Swapping Colored Tokens on Graphs[☆]

Katsuhisa Yamanaka[a], Takashi Horiyama[b], J. Mark Keil[c], David Kirkpatrick[d], Yota Otachi[e], Toshiki Saitoh[g], Ryuhei Uehara[f], Yushi Uno[h]

[a] *Iwate University, Japan.*
[b] *Saitama University, Japan.*
[c] *University of Saskatchewan, Canada*
[d] *University of British Columbia, Canada.*
[e] *Kumamoto University, Japan.*
[f] *Japan Advanced Institute of Science and Technology, Japan.*
[g] *Kyushu Institute of Technology, Japan.*
[h] *Osaka Prefecture University, Japan.*

Abstract

We investigate the computational complexity of the following problem. We are given a graph in which each vertex has an initial and a target color. Each pair of adjacent vertices can swap their current colors. Our goal is to perform the minimum number of swaps so that the current and target colors agree at each vertex. When the colors are chosen from $\{1, 2, \ldots, c\}$, we call this problem c-Colored Token Swapping since the current color of a vertex can be seen as a colored token placed on the vertex. We show that c-Colored Token Swapping is NP-complete for $c = 3$ even if input graphs are restricted to connected planar bipartite graphs of maximum degree 3. We then show that 2-Colored Token Swapping can be solved in polynomial time for general graphs and in linear time for trees. Besides, we show that, the problem for complete graphs is fixed-parameter tractable when parameterized by the number of colors, while it is known to be NP-complete

[☆]A preliminary version appeared in the proceedings of the 14th International Symposium on Algorithms and Data Structures (WADS 2015), vol. 9214 of Lecture Notes in Computer Science, pp. 619–628, 2015.

Email addresses: yamanaka@cis.iwate-u.ac.jp (Katsuhisa Yamanaka), horiyama@al.ics.saitama-u.ac.jp (Takashi Horiyama), keil@cs.usask.ca (J. Mark Keil), kirk@cs.ubc.ca (David Kirkpatrick), otachi@cs.kumamoto-u.ac.jp (Yota Otachi), toshikis@ces.kyutech.ac.jp (Toshiki Saitoh), uehara@jaist.ac.jp (Ryuhei Uehara), uno@mi.s.osakafu-u.ac.jp (Yushi Uno)

when the number of colors is unbounded.

Keywords: computational complexity, NP-completeness, fixed-parameter algorithm, token swapping, colored token swapping

---

1. Introduction

Sorting problems are fundamental and important in computer science. Let us consider the problem of sorting a given permutation by applying the minimum number of swaps of two elements. If we are allowed to swap only adjacent elements (that is, we can apply only adjacent transpositions), then the minimum number of swaps is equal to the number of inversions of a permutation [12, 15]. If we are allowed to swap any two elements, then the minimum number of swaps is equal to the number of elements of a permutation minus the number of cycles of the permutation [2, 12]. Now, if we are given a set of "allowed swaps", can we exactly estimate the minimum number of swaps? We formalize this question as a problem of swapping tokens on graphs.

Let $G = (V, E)$ be an undirected unweighted graph with vertex set V and edge set E. Suppose that each vertex in G has a token, and each token has a color in $C = \{1, 2, \ldots, c\}$. Given two token-placements, we wish to transform one to the other by applying the fewest number of token swaps on adjacent vertices. We call the problem c-Colored Token Swapping if c is a constant, otherwise, we simply call it Colored Token Swapping(a formal definition can be found in the next section). See Figure 1 for an example.

In this paper, we study the computational complexity of c-Colored Token Swapping. We consider the case where c is a fixed constant and show the following results. If $c = 2$, then the problem can be solved in polynomial time (Theorem 6). On the other hand, 3-Colored Token Swapping is NP-complete even for planar bipartite graphs of maximum degree 3 (Theorem 2). We also show that c-Colored Token Swapping is $O(n^{c+2})$-time solvable for graphs of maximum degree at most 2 (Theorem 4), 2-Colored Token Swapping is linear-time solvable for trees (Theorem 8), and c-Colored Token Swapping is fixed-parameter tractable for complete graphs if c is the parameter (Theorem 15).

If the tokens have distinct colors, then the problem is called Token Swapping [20]. This variant has been investigated for several graph classes.

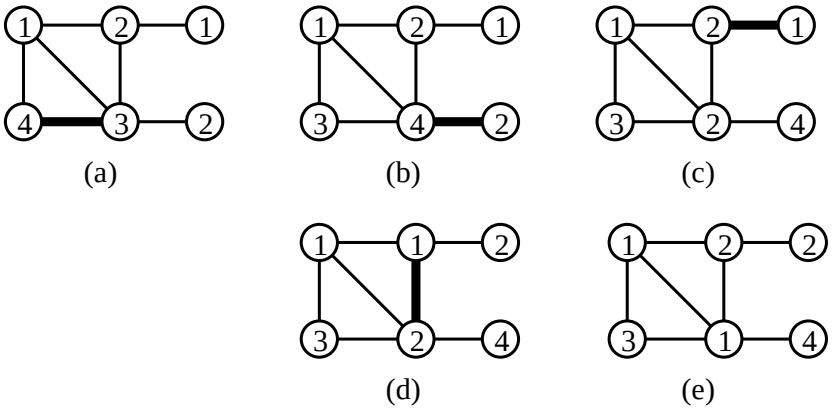

Figure 1: An instance of 4-Colored Token Swapping. Tokens on vertices are written inside circles. We swap the two tokens along each thick edge. (a) The initial token-placement. (b)–(d) Intermediate token-placements. (e) The target token-placement.

Token Swapping can be solved in polynomial time for paths [12, 15], cycles [12], stars [18], complete graphs [2, 12], and complete bipartite graphs [20]. Heath and Vergara [10] gave a polynomial-time 2-approximation algorithm for squares of paths (see also [6, 7]). Yamanaka et al. [20] gave a polynomial-time 2-approximation algorithm for trees.

