and many topologies have been proposed and studied to interconnect many computers.

One of the factors that determine the performance of an interconnection network is fault tolerance. As the number of processors in a parallel computer increases, the probability of existence of faulty processors also increases. In practice, we face with the situation where not only the single processor fault but also a set of faulty nodes will arise. Hence, to address a fault-tolerant routing problem in a graph with multiple cluster faults has a merit to establish a fault-free communication, and there are many research activities about it. Similarly, to address a disjoint paths problem has a merit to establish full-bandwidth communication that gets no interference from other communication, and it is also studied very hard.

To solve the pairwise disjoint paths in a given graph with degree \( n \) is to find \([n/2]\) disjoint paths \( s_i \rightarrow t_i \) \((1 \leq i \leq [n/2])\) for any two arbitrary sets of nodes \( S = \{s_1,s_2,\ldots,s_{[n/2]}\} \) and \( T = \{t_1,t_2,\ldots,t_{[n/2]}\} \) in the graph. Practically, it is crucial to find disjoint paths in a network since they make it possible to use the full bandwidth and enhance communication reliability. Therefore, solving the pairwise disjoint paths problem \([1], [2], [3]\) is important as well as solving the node-to-node disjoint paths problem \([4], [5], [6], [7], [8]\), the node-to-set disjoint paths problem \([9], [10], [11], [12], [13], [14]\), and the set-to-set disjoint paths problem \([15], [16], [17]\).

In this paper, we focus on a burnt pancake graph \([18], [19], [20], [21], [5]\), which is derived from a pancake graph \([22], [23], [24], [25], [26], [27], [8]\) of a Cayley graph. A burnt pancake graph can connect many nodes with a small degree. Also, burnt pancake graphs are expected to fill in the gaps of incremental expandability of pancake graphs because they can connect different numbers of nodes from pancake graphs. However, there are many unsolved problem with a burnt pancake graph such as the shortest-path routing problem, the pairwise cluster-fault-tolerant disjoint paths routing problem, and so on.

In this paper, we pick up the pairwise cluster-fault-tolerant routing problem among the unsolved problems in a burnt pancake graph. This problem, we propose an algorithm that solves it in a polynomial time of the degree of the burnt pancake graph. That is, in a \( n \)-burnt pancake graph with at most \( (n-2k+1) \) faulty clusters whose diameters are at most 3, for \( k \) pairs of the source and destination nodes, we prove that our algorithm can construct \( k \) fault-free disjoint paths between them. We also prove that the time complexity of the algorithm is \( O(kn^3) \) and the maximum path length is \( 2n+13 \).

2. Preliminaries

In this section, we first introduce a definition of a burnt pancake graph and related definitions.

Definition 1: A permutation \( u = (u_1, u_2, \ldots, u_n) \) that satisfies that \( \{ |u_1|, |u_2|, \ldots, |u_n| \} = \langle n \rangle \) is called a signed permutation where \( \langle n \rangle = \{1, 2, \ldots, n\} \).

Definition 2: For a signed permutation \( u = (u_1, u_2, \ldots, u_n) \) and an integer \( i \) \((1 \leq i \leq n)\), the signed prefix reversal operation \( u^{(i)} \) is defined by \( u^{(i)} = (-u_i, -u_{i-1}, \ldots, -u_2, -u_1, u_{i+1}, \ldots, u_n) \).
We use the notation \( u^{(i_1, \ldots, i_k)} \) as a short hand of a signed prefix reversal operation \( u^{(i_1, \ldots, i_k)(k)} \). A signed prefix reversal operation is invertible and \( u^{(i,i)} = u \) holds.

**Definition 3:** If a graph \( G(V, E) \) satisfies the conditions that \( V = \{(u_1, u_2, \ldots, u_n) | (u_1, u_2, \ldots, u_n) \text{ is a signed permutation of } (n) \} \) and \( E = \{(u, u^{(i)}) | u \in V, 1 \leq i \leq n \} \), \( G \) is called an \( n \)-burnt pancake graph.

In this paper, we denote \( B_n \) and \( i \) to represent an \( n \)-burnt pancake graph and \(-i\), respectively. Figure 1 shows examples of burnt pancake graphs, \( B_1 \), \( B_2 \), and \( B_3 \).