Recently, Miltzow et al. [17] gave computational complexity results for Token Swapping. First, they showed the NP-completeness of Token Swapping. Second, they proposed an exact exponential algorithm for Token Swapping on general graphs and showed that Token Swapping cannot be solved in $2^{o(n)}$ time unless the Exponential Time Hypothesis (ETH) fails, where n is the number of vertices. Third, they proposed polynomial-time 4-approximation algorithms for Token Swapping on general graphs and showed the APX-hardness of Token Swapping. Independently, Kawahara et al. [14] proved the NP-completeness of Token Swapping even if an input graph is bipartite and has only vertices of degree at most 3. More recently, Bonnet et al. [1] gave some results for Token Swapping. They showed the parameterized hardness: Token Swapping is W[1]-hard, parameterized by the number of swaps and Token Swapping cannot be solved in $f(k)(n+m)^{o(k/\log k)}$ unless the ETH fails, where n is the number of vertices, m is the number of edges and k is the number of swaps. Token Swapping on trees is one of the attractive open problems. They gave an interesting result for the open problem: Token Swapping is NP-hard even when both

the treewidth and the diameter are constant, and cannot be solved in $2^{o(n)}$ time unless the ETH fails.

Miltzow et al. [17] and Bonnet et al. [1] also gave important results on Colored Token Swapping. Here, we mention only the results related to our contributions. Miltzow et al. [17] gave the NP-completeness of Colored Token Swapping using $\Omega(n)$ colors. On the other hand, in this paper, we show the the NP-completeness of c-Colored Token Swapping even if the number of colors is only 3. Bonnet et al. [1] showed the NP-completeness of Colored Token Swapping on complete graphs using $\Omega(n)$ colors. To complement their result, we show the fixed-parameter tractability of c-Colored Token Swapping on complete graphs, parameterized by the number of colors.

2. Preliminaries

The graphs considered in this paper are finite, simple, and undirected. Let $G = (V, E)$ be an undirected unweighted graph with vertex set V and edge set E. We sometimes denote by $V(G)$ and $E(G)$ the vertex set and the edge set of G, respectively. We always denote $|V|$ by n. For a vertex v in G, let $N(v)$ be the set of all neighbors of v.

We formalize our problem as a problem reconfiguring an initial coloring of vertices to the target one by repeatedly swapping the two colors on adjacent vertices as follows. Let $C = \{1, 2, \ldots, c\}$ be a set of colors. In this paper, we assume that c is a constant unless otherwise noted. A *token-placement* of G is a surjective function $f\colon V \to C$. For a vertex v, $f(v)$ represents the color of the token placed on v. Note that we assume that each color in C appears at least once. Two distinct token-placements f and f' of G are *adjacent* if the following two conditions (a) and (b) hold:

(a) there exists $(u, v) \in E$ such that $f'(u) = f(v)$ and $f'(v) = f(u)$;
(b) $f'(w) = f(w)$ for all vertices $w \in V \setminus \{u, v\}$.

In other words, the token-placement f' is obtained from f by *swapping* the tokens on the two adjacent vertices u and v. For two token-placements f and f' of G, a sequence $\mathcal{S} = \langle f_1, f_2, \ldots, f_h \rangle$ of token-placements is a *swapping sequence* between f and f' if the following three conditions (1)–(3) hold:

(1) $f_1 = f$ and $f_h = f'$;
(2) f_k is a token-placement of G for each $k = 1, 2, \ldots, h$;
(3) f_{k-1} and f_k are adjacent for every $k = 2, 3, \ldots, h$.

The *length* of a swapping sequence $\mathcal{S}$, denoted by $\mathrm{len}(\mathcal{S})$, is defined to be the number of token-placements in $\mathcal{S}$ minus one, that is, $\mathrm{len}(\mathcal{S})$ indicates the number of swaps in $\mathcal{S}$. For two token-placements f and f' of G, we denote by $\mathrm{OPT}(f, f')$ the minimum length of a swapping sequence between f and f'. If there is no swapping sequence between f and f', then we set $\mathrm{OPT}(f, f') = \infty$.

Given two token-placements f_0 and f_t of a graph G and a nonnegative integer ℓ, the c-Colored Token Swapping problem is to determine whether $\mathrm{OPT}(f_0, f_t) \leq \ell$ holds. From now on, we always denote by f_0 and f_t the *initial* and *target* token-placements of G, respectively.

Let f and f' be two token-placements of G. If there exist a component C of G and a color i such that the numbers of tokens of color i in C are not equal in f and f', then we cannot transform f into f'. If there are no such C and i, then we write $f \simeq f'$. Note that one can easily check whether $f \simeq f'$ in linear time. Thus we can assume that input instances satisfy this condition. It holds that if $f \simeq f'$, then f can be transformed into f' with at most $\binom{n}{2}$ swaps. This can be shown by slightly modifying the proof of Theorem 1 in [20].

Lemma 1. *Let f_0 and f_t be token-placements of G. If $f_0 \simeq f_t$, then*

$$\mathit{OPT}(f_0, f_t) \leq \binom{n}{2}.$$

Proof. No two tokens are swapped twice or more in a swapping sequence with the minimum length. Hence, the claim holds. □

The bound in Lemma 1 is tight. For the path graph $(v_1, v_2, \ldots, v_n)$, we set $f_0(v_i) = i$ and $f_t(v_i) = n - i + 1$ for $i = 1, 2, \ldots, n$. It is known that this instance requires $\binom{n}{2}$ swaps [12, 15].

3. Hardness results

In this section, we show that 3-Colored Token Swapping is NP-complete by constructing a polynomial-time reduction from Planar 3DM [4]. To define Planar 3DM, we first introduce the following well-known NP-complete problem.