![Fig. 1: Examples of burnt pancake graphs.](image)

A \( B_n \) is a symmetric graph, and the number of nodes, the number of edges, the degree, and the connectivity are \( n! \times 2^n, n! \times n \times 2^{n-1}, n \), and \( n \), respectively. There is no shortest-path routing algorithm found for a \( B_n \) in time complexity of the polynomial order of \( n \). However, the fact that \( d(B_n) \leq 2n \) is proved.

**Definition 4:** In a \( B_n \), for an arbitrary node \( u = (u_1, u_2, \ldots, u_n) \) and an arbitrary integer \( k \ (1 \leq |k| \leq n) \), an extended signed prefix reversal operation \( u^{(k)} \) is defined by

\[
 u^{(k)} = \begin{cases} u^{(i)} & (u_i = k) \\ u^{(i,1)} & (u_i = k) 
\end{cases}
\]

**Definition 5:** In a \( B_n \), the sub graph induced by the subset of nodes that have \( k \) at the rightmost positions in their permutations is isomorphic to a \( B_{n-1} \). The sub graph is specified by \( B_{n-1}(k) \) by using the \( k \) as its index. A \( B_n \) is decomposable into \( 2n \) \( B_{n-1} \)'s that are mutually disjoint. Each sub graph is called a sub burnt pancake graph. In addition, the sub burnt pancake graph that contains the node \( u(\in B_n) \) is denoted by \( B_{n-1}(u) \).

In Figure 1, the sub graph indicated by the dashed circle is specified by \( B_2(1) \), which is isomorphic to a \( B_2 \).

**Definition 6:** A connected sub graph in a graph is called a cluster. If all of the nodes in a cluster are faulty, the cluster is called a faulty cluster. In addition, in a graph \( G(V, E) \) the nodes defined by \( \min_{v \in V} \max_{v \in V} d(c, v) \) are called the centers of the graph \( G \).

**Definition 7:** In a \( B_n \) with faulty clusters, if a \( B_{n-1}(k) \) does not include any center of the faulty clusters, it is called a candidate sub burnt pancake graph and denoted by \( CB_{n-1}(k) \).

**Definition 8:** The set that consists of the nodes that have \( j \) and \( i \) at the leftmost and rightmost positions in their permutations, respectively, is called a portset from \( B_{n-1}(i) \) to \( B_{n-1}(j) \), and denoted by \( P(i, j) \).

**Theorem 1:** In a \( B_n \), for two non-faulty nodes \( s = (s_1, s_2, \ldots, s_n) \), \( t = (t_1, t_2, \ldots, t_n) \) and a set of faulty nodes \( F (|F| \leq n-1) \), we can construct a fault-free path between \( s \) and \( t \) of length at most \( 2n + 4 \) in time complexity \( O(n^2) \).

**Proof:** See [19].

\( \square \)

### 3. Algorithm

In this section, we show an algorithm that solves the pairwise cluster-fault-tolerant disjoint paths problem in a burnt pancake graph.

#### 3.1. Lemmas

**Lemma 1:** For two distinct nodes \( u \) and \( v \) in a port set \( P(l, m) \ (1 \leq |l|, |m| \leq n, |l| \neq |m|) \), the distance between them \( d(u, v) \) is no less than 3.

**(Proof)** Let \( u = (u_1, u_2, \ldots, u_n) \) and \( v = (v_1, v_2, \ldots, v_n) \) be two distinct nodes in a port set \( P(l, m) \). From assumption, \( u_1 = v_1 = m \) and \( u_n = v_n = l \) hold. If \( d(u, v) = 0 \), \( u = v \) holds. It is contradictory to the fact that \( u \) and \( v \) are distinct. If \( d(u, v) = 1 \), \( u \) and \( v \) are adjacent. For any two adjacent nodes in \( B_n \), the elements in their leftmost positions are different. However, it is contradictory to the
fact that \( u, v \in P(l, m) \). If \( d(u, v) = 2, \exists i, j \ (i \neq j) \) such that \( u \to u^{(i)} \to u^{(j)} = v \in P(l, m) \). However, if \( i \neq j \), \( u_1 \neq v_1 \) holds. It means that \( u \) and \( v \) do not belong to a single port set \( P(l, m) \). Consequently, Lemma 1 holds. \( \square \)

**Lemma 2:** There is not a cycle in a \( B_n \) whose length is less than 8.

(Proof) We prove this lemma based on mathematical induction. In \( B_1 \), there is no cycle and \( B_2 \) is a cycle of length 8. Hence, this lemma holds for \( n \leq 2 \). Now, we prove that Lemma 2 holds for \( B_n \) with the hypothesis that the lemma holds for \( B_{n-1} \) \((n \geq 3)\). If there is a cycle \( C \) whose length is less than 8, \( C \) has an edge between two sub burnt pancake graphs since this lemma holds for any sub burnt pancake graphs by hypothesis. To be a cycle, \( C \) must have at least two such edges. If \( C \) has exactly two edges \((a_0, b_0)\) and \((a_1, b_1)\) between sub burnt pancake graphs, we can assume that \( a_0, a_1 \in P(l, m) \) and \( b_0, b_1 \in P(m, l) \) without loss of generality. Then, from Lemma 1, \( d(a_0, a_1) \geq 3 \) and \( d(b_0, b_1) \geq 3 \) hold. Therefore, the length of \( C \) is at least 8. It is a contradiction, and it means that \( C \) does not exist. If \( C \) has exactly three edges between sub burnt pancake graphs \((a_0, b_0), (a_1, b_1), \) and \((a_2, b_2)\), we can assume that \( a_0 \in P(j, l), b_0 \in P(j, l), a_1 \in P(l, m), b_1 \in P(m, l), a_2 \in P(m, j), \) and \( b_2 \in P(m, j) \) without loss of generality. Since the length of \( C \) is less than 8, at least two of the distances \( d(a_0, b_2), d(a_1, b_0), \) and \( d(a_2, b_1) \) must be 1. Here, without loss of generality, we can assume that \( d(a_0, b_2) = 1 \) and \( d(b_0, a_1) = 1 \) hold. Then, since \( b_2 \in P(j, l), b_2 = (m, \ldots, j) \) holds. In addition, because \( a_0 \in P(j, l) \) and \( d(a_0, b_2) = 1 \) hold, \( a_0 = (l, \ldots, m, \ldots, j) \) holds. Hence, \( b_0 = a_0^{(n)} = (j, \ldots, m, \ldots, l) \) holds. On the other hand, \( a_1 = (m, \ldots, l) \) holds since \( a_1 \in P(l, m) \). Then, from the sign of \( m, b_0 \) and \( a_1 \) cannot be adjacent. Therefore, \( d(b_0, a_1) \neq 1 \) holds. It is a contradiction, and \( C \) does not exist. Finally, if \( C \) has 4 or more edges between sub burnt pancake graphs, still 4 or more edges are necessary to make \( C \) a cycle. It means that the length of \( C \) is at least 8. From the discussion above, we have proved this lemma. \( \square \)

**Lemma 3:** In a \( B_n \), if there are at most \((n - 2k + 1)\) faulty clusters whose diameters are at most 3, there are at least \((4k - 2)\) candidate sub burnt pancake graphs.

(Proof) There are \( 2n \) sub burnt pancake graphs. Because a cluster whose diameter is at most 3 has at most two centers, there are at least \((2n - 2(n - 2k + 1) = 4k - 2)\) candidate sub burnt pancake graphs. \( \square \)

**Lemma 4:** In a \( B_n \), for a node \( u = (u_1, u_2, \ldots, u_n) \), we can construct \( n \) disjoint paths of length at most 3 from \( u \) to \( n \) distinct sub burnt pancake graphs \( B_{n-1}(k) \) \((k \neq |u_n|)\).

(Proof) We can construct the paths of lengths at most 3 from \( u \) to \((2n - 2)\) sub burnt pancake graphs as follows:

\[
\begin{align*}
\{ & u \to u^{(n)} \in B_{n-1}(u_1) \\
& u \to u^{(i)} \to u^{(i, n)} \in B_{n-1}(u_i) \ (1 \leq i \leq n - 1) \\
& u \to u^{(i)} \to u^{(i, 1)} \to u^{(i, n)} \in B_{n-1}(u_i) \\
& (2 \leq i \leq n - 1)
\end{align*}
\]

Among these \((2n - 2)\) paths we can select \( n \) disjoint paths \( u \to u^{(n)} \), \( u \to u^{(1, n)} \), \( u \to u^{(2, n)}, \ldots, u \to u^{(n-1, n)} \), for instance.