Problem: 3-Dimensional Matching (3DM) [9, SP1]
Instance: Set $T \subseteq X \times Y \times Z$, where X, Y, and Z are disjoint sets having

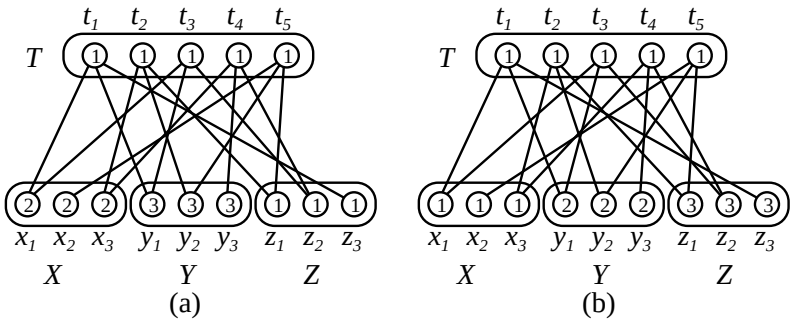

Figure 2: (a) The initial token-placement and (b) the target token-placement of the graph constructed from an instance ($X = \{x_1, x_2, x_3\},\ Y = \{y_1, y_2, y_3\},\ Z = \{z_1, z_2, z_3\},\ T = \{t_1 = (x_1, y_1, z_3),\ t_2 = (x_3, y_2, z_1),\ t_3 = (x_1, y_1, z_2),\ t_4 = (x_3, y_3, z_2),\ t_5 = (x_2, y_2, z_1)\}$).

the same number m of elements.

Question: Does T contain a matching, i.e., a subset $T' \subseteq T$ such that $|T'| = m$ and it contains all elements of X, Y, and Z?

Planar 3DM is a restricted version of 3DM in which the following bipartite graph G is planar. The graph G has the vertex set $V(G) = T \cup X \cup Y \cup Z$ with a bipartition $(T, X \cup Y \cup Z)$. Two vertices $t \in T$ and $w \in X \cup Y \cup Z$ are adjacent in G if and only if $w \in t$. Planar 3DM is NP-complete even if G is a connected graph of maximum degree 3 [4].

Theorem 2. 3-Colored Token Swapping *is NP-complete even for connected planar bipartite graphs of maximum degree 3.*

Proof. By Lemma 1, there is a polynomial-length swapping sequence between two token-placements, and thus 3-Colored Token Swapping is in NP.

Now we present a reduction from Planar 3DM, as illustrated in Figure 2. Let $(X, Y, Z; T)$ be an instance of Planar 3DM and $m = |X| = |Y| = |Z|$. As mentioned above, we construct a bipartite graph $G = (T, X \cup Y \cup Z; E)$ from $(X, Y, Z; T)$. We set $f_0(x) = 2$ and $f_t(x) = 1$ for every $x \in X$, set $f_0(y) = 3$ and $f_t(y) = 2$ for every $y \in Y$, set $f_0(z) = 1$ and $f_t(z) = 3$ for every $z \in Z$, and set $f_0(t) = 1$ and $f_t(t) = 1$ for every $t \in T$. From the assumptions, G is a planar bipartite graph of maximum degree 3. The reduction can be done in polynomial time. We prove that the instance $(X, Y, Z; T)$ is a yes-instance if and only if $\mathrm{OPT}(f_0, f_t) \leq 3m$.

To show the only-if part, assume that there exists a subset T' of T such that $|T'| = m$ and T' contains all elements of X, Y, and Z. Since the

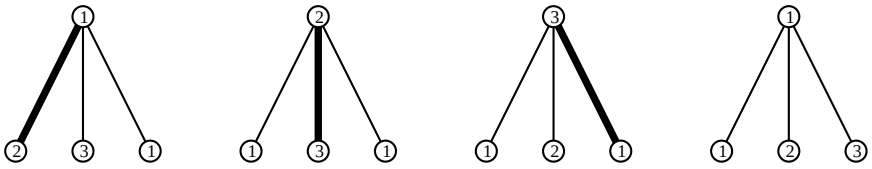

Figure 3: A swapping sequence to resolve the token-placement of a triple.

elements of T' are pairwise disjoint, we can cover the subgraph of G induced by $T' \cup X \cup Y \cup Z$ with m disjoint stars of four vertices, where each star is induced by an element t of T' and its three elements. To locally move the tokens to the target places in such a star, we need only three swaps. See Figure 3. This implies that a swapping sequence of length $3m$ exists.

To show the if part, assume that there is a swapping sequence $\mathcal{S}$ from f_0 to f_t with at most $3m$ swaps. Let $T' \subseteq T$ be the set of vertices such that the tokens on them are moved in $\mathcal{S}$. Let G' be the subgraph of G induced by $T' \cup X \cup Y \cup Z$. Let $w \in X \cup Y \cup Z$. Since $f_0(w) \neq f_t(w)$ and $N(w) \subseteq T$, the sequence $\mathcal{S}$ swaps the tokens on w and on a neighbor $t \in T'$ of w at least once. This implies that w has degree at least 1 in G'. Since each $t \in T'$ has degree at most 3 in G', we can conclude that $|T'| \geq \frac{1}{3}|X \cup Y \cup Z| = m$. In $\mathcal{S}$, the token placed on a vertex in $X \cup Y$ in the initial token-placement is moved at least twice, while the token placed on a vertex in $Z \cup T'$ is moved at least once. As a swap moves two tokens at the same time,

$$\mathrm{len}(\mathcal{S}) \geq \frac{1}{2}(2|X| + 2|Y| + |Z| + |T'|) \geq 3m.$$

From the assumption that $\mathrm{len}(\mathcal{S}) \leq 3m$, it follows that $|T'| = m$, and hence each $w \in X \cup Y \cup Z$ has degree exactly 1 in G'. Therefore, G' consists of m disjoint stars centered at the vertices of T' which form a solution of Planar 3DM. □

The proof above can be modified to show hardness for instances with more colors. We transform the reduced instance by adding a path of polynomial length containing additional colors as follows.

Corollary 3. *For every constant $c \geq 3$, c-Colored Token Swapping is NP-complete even for connected planar bipartite graphs of maximum degree 3.*

Proof. Let G be the graph obtained in the proof of Theorem 2. Recall that the token-placement of G uses three colors 1, 2, and 3. It is known that we can

assume that G has a degree-2 vertex [4]. We connect a path $(p_4, p_5, \ldots, p_c)$ to G by adding an edge between p_4 and a degree-2 vertex in G. We set $f_0(p_i) = f_t(p_i) = i$ for every $i \in \{4, \ldots, c\}$. The proof of Theorem 2 still works for the obtained graph. □

4. Positive results

In this section, we give some positive results. First, we show that c-Colored Token Swapping for graphs of maximum degree at most 2 is in XP[1] when c is the parameter. Second, we show that 2-Colored Token Swapping for general graphs can be solved in polynomial time. Third, we show that the 2-Colored Token Swapping problem for trees can be solved in linear time without constructing a swapping sequence. Finally, we show that the c-Colored Token Swapping problem for complete graphs is fixed-parameter tractable[2] when c is the parameter.

4.1. c-Colored Token Swapping on graphs of maximum degree 2

In this subsection, we show that the degree bound in Theorem 2 and Corollary 3 is tight. If a graph has maximum degree at most 2, then we can solve c-Colored Token Swapping in XP time for every constant c as follows. Each component of a graph of maximum degree at most 2 is a path or a cycle. Observe that a shortest swapping sequence does not swap tokens of the same color. This immediately gives a unique matching between tokens and target vertices for a path component. For a cycle component, observe that each color class has at most n candidates for such a matching restricted to the color class. This is because after we guess the target of a token in a color class, the targets of the other tokens in the color class can be uniquely determined. In total, there are at most n^c matchings between tokens and target vertices. By guessing such a matching, we can reduce c-Colored Token Swapping to Token Swapping. Now we can apply Jerrum's $O(n^2)$-time algorithms for solving Token Swapping on paths and cycles [12]. Thus we have the following theorem.

Theorem 4. *c-Colored Token Swapping can be solved in $O(n^{c+2})$ time for graphs of maximum degree at most 2.*

[1] See [3, p. 13] for a formal definition of XP.

[2] Also see [3, p. 13] for a formal definition of fixed-parameter tractability.

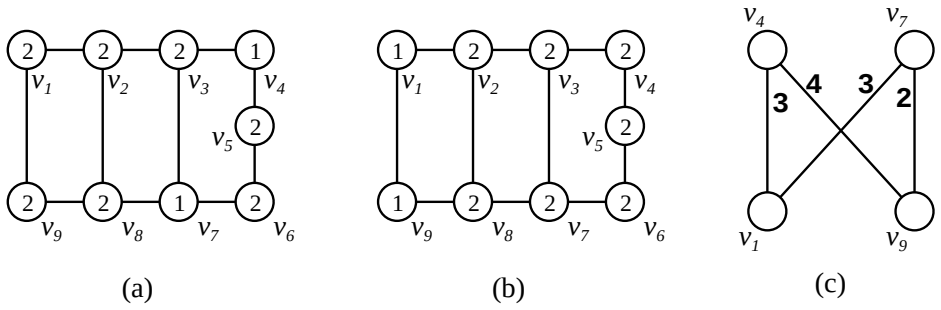

Figure 4: (a) The initial token-placement. (b) The target token-placement. (c) The weighted complete bipartite graph constructed from (a) and (b). The edge weights are written beside the edges.

4.2. 2-Colored Token Swapping *on general graphs*

In this section, we show that 2-Colored Token Swapping is polynomially solvable. We lately noticed that an equivalent problem was solved previously by van den Heuvel [19, Theorem 4.3]. Our proof is quite similar to the one in [19]. However, to be self-contained we describe our proof.

Let $C = \{1, 2\}$ be the color set. Let $G = (V, E)$ be a graph, and let f_0 and f_t be initial and target token-placements. We first construct the following weighted complete bipartite graph $G_B = (X, Y, E_B, w)$ as follows. The vertex sets X, Y and the edge set E_B are defined as follows:

$$\begin{aligned} X &= \{x_v \mid v \in V \text{ and } f_0(v) = 1\}, \\ Y &= \{y_v \mid v \in V \text{ and } f_t(v) = 1\}, \\ E_B &= \{(x, y) \mid x \in X \text{ and } y \in Y\}. \end{aligned}$$

The vertices in X correspond to the vertices in V having tokens of color 1 in f_0, and the vertices in Y correspond to the vertices in V having tokens of color 1 in f_t. For $x \in X$ and $y \in Y$, the weight $w(e)$ of the edge $e = (x, y)$ is defined as the length of a shortest path from x to y in G. Figure 4 gives an example of an initial token-placement, a target token-placement, and the associated weighted complete bipartite graph.

We bound $\mathrm{OPT}(f_0, f_t)$ from below as follows. Let $\mathcal{S}$ be a swapping sequence between f_0 and f_t. The swapping sequence gives a perfect matching of G_B, as follows. For each token of color 1, we choose an edge (x, y) of G_B if the token is placed on $x \in X$ in f_0 and on $y \in Y$ in f_t. The obtained set is a perfect matching of G_B. A token corresponding to an edge e in the

matching needs $w(e)$ swaps, and two tokens of color 1 are never swapped in $\mathcal{S}$. Therefore, for a minimum weight matching M of G_B, we have the following lower bound:

$$\mathrm{OPT}(f_0, f_t) \geq \sum_{e \in M} w(e).$$

Now we describe our algorithm. First we find a minimum weight perfect matching M of G_B. We then choose an edge e in M. Let $P_e = \langle p_1, p_2, \ldots, p_q \rangle$ be a shortest path in G corresponding to e. We have the following lemma.