(Proof) Let \( P \): \( u \leadsto v \) and \( Q \): \( u \leadsto w \) are two distinct paths among the \( n \) paths given in Lemma 4. Then, from Lemma 2, a cluster cannot overlap simultaneously the nodes on \( P \) and \( Q \) inside \( B_{n-1}(u_n) \). Also, from Lemma 1, it is impossible for a cluster to overlap a node on \( P \) inside \( B_{n-1}(u_n) \) and the node \( w \) on \( Q \) simultaneously. Finally, let us construct a path of length 2 \( R: v \to x \to x^{(n)} \) so that \( x^{(n)} \) is in the same sub burnt pancake graph as \( w \). Then, because \( v = (u_n, \ldots) \), \( x \) contains \( u_n \) and \( x^{(n)} \) contains \( u_n \). Since \( u = (u_n, \ldots) \), \( w \) and \( x^{(n)} \) cannot be adjacent. Hence, \( d(v, w) > 3 \). From the discussion above, this lemma holds.
Lemma 6: In a Bn, for a node u = (u1, u2, . . . , un) and a sub burnt pancake graph Bn−1(k), (k ≠ u1, {ui}], we can construct n disjoint paths of length at most 5 from u to the Bn−1(k) in O(n^2) time complexity. (Proof) If |k| ≠ |u1|, |ui|, we can construct n paths of lengths at most 3 as follows:

\[
\begin{align*}
& u \rightarrow u(i) \rightarrow u(i, [k]) \rightarrow u(i, [k], n) & (1 \leq i \leq n, |u_i| \neq |k|) \\
& u \rightarrow u(i) \rightarrow u(i, n) & (u_i = k) \\
& u \rightarrow u(i) \rightarrow u(i, 1) \rightarrow u(i, 1, n) & (u_i = k)
\end{align*}
\]

If k = u, we can construct n paths of lengths at most 4 as follows:

\[
\begin{align*}
& u \rightarrow u(i) \rightarrow u(i, n) \rightarrow u(i, n, 1) \rightarrow u(i, n, 1, n) & (1 \leq i \leq n - 1) \\
& u \rightarrow u(n) \rightarrow u(n, 1) \rightarrow u(n, 1, n) & (i = n)
\end{align*}
\]

If k = u1, we can construct n paths of lengths at most 5 as follows:

\[
\begin{align*}
& u \rightarrow u(1) \rightarrow u(1, n) & (i = 1) \\
& u \rightarrow u(i) \rightarrow u(i, 1) \rightarrow u(i, 1, i) \rightarrow u(i, 1, i) \rightarrow u(i, 1, i, i) & (2 \leq i \leq n)
\end{align*}
\]

From above discussion, we can construct n paths from u to Bn−1(k) of lengths at most 5 that are disjoint except for u. Also, it takes O(n) time to construct a path. Hence, it takes O(n^2) in total to construct n paths.

Lemma 7: For CBn−1(p) and distinct (n − 1) CBn−1(l)’s (1 ≤ |l_i| ≤ n, 1 ≤ i ≤ n, |p| ≠ ∀|l_i|, and |l_i| ≠ |l_j| for i ≠ j) such that each CBn−1 has at most (n − 2) faulty nodes, we can construct (n − 1) disjoint fault-free paths of lengths 6 each of which is from an arbitrary node u_i in each CBn−1(l_i) to CBn−1(l_i) via CBn−1(p) in the time complexity O(n^3).

(Proof) Because CBn−1(l_i), CBn−1(l_j), and CBn−1(p) are sub burnt pancake graphs, P(l_i), P(p), P(l_j), and P(l_i, j) do not include any faulty node. Therefore, if u_i = p, we can construct a fault-free path u_i,1 → u_i(n) ∈ CBn−1(p) → u_i(n, 1) → u_i(n, 1, n) ∈ CBn−1(l_i) of length 3. Moreover, if u_i = p, we can construct a fault-free path u_i,1 → u_i(1) → u_i(1, n) ∈ CBn−1(p) → u_i(1, 1) → u_i(1, 1, n) ∈ CBn−1(l_i) of length 4.