Lemma 5. *Suppose that the two tokens on the endpoints of P_e have different colors. The two tokens can be swapped by $w(e)$ swaps such that the color of the token on each internal vertex does not change.*

Proof. Without loss of generality, we assume that $f_0(p_1) = 2$ and $f_0(p_q) = 1$ hold. We first choose the minimum i such that $f_0(p_i) = 1$ holds. We next move the token on p_i to p_1 by $i - 1$ swaps. Note that, after the swaps above, p_1 has a token of color 1 and p_j for each $j = 2, 3, \ldots, i$ has a token of color 2. We repeat the same process to the subpath $\langle p_i, p_{i+1}, \ldots, p_q \rangle$. Finally, we obtain the desired token-placement. Recall that there are only two colors on graphs, and so the above "color shift" operation works. See Figure 5 for an example. Since each edge of P_e is used by one swap, the total number of swaps is $w(e) = q - 1$. □

This lemma permits to move the two tokens on the two endpoints p_1 and p_q of P_e to their target positions in $w(e)$ swaps without changing the token-placement of the other vertices. Let g be the token-placement obtained after the swaps. We can observe that $f_0(v) = g(v)$ for every $v \in V \setminus \{p_1, p_q\}$ and $g(v) = f_t(v)$ for $v \in \{p_1, p_q\}$. Then we remove e from the matching M. We repeat the same process until M becomes empty. Our algorithm always exchanges tokens on two vertices using a shortest path between the vertices. Hence, the length of the swapping sequence constructed by our algorithm is equal to the lower bound.

Now we estimate the running time of our algorithm. The algorithm first constructs the weighted complete bipartite graph. This can be done using the Floyd-Warshall algorithm, which computes all-pairs shortest paths in a graph, in $\mathrm{O}(n^3)$ time. Then, our algorithm constructs a minimum weight perfect matching. This can be done in $\mathrm{O}(n^3)$ time [16, p.252]. Finally, for each edge in the matching, along the corresponding shortest path, our

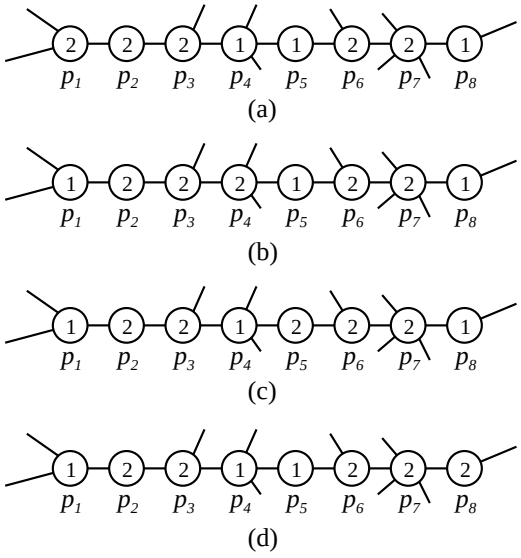

Figure 5: An example of the color shift operation on the path $\langle p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8 \rangle$. (a) The initial token-placement. (b) The token-placement obtained from (a) by moving the token on p_4 to p_1. (c) The token-placement obtained from (b) by moving the token on p_5 to p_4. (d) The token-placement obtained from (c) by moving the token on p_8 to p_5. The total number of swaps is 7.

algorithm moves the tokens on the endpoints of the path in linear time. We have the following theorem.

Theorem 6. 2-Colored Token Swapping *is solvable in* $O(n^3)$ *time. Furthermore, a swapping sequence of the minimum length can be constructed in the same running time.*

4.3. 2-Colored Token Swapping *on trees*

In the previous subsection, we proved that, for general graphs, 2-Colored Token Swapping can be solved in $O(n^3)$ time. In this subsection, we show that 2-Colored Token Swapping for trees can be solved in linear time without constructing a swapping sequence.

Let T be an input tree, and let f_0 and f_t be an initial token-placement and a target token-placement of T. Let $e = (x, y)$ be an edge of T. Removal of e disconnects T into two subtrees: $T(x)$ with x and $T(y)$ with y.

Now we define the value $\mathrm{diff}(e)$ for each edge e of T. Intuitively, $\mathrm{diff}(e)$ is the number of tokens of color 1 which we wish to move along e. More

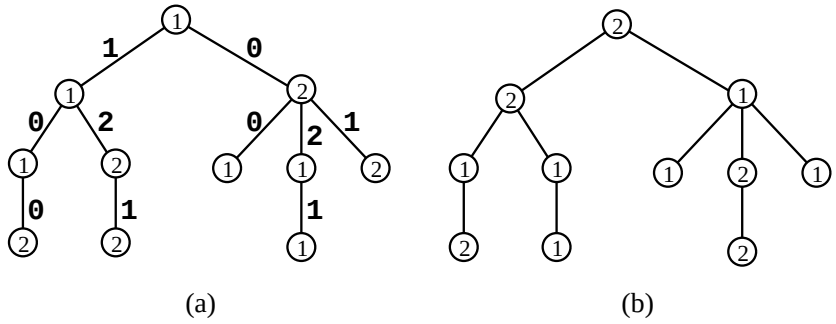

Figure 6: An example of diff(e). For each edge e, the value diff(e) is written beside e. (a) The initial token-placement. (b) The target token-placement.

formally, we give the definition of diff(e) as follows. Let $n^1(f_0)$ and $n^1(f_t)$ be the numbers of tokens of color 1 in f_0 and f_t in $T(x)$, respectively. Then, we define $\mathrm{diff}(e) = |n^1(f_0) - n^1(f_t)|$. (Note that, even if we count tokens of color 1 in f_0 and f_t in $T(y)$ instead of $T(x)$, the value diff(e) takes the same value as $f_0 \simeq f_t$.) See Figure 6 for an example. For each edge e of T, we need to move at least diff(e) tokens of color 1 from a subtree to the other along e. Therefore, $\mathrm{OPT}(f_0, f_t)$ is lower bounded by the sum $D = \sum_{e \in E(T)} \mathrm{diff}(e)$.

To give an upper bound of $\mathrm{OPT}(f_0, f_t)$, we next show that there exists a swapping sequence of length D. The following lemma is the key to construct the swapping sequence.

Lemma 7. *If $D \neq 0$, then there exists an edge e such that the swap on e decreases D by one.*

Proof. We first give an orientation of edges of T. For each edge $e = (x, y)$, we orient e from x to y if the number of tokens of color 1 in f_0 in $T(x)$ is greater than the number of tokens of color 1 in f_t in $T(x)$. Intuitively, the direction of an edge means that we need to move one or more tokens of color 1 from $T(x)$ to $T(y)$. If the two numbers are equal, we remove e from T. Let T' be the obtained directed forest. For an edge $e = (x, y)$ oriented from x to y, if x has a token of color 1 and y has a token of color 2, swapping the two tokens decreases D by one. We call such an edge a *desired* edge. We now show that there exists a desired edge in T'. Observe that if no vertex u with $f_0(u) = 1$ is incident to a directed edge in T', then indeed T' has no edge and $D = 0$. Let u be a vertex with $f_0(u) = 1$ that has at least one incident edge in T'. If u has no out-going edge, then the number of the color-1 tokens

in f_0 exceeds the number of the color-1 tokens in f_t. Thus we can choose an edge (u, v) oriented from u to v. If $f_0(v) = 2$ holds, the edge is desired. Now we assume that $f_0(v) = 1$ holds. We apply the same process for v, then an edge (v, w) oriented from v to w can be found. Since trees have no cycle, by repeating the process, we always find a desired edge. □

From Lemma 7, we can find a desired edge, and we swap the two tokens on the endpoints of the edge. Since a swap on a desired edge decreases D by one, by repeatedly swapping on desired edges, we obtain the swapping sequence of length D. Note that $D = 0$ if and only if $f_0(v) = f_t(v)$ for every $v \in V(T)$. Hence, we have $\mathrm{OPT}(f_0, f_t) \leq D$.

Therefore $\mathrm{OPT}(f_0, f_t) = D$ holds, and so we can solve 2-Colored Token Swapping by calculating D. The value $\mathrm{diff}(e)$ for every edge e, and thus the value D, can be calculated in a bottom-up manner from the edges incident to leaves of T in linear time in total. We have the following theorem.

Theorem 8. 2-Colored Token Swapping *is solvable in linear time for trees.*

4.4. c-Colored Token Swapping *on complete graphs*

In the previous subsections, we showed that 2-Colored Token Swapping can be solved in polynomial time for graphs and linear time for trees. In this subsection, let us consider if c-Colored Token Swapping for constant $c > 2$ can be solved in polynomial time for restricted graphs classes. We show that, for complete graphs, the c-Colored Token Swapping problem is fixed-parameter tractable when c is the parameter.

As a preliminary of this subsection, we first introduce a destination graph which represents the target vertex of each token. Let $G = (V, E)$ be a complete graph, and let f_0 and f_t be an initial token-placement and a target one. The *destination graph* $D(f_0, f_t) = (V_D, E_D)$ of two token-placement f_0 and f_t is the directed graph such that

- $V_D = V$; and
- there is an arc (u, v) from u to v if and only if $f_0(u) = f_t(v)$.

Upper bound

We here design an algorithm that computes a swapping sequence between f_0 and f_t to give an upper bound. Let us show an observation before describing the details of the algorithm. Let A be a cycle in $D(f_0, f_t)$, and we

assume that A is not a self-loop. We can move every token in A to its target vertex so that the tokens on the vertices in the other cycles are unchanged, as follows. First we choose any vertex v in A. We swap the two tokens on v and the out-neighbor u of v in A. After this swap, A is split into the two cycles A_1 and A_2: A_1 is a self-loop, that is the cycle with only u, and A_2 is the cycle with all the vertices in A except u. Then, u has its desired token and hence the remaining task is to move every token on the vertices in A_2. We repeat to swap the two tokens on v and its out-neighbor until v has its desired token. If a cycle has 3 or more vertices, the swap split the cycle into the two cycles: a self loop and a cycle with one less edges. If a cycle has 2 vertices, the swap split the cycle into two self-loops. Hence, the number of swaps to move all tokens in A to their target vertices is $|V(A)| - 1$.

We now describe the details of our algorithm. The algorithm uses a (vertex-disjoint) cycle cover of vertices in the destination graph. Let $\mathcal{C}(f_0, f_t)$ be a cycle cover of $D(f_0, f_t)$ and let A be a cycle in $\mathcal{C}(f_0, f_t)$. (We will present how to find a cycle cover later.) The algorithm swaps tokens cycle by cycle. If A is a self-loop, the algorithm does nothing. Let us assume that A is not a self-loop. The algorithm moves all tokens in A to their target vertices by $|V(A)| - 1$ swaps while keeping the rest unchanged. The algorithm repeats the same process for every cycle in $\mathcal{C}(f_0, f_t)$. The total number of swaps is

$$\sum_{A \in \mathcal{C}(f_0, f_t)} (|V(A)| - 1) = \sum_{A \in \mathcal{C}(f_0, f_t)} |V(A)| - |\mathcal{C}(f_0, f_t)| = n - |\mathcal{C}(f_0, f_t)| \,.$$

Thus we have the following upper bound.

Lemma 9. *Let G be a complete graph with n vertices, and let f_0 and f_t be initial and target token-placements. If there exists a vertex disjoint cycle cover $\mathcal{C}(f_0, f_t)$ of $D(f_0, f_t)$, we have the following inequality:*

$$OPT(f_0, f_t) \leq n - |\mathcal{C}(f_0, f_t)| \,.$$

If we apply the above lemma to a cycle cover such that the number of cycles in the cover is maximized, we obtain the smallest upper bound among cycle covers of $D(f_0, f_t)$. We call such a cycle cover *optimal*.

Corollary 10. *Let $\mathcal{C}_{OPT}(f_0, f_t)$ be an optimal cycle cover. Then we have*

$$OPT(f_0, f_t) \leq n - |\mathcal{C}_{OPT}(f_0, f_t)| \,.$$

Next, we show that the problem of finding an optimal cycle cover is fixed-parameter tractable when parameterized by the number of colors

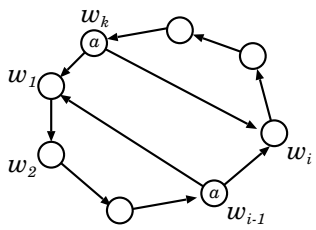

Figure 7: An illustration for Lemma 11.