For each u_i with u_i,j = p and j ≥ 2, we can construct (n − 1) disjoint paths R_i,1 ≤ h ≤ n − 1 from u_i to CBn−1(l_i) via CBn−1(p) as follows:

\[
\begin{align*}
& u_i \rightarrow u_i^{(h)} \rightarrow u_i^{(h, j)} \rightarrow u_i^{(h, j, 1)} & (1 \leq h \leq n - 1) \\
& u_i \rightarrow u_i^{(h, j, 1)} \rightarrow u_i^{(h, j, 1, n)} & (1 \leq h \leq j - 1) \\
& u_i \rightarrow u_i^{(h, j, 1, n)} & (1 \leq h \leq j - 1) \\
& u_i \rightarrow u_i^{(h, j, 1, 1, n)} & (1 \leq h \leq j - 1)
\end{align*}
\]

Then, we can find a fault-free path among the above (n − 1) paths for each u_i. Each path construction takes O(1) time. It takes O(n) time to check whether a path is fault-free or not. Hence, it takes O(n^2) time to find a fault-free path for one u_i. Therefore, construction of (n − 1) paths takes O(n^3) time in total.

Figure 3 shows the (n − 1) fault-free disjoint paths of length at most 6 constructed in Lemma 7.

3.2 Algorithm Description

In this section, we describe the details of the algorithm for cluster-fault-tolerant k-pairwise disjoint path routing, and estimate the maximum path length and its time complexity. In a Bn, for k pairs of source and destination nodes (3 ≤ k ≤ n/2), the algorithm first constructs paths s_i → s_î and t_i → t_î where s_î and t_î belong to a same Bn−1 and the paths do not include any node on the other paths s_j → s_ĵ nor t_j → t_ĵ where j ≠ i. Then, it connects s_î and t_î by a fault-free path in the Bn−1 by using the fault-tolerant routing algorithm. The Bn−1 is called the target sub burnt pancake graph for the pair of nodes s_i and t_i, and denoted by Bn−1(l_i) (1 ≤ |l_i| ≤ n, 1 ≤ i ≤ k). Also, in the rest of this paper, we assume that the candidate sub burnt pancake graph for s_i satisfies the condition that it does not include any nodes on the other paths s_j → s_ĵ nor t_j → t_ĵ in
addition to the condition that it does not include any center of faulty clusters.

The algorithm consists of the following four steps.

**Step 1** If there is a \( B_{n-1}(m) \) that contains \( s_i \) or \( t_i \) and the \( B_{n-1}(m) \) is a candidate sub burnt pancake graph for \( s_i \) or \( t_i \), assign the \( B_{n-1}(m) \) to the target sub burnt pancake graph \( B_{n-1}(l_i) \). If \( s_i \) and \( t_i \) are included in distinct \( B_{n-1}(p) \) and \( B_{n-1}(q) \), respectively, and both of \( B_{n-1}(p) \) and \( B_{n-1}(q) \) satisfy the conditions of candidate sub burnt pancakes for \( s_i \) and \( t_i \), either of them are assigned to \( B_{n-1}(l_i) \). We can assign a target sub burnt pancake graph for each pair of source and destination nodes in \( O(n) \) time.

**Step 2** For each pair of the source node \( s_i = (s_{i1}, s_{i2}, \ldots, s_{in}) \) and the destination nodes \( t_i \) to which any target sub burnt pancake graph is not assigned, construct a path from either of the source or destination nodes to a candidate sub burnt pancake graph for it. Here, we assume that we found a candidate sub burnt pancake graph for \( s_i \). Then, we can assign a target sub burnt pancake graph and construct a path by the following three sub steps.

**Sub Step 2a** If the \( B_{n-1}(s_{i1}) \) is a candidate sub burnt pancake graph for \( s_i \), we assign \( B_{n-1}(s_{i1}) \) to the target sub burnt pancake graph \( B_{n-1}(l_i) \), construct a path \( s_i \rightarrow s_{i1}^{(n)} \) of length 1 to the sub burnt pancake graph, and let \( s_i^{(n)} = s_{i1}' \).

**Sub Step 2b** If the \( B_{n-1}(s_{ip}) \) (\( 1 < p < n \)) is a candidate sub burnt pancake graph for \( s_i \), we try to construct a path \( s_i \rightarrow s_{ip}^{(p)} \rightarrow s_{ip}^{(p,n)} \) of length 2. If this path is fault-free and disjoint from other paths, we can assign \( B_{n-1}(s_{ip}) \) to the target sub burnt pancake graph \( B_{n-1}(l_i) \), and let \( s_i^{(p,n)} = s_{ip}' \).

**Sub Step 2c** If the \( B_{n-1}(s_{ip}) \) (\( 1 < p < n \)) is a candidate sub burnt pancake graph for \( s_i \), we try to construct a path \( s_i \rightarrow s_{ip}^{(p)} \rightarrow s_{ip}^{(p,n)} \) of length 3. If this path is fault-free and disjoint from other paths, we can assign \( B_{n-1}(s_{ip}) \) to the target sub burnt pancake graph \( B_{n-1}(l_i) \), and let \( s_i^{(p,n)} = s_{ip}' \).

In Step 2, we can assign a target sub burnt pancake graph for each pair of source and destination nodes by constructing a path of length at most 3 in \( O(n^3) \) time.

**Step 3** By Steps 1 and 2, for \( k \) pairs of nodes \( s_i \) and \( t_i \), target sub burnt pancake graphs \( B_{n-1}(l_i) \) are assigned and at least one path from either of the nodes is constructed. Here, we construct a path to \( B_{n-1}(l_i) \) from either of \( s_i \) or \( t_i \) from which a path to \( B_{n-1}(l_i) \) has not been constructed. For simplicity, we assume that a path from \( s_i \) to \( B_{n-1}(l_i) \) has been already constructed, and a path from \( t_i \) has not been constructed without loss of generality. Here, if \( t_{i1}, t_{i2}, \ldots, t_{in} \neq l_i \), consider the \( (n-1) \) paths of lengths at most 4 given in Lemma 6 excluding one path that includes \( t_i^{(n)} \). If there is a path among them that is fault-free and disjoint from other constructed paths, let the path be \( t_i^{(n)} \). If \( t_{i1} = l_i \), check whether the path of length 2, \( t_i \rightarrow t_i^{(1)} \rightarrow t_i^{(1,n)} \) is fault-free and disjoint from other constructed paths. If it is fault-free and disjoint from other constructed paths, let the path be \( t_i \rightarrow t_i^{(1)} \). If there is not such path, or if \( t_{im} = l_i \), we construct a path to a candidate sub burnt pancake graph for \( t_i \) as similar to the Sub Steps 2a, 2b, and 2c. Let this candidate sub burnt pancake graph be \( B_{n-1}(l_i^{(n)}) \). Then, this step is divided into two cases depending on \( l_i \) and \( l_i' \) to construct the path.

**Case 1** If \( l_i = l_i' \), For pairs of the nodes such that \( l_i = l_i' \) hold, we can construct disjoint paths of lengths at most 7 that pass a candidate sub burnt pancake graph \( B_{n-1}(p) \) that does not include any source nor destination node from Lemma 7. Note that if \( l_i = l_i' \), in case that there is a path \( s_i \rightarrow s_i' (\subset B_{n-1}(l_i')) \) of length 3 is constructed among the paths given in Lemma 4, it is possible to construct the path \( s_i \rightarrow s_i' (\subset B_{n-1}(l_i')) \) of length 2. Similar discussion holds for \( t_i \). From Step 2, the destination sub burnt pancake graph \( B_{n-1}(l_i) \) is selected among the candidate sub burnt pancake graphs so that it can be reached from the node \( s_i \) or \( t_i \) with the shortest path. Therefore, if \( l_i = l_i' \), the lengths of the paths \( s_i \rightarrow s_i' \) and \( t_i \rightarrow t_i' \) are
both 2. Therefore, the sum of the paths from $s_i$ and $t_i$ to $B_{n-1}(l_i)$ is at most $2 + 2 + 7 = 11$.

**Case 2)** ($l_i \neq l'_i$) If $l_i \neq l'_i$, consider the paths from $B_{n-1}(l'_i)$ to $B_{n-1}(l_i)$ of lengths at most 5 given by Lemma 6. Then, there is at least one fault-free path among them. Hence, if $l_i \neq l'_i$, the sum of the paths from $s_i$ and $t_i$ to $B_{n-1}(l_i)$ is at most $3 + 3 + 5 = 11$.

In this step, we can construct a path of length at most 11 between a pair of a source node and a destination node in $O(n^3)$ time.

**Step 4)** For $k$ pairs of nodes $s_i$ and $t_i$ (1 ≤ $i$ ≤ $k$), from Steps 1 to 3, we have constructed paths $s_i \rightsquigarrow s'_i (\in B_{n-1}(l_i))$ and $t_i \rightsquigarrow t'_i (\in B_{n-1}(l_i))$ where $B_{n-1}(l_i)$ is the target sub burnt pancake graph for $s_i$ and $t_i$. $B_{n-1}(l_i)$ does not include any node on $s_j \rightsquigarrow s'_j$ or $t_j \rightsquigarrow t'_j$ ($j \neq i$), and contains at most $(n - 2k + 1)$ faulty nodes. Therefore, from Theorem 1, we can construct a path $s'_i \rightsquigarrow t'_i$ of length at most $2n + 2$ in $O(n^2)$ time.