Finding an optimal cycle cover

We show that, when c is the parameter, an optimal cycle cover of a destination graph can be found in FPT time. We reduce the problem into an integer linear program such that the number of variables and the number of constraints are both upper bounded by functions of c and all values in the program are nonnegative and at most n.

Lemma 11. *If A is a cycle in an optimal cycle cover $\mathcal{C}_{OPT}(f_0, f_t)$, then A contains at most c vertices.*

Proof. Let $A = \langle w_1, w_2, \ldots, w_k \rangle$. Suppose to the contrary that $k > c$. Then, there are two vertices with the same target color. Without loss of generality, assume that $f_t(w_1) = f_t(w_i) = a$. This implies that $f_0(w_k) = f_0(w_{i-1}) = a$. Hence, there exist the edges (w_k, w_i) and (w_{i-1}, w_1) in the destination graph. See Figure 7. Therefore, the vertices of A can be cover by the two disjoint cycles $A_1 = \langle w_1, w_2, \ldots, w_{i-1} \rangle$ and $A_2 = \langle w_i, w_{i+1}, \ldots, w_k \rangle$. This contradicts the optimality of $\mathcal{C}_{OPT}(f_0, f_t)$. □

We define "type" of vertices in a graph. The *type* of a vertex v is (a, b) if $f_t(v) = a$ and $f_0(v) = b$ hold. Intuitively, a vertex of type (a, b) has a token of color b in f_0 but the vertex desires to have a token of color a in f_t. The number of all possible types is c^2.

Now we reduce the problem of finding an optimal cycle cover to an integer linear program as follows. We first define a type sequence of a cycle. For a cycle $A = \langle w_1, w_2, \ldots, w_k \rangle$, the *type sequence* of A is a sequence $\langle tp(w_1), tp(w_2), \ldots, tp(w_k) \rangle$, where $tp(w_i)$ is the type of w_i. We list all possible type sequences that appear in $D(f_0, f_t)$. The length of a type sequence is at most c from Lemma 11, and the number of type sequences is at most $c!$. For each type sequence s, we prepare a variable x_s which represents how

many times the corresponding type sequences appear in a cycle cover. Let S be the set of type sequences. Finally, we have the following integer linear program:

$$\begin{aligned} \text{Maximize} \quad & \sum_{s \in S} x_s, \\ \text{Subject to} \quad & \sum_{\substack{s \in S \text{ with} \\ (a,b) \in s}} x_s = \#(a,b) \quad \text{for each } (a,b), \\ & x_s \geq 0 \quad \text{for each } s \in S, \end{aligned}$$

where $\#(a,b)$ is the number of vertices of type (a,b) in $D(f_0, f_t)$. This formula can be constructed in FPT time with parameter c.

The feasibility test of an ILP formula is fixed-parameter tractable when parameterized by the number of variables [8, 13, 11]. For our purpose, we need to solve the optimization version. We formally describe the problem and the theorem presented by Fellows et al. [5] below.

Problem: p-Variable Integer Linear Programming Optimization (p-Opt-ILP)
Instance: A matrix $\boldsymbol{A} \in \mathbb{Z}^{m \times p}$, and vectors $\boldsymbol{b} \in \mathbb{Z}^m$ and $\boldsymbol{c} \in \mathbb{Z}^p$.
Objective: Find a vector $\boldsymbol{x} \in \mathbb{Z}^p$ that minimizes $\boldsymbol{c}^\top \boldsymbol{x}$ and satisfies that $\boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b}$.
Parameter: p, the number of variables

Theorem 12 (Fellows et al. [5])**.** *p-Opt-ILP can be solved using $O(p^{2.5p+O(p)} \cdot L \cdot \log MN)$ arithmetic operations and space polynomial in L, where L is the number of bits in the input, N is the maximum absolute values any variable can take, and M is an upper bound on the absolute value of the minimum taken by the objective function.*

In our ILP formulation, (1) the number of variables is at most $c!$, (2) the constraints are represented using $\mathrm{O}(f(c) \log n)$ bits for some computable f, (3) each variable takes value at most n, and (4) the value of the objective function is at most n. Therefore, by Theorem 12, the ILP formulation can be solved in FPT time.

Lower bound

Let G be a complete graph and let f_0 and f_t be an initial token-placement and a target token-placement. Let $\mathcal{C}_{OPT}(f_0, f_t)$ an optimal cycle cover of the

destination graph $D(f_0, f_t)$. Now we define a potential function for f_0 and f_t:

$$\Phi(f_0, f_t) = n - |\mathcal{C}_{OPT}(f_0, f_t)|\,.$$

Note that $\Phi(f_t, f_t) = 0$ holds.

Let f be a token-placement of G, and let $\mathcal{C}(f, f_t)$ be a cycle cover of $D(f, f_t)$. Then, for $\mathcal{C}(f, f_t)$, we define an adjacent cycle cover as follows. Let f' be a token-placement adjacent to f, and assume that f' is obtained from f by swapping along an edge $e = (u, v)$. Let (u, u') and (v, v') be the directed edges leaving from u and v, respectively, in $\mathcal{C}(f, f_t)$. We define $\mathcal{C}'(f', f_t)$ as the cycle cover obtained by replacing (u, u') and (v, v') with (v, u') and (u, v'). Note that $\mathcal{C}'(f', f_t)$ is a cycle cover of $D(f, f_t)$. Then, we say that $\mathcal{C}'(f', f_t)$ is *adjacent* to $\mathcal{C}(f, f_t)$ with respect to e. Note that any cycle cover of $D(f', f_t)$ is adjacent to some cycle cover of $D(f, f_t)$ with respect to e.

Lemma 13. *For any adjacent two token-placements f and f', $\Phi(f', f_t) \geq \Phi(f, f_t) - 1$ holds.*

Proof. Suppose f' is obtained from f by swapping along an edge $e = (u, v)$. Let $\mathcal{C}(f, f_t)$ be a cycle cover of f, and let $\mathcal{C}'(f', f_t)$ be its adjacent cycle cover with respect to e. Now, we investigate how the number of cycles in $\mathcal{C}(f, f_t)$ changes by swapping along e.