Consequently, our algorithm can construct each path $s_i \rightsquigarrow t_i$ of length at most $2n + 13$ in $O(n^3)$ time. Therefore, it takes $O(kn^3)$ time to construct $k$ paths.

4. Evaluation

To evaluate performance of our algorithm, we conducted a computer experiment. The algorithm constructed $k$ disjoint fault-free paths between the $k$ pairs of source and destination nodes in a $n$-burnt pancake graph with $(n - 2k + 1)$ faulty clusters whose diameters are 3. In this section, we give the method, the results, and consideration.

4.1 Method

In the experiment, we applied our algorithm to solve the $k$-pairwise cluster-fault-free disjoint paths problem (3 ≤ $k$ ≤ $\lceil n/2 \rceil$) in a $B_n$ (5 ≤ $n$ ≤ 40). We repeated the following steps for 10,000 times and measured the average execution time and the maximum path length as well as the average path length.

1) We first set up $(n - 2k + 1)$ disjoint faulty clusters whose diameter is fixed to 3.
2) Then we select $k$ source nodes $s_1, s_2, \ldots, s_k$ and $k$ destination nodes $t_1, t_2, \ldots, t_k$ among non-faulty nodes.
3) We apply the algorithm to construct $k$ disjoint fault-free paths $s_i \rightsquigarrow t_i$ (1 ≤ $i$ ≤ $k$) and measure the execution time, the maximum path length, and the average path length.

4.2 Results

Figure 4 shows the results of the maximum path lengths and the average path lengths for 5 ≤ $n$ ≤ 40. In addition, Figure 5 shows the result of the average execution time for 5 ≤ $n$ ≤ 40 and 3 ≤ $k$ ≤ $\lceil n/2 \rceil$. From Figure 4, we can see that there is no path whose length attained the theoretical maximum path lengths. From Figure 5, the average execution time seems to converge to $O(n^{2.2})$.

![Fig. 4: Maximum and average path lengths of our algorithm](image)

![Fig. 5: Execution time of our algorithm](image)

4.3 Limitation

The algorithm that we have proposed cannot solve the $k$-pairwise cluster-fault-tolerant disjoint paths problem in a burnt pancake graph with $k = 2$. Our algorithm makes use of a redundant candidate sub burnt pancake graph. From Lemma 3, if there are at most $(n - 2k + 1)$ faulty clusters whose diameters are at most 3 in $B_n$, there are at least $(4k - 2)$ candidate sub burnt pancake graphs, which do not include any center of the faulty clusters. If there are at least two source or destination nodes in a candidate sub burnt pancake graph and they are not the corresponding pair, the candidate sub burnt pancake graph is unavailable. Therefore, in the worst case, $2k$ candidate sub burnt pancake graphs are not available. Moreover, it is necessary to assign one distinct candidate sub burnt pancake graph to each of the source and destination pair. Therefore, at least $k$ candidate
sub burnt pancake graphs must be required. In addition, as shown in Lemma 7, one candidate sub burnt pancake graph is used to connect paths between two $B_{n-1}$'s that do not have direct edges between them. Therefore, in total, $(3k+1)$ candidate sub burnt pancake graphs are necessary. Then, from $4k - 2 \geq 3k + 1$, $k \geq 3$ is a necessary condition to apply our algorithm.

5. Conclusions and Future Works

In this paper, we have proposed an algorithm that solves the $k$-pairwise disjoint paths problem in an $n$-burnt pancake graph with $(n - 2k + 1)$ faulty clusters whose diameters are at most 3. The time complexity of the algorithm is $O(kn^3)$ and the maximum path length is $2n+13$. We have conducted a computer experiment and its results showed that there was not any path that attained the theoretical maximum path length and the average time complexity of the algorithm is $O(n^{2.2})$.

Future works include extension of the algorithm so that it can address the cluster-fault-tolerant disjoint paths problem with two pairs of nodes in $B_n$ as well as improvement of the maximum path lengths.

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. 25330079.

References