Case 1: u and v are in the same cycle. See Figure 8(a).

Swapping along e splits the cycle including u and v into two cycles. Hence, the number of cycles increases by one.

Case 2: u and v are in different cycles. See Figure 8(b).

Swapping along e combines the two cycle including u and v into one cycle. Hence, the number of cycles decreases by one.

Hence, we have $|\mathcal{C}(f', f_t)| \leq |\mathcal{C}(f, f_t)| + 1$. Recall that any cycle cover of $D(f', f_t)$ is adjacent to a cycle cover of $D(f, f_t)$. Thus we also have $|\mathcal{C}_{OPT}(f', f_t)| \leq |\mathcal{C}_{OPT}(f, f_t)|+1$. Therefore, $\Phi(f', f_t) \geq \Phi(f, f_t)-1$ holds. □

From Lemma 13, we have the following lower bound.

Lemma 14. *Let G be a complete graph and let f_0 and f_t be an initial token-placement and a target one. Then we have*

$$OPT(f_0, f_t) \geq n - |\mathcal{C}_{OPT}(f_0, f_t)|\,.$$

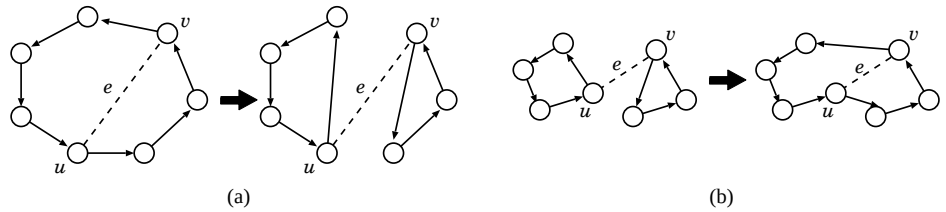

Figure 8: (a) An illustration for Case 1. The cycle in the left side is split into the two cycles in the right side. (b) An illustration for Case 2. The two cycles in the left side are combined into the cycle in the right side.

Therefore, we have the following theorem.

Theorem 15. *Given a complete graph, an initial token-placement f_0 and a target one f_t, c-*Colored Token Swapping *is fixed-parameter tractable when c is the parameter.*

Acknowledgments

This work is partially supported by MEXT/JSPS KAKENHI Grant Numbers 15H00853, 15K00008, 16K00002, 16K16006, 24106004, 24106007 and 26330009, JST CREST Foundation of Innovative Algorithms for Big Data, and the National Science and Engineering Research Council of Canada.

References

[1] É. Bonnet, T. Miltzow, and P. Rzążewski. Complexity of token swapping and its variants. In *34th Symposium on Theoretical Aspects of Computer Science, STACS 2017, March 8-11, 2017, Hannover, Germany*, pages 16:1–16:14, 2017.

[2] A. Cayley. Note on the theory of permutations. *Philosophical Magazine*, 34:527–529, 1849.

[3] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer Publishing Company, Incorporated, 1st edition, 2015.

[4] M. Dyer and A. Frieze. Planar 3DM is NP-complete. *Journal of Algorithms*, 7:174–184, 1986.

[5] M. Fellows, D. Lokshtanov, N. Misra, F. Rosamond, and S. Saurabh. Graph layout problems parameterized by vertex cover. In *19th International Symposium on Algorithms and Computation (ISAAC 2008)*, volume 5369 of *Lecture Notes in Computer Science*, pages 294–305, 2008.

[6] X. Feng, Z. Meng, and I. Sudborough. Improved upper bound for sorting by short swaps. In *IEEE Symposium on Parallel Architectures, Algorithms and Networks*, pages 98–103, 2004.

[7] X. Feng, I. Sudborough, and E. Lu. A fast algorithm for sorting by short swap. In *IASTED International Conference Computational and Systems Biology*, pages 62–67, 2006.

[8] A. Frank and É. Tardos. An application of simultaneous diophantine approximation in combinatorial optimization. *Combinatorica*, 7:49–65, 1987.

[9] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979.

[10] L. Heath and J. Vergara. Sorting by short swaps. *Journal of Computational Biology*, 10(5):775–789, 2003.

[11] J. H.W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983.

[12] M. Jerrum. The complexity of finding minimum-length generator sequence. *Theoretical Computer Science*, 36:265–289, 1985.

[13] R. Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12:415–440, 1987.

[14] J. Kawahara, T. Saitoh, and R. Yoshinaka. The time complexity of the token swapping problem and its parallel variants. In *WALCOM: Algorithms and Computation, 11th International Conference and Workshops, WALCOM 2017, Hsinchu, Taiwan, March 29-31, 2017, Proceedings.*, pages 448–459, 2017.

[15] D. Knuth. *The art of computer programming*, volume 3. Addison-Wesley, 2nd edition, 1998.

[16] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer, 2nd edition, 2005.

[17] T. Miltzow, L. Narins, Y. Okamoto, G. Rote, A. Thomas, and T. Uno. Approximation and hardness of token swapping. In *24th Annual European Symposium on Algorithms (ESA 2016)*, volume 57 of *LIPIcs*, pages 66:1–66:15, 2016.

[18] I. Pak. Reduced decompositions of permutations in terms of star transpositions, generalized catalan numbers and k-ary trees. *Discrete Mathematics*, 204:329–335, 1999.

[19] J. van den Heuvel. The complexity of change. In S. R. Blackburn, S. Gerke, and M. Wildon, editors, *Surveys in Combinatorics 2013*, volume 409 of *London Mathematical Society Lecture Note Series*, pages 127–160. Cambridge University Press, 2013.

[20] K. Yamanaka, E. Demaine, T. Ito, J. Kawahara, M. Kiyomi, Y. Okamoto, T. Saitoh, A. Suzuki, K. Uchizawa, and T. Uno. Swapping labeled tokens on graphs. *Theoretical Computer Science*, 586:81–94, 2015.