Token Shifting on Graphs

Win Hlaing Hlaing Myint, Ryuhei Uehara, and Giovanni Viglietta

School of Information Science, Japan Advanced Institute of Science and Technology (JAIST) {winhlainghlaingmyint,uehara,johnny}@jaist.ac.jp

Abstract. We investigate a new variation of a token reconfiguration problem on graphs using the cyclic shift operation. A colored or labeled token is placed on each vertex of a given graph, and a "move" consists in choosing a cycle in the graph and shifting tokens by one position along its edges. Given a target arrangement of tokens on the graph, our goal is to find a shortest sequence of moves that will re-arrange the tokens as in the target arrangement. The novelty of our model is that tokens are allowed to shift along any cycle in the graph, as opposed to a given subset of its cycles. We first discuss the problem on special graph classes: we give efficient algorithms for optimally solving the 2-Colored Token Shifting Problem on complete graphs and block graphs, as well as the Labeled Token Shifting Problem on complete graphs and variants of barbell graphs. We then show that, in the 2-Colored Token Shifting Problem, the shortest sequence of moves is NP-hard to approximate within a factor of $2-\varepsilon$, even for grid graphs. The latter result settles an open problem posed by Sai et al.

Keywords: reconfiguration problem · cyclic shift · barbell graph · block graph · NP-hard.

1 Introduction

Reconfiguration arises in countless problems that involve movement and change, including problems in computational geometry such as morphing graph drawings and polygons, and problems relating to games and puzzles, such as the 15-puzzle, a topic of research since 1879 [5]. The general questions that are considered in reconfiguration problems are: can any arrangement be reconfigured to any other; what is the worst-case number of steps required; and what is the complexity of computing the minimum number of steps required to get from one given configuration to another given configuration [5]. These questions can be rephrased in terms of the *configuration graph*, which is the graph whose vertices are all possible configurations, and whose edges represent feasible moves: is the configuration graph connected; what is its diameter; how efficiently can one compute distances between vertices in this graph? Previously studied token reconfiguration problems include the Token Swapping Problem, where pairs of tokens can be swapped along the edges of a graph. The Token Swapping Problem is proved to be NP-complete, and there are many special classes of graphs on which the Token Swapping Problem can be solved exactly by polynomial-time algorithms, including complete graphs, paths, cycles, stars, brooms, complete bipartite graphs, and complete split graphs (see, e.g., [2] for comprehensive surveys).

Recently, the Token Shifting Problem was introduced by Sai et al. in [6], inspired by puzzles based on cyclic shift operations. The input of the problem is a graph with a distinguished set of cycles $\mathcal{C}$, and an initial and a final arrangement of colored tokens on the vertices of the graph. The basic operation is called "shift" along a cycle $C \in \mathcal{C}$, and it moves each token located on a vertex of C into the next vertex along C. The problem asks for a sequence of shift operations that transforms the initial configuration into the final configuration. We can further distinguish between the Labeled Token Shifting Problem, where all tokens are distinct, and the k-Colored Token Shifting Problem, where tokens come in k different colors, and same-colored tokens are indistinguishable.

It was shown in [6] that the Labeled Token Shifting Problem is solvable in polynomial time on a large class of graphs, while solving the k-Colored Token Shifting Problem in the minimum number of moves is NP-hard, even for $k = 2$.

In this paper, we study a variation of the Token Shifting Problem where the set of cycles $\mathcal{C}$ consists of *all* cycles in the graph (as opposed to a subset of them). On one hand, our choice makes the problem's description more natural and compact; on the other hand, proving hardness results is now more challenging. Indeed, previous NP-hardess proofs for variations of the Token Shifting Problem crucially relied on the fact that only shifts along certain cycles were allowed.

In Section 3, we give linear-time algorithms for the shortest shift sequence for both the 2-Colored and the Labeled Token Shifting Problem for complete graphs. In Section 4, we discuss the shortest shift sequence for the Labeled Token Shifting Problem on standard barbell graphs, and then on generalized barbell graphs with more than one connecting edge. In Section 5, we study the 2-Colored Token Shifting Problem for block graphs. Finally, in Section 6 we prove that, in the 2-Colored Token Shifting Problem, the shortest sequence of moves is NP-hard to approximate within a factor of $2-\varepsilon$, even for planar graphs with a maximum degree of 4.

Notably, our NP-hardness result settles a problem left open in [6], which asked whether the Token Shifting Problem remains NP-hard when restricted to planar graphs or graphs of constant maximum degree. We remark that in [1], Amano et al. proved that a 2-Colored Token Shifting Problem called *Torus Puzzle* is NP-hard to solve in the minimum number of shifts. This puzzle consists of two arrays of horizontal and vertical cycles arranged in a grid, which yields a planar graph of maximum degree 4. However, in this puzzle the number of moves is measured in a different way: any number $k > 0$ of consecutive shifts along the same cycle is counted as only one move, while in our model (as well as in [6]) we count them as k moves. Because of this, the NP-hardness reduction in [1] does not work in our model. In addition, the majority of cycles in the graph of the Torus Puzzle are forbidden from shifting (such is, for example, the 4-cycle determined by any cell in the grid). However, as already remarked, in our model we insist on allowing shifts along any cycle.

2 Preliminaries

Let $G = (V, E)$ be an undirected connected graph, where V is the vertex set and E is the edge set, and let $\mathrm{Col} = \{1, 2, \ldots, c\}$ be the color set for tokens, where c is constant. A *token arrangement* (or *configuration*) is a function $f : V \to \mathrm{Col}$, where $f(v)$ represents the color of the token located on the vertex $v \in V$.

The *token shift operation* can be defined as follows. Let $C = (v_1, v_2, \ldots, v_k)$ be a cycle of $k > 1$ distinct vertices of $G = (V, E)$, where $\{v_i, v_{i+1}\} \in E$ for all $1 \le i < k$ and $\{v_k, v_1\} \in E$. Then, a token shift along C will transform any arrangement f into the arrangement f', which coincides with f on all vertices except the ones in C. Specifically, for $v_i \in \{v_1, v_2, \ldots, v_{k-1}\}$, we have $f'(v_{i+1}) = f(v_i)$, and $f'(v_1) = f(v_k)$. All cycles in G are eligible for token shift, and the length of the cycle can range from 2 to $|V|$. Note that we consider each edge of G as a cycle of length 2; in this case, the result of the shift operation will be equivalent to a token swap along that edge.

The *Token Shifting Problem* takes as input a connected graph $G = (V, E)$, a color set Col, an initial arrangement f_0, and a final arrangement f_t. The problem asks to determine a shortest sequence of shift operations OPT that transforms f_0 into f_t, assuming that such a sequence exists.

Note that, since swaps along edges are allowed, it is possible to transform f_0 into f_t if and only if they have the same number of tokens of each color, which is checkable in linear time given f_0 and f_t. Thus, without loss of generality, we may assume that there is always a sequence of shift operations that transforms f_0 into f_t, and our goal is to find the shortest one. Furthermore, it is easy to prove that $|\mathrm{OPT}| \le |V|(|V| - 1)/2$ (this bound is obtained by using swap operations only; cf. [7, Theorem 1]). Since we have a polynomial upper bound of the number of shift operations, the Token Shifting Problem is in NP.

We distinguish between the *k-Colored Token Shifting Problem*, where the size of Col is a fixed constant k, and the *Labeled Token Shifting Problem*, where $\mathrm{Col} = V$, and f_0 and f_t are permutations of V (that is, all tokens have distinct labels). In this paper, we will mostly focus on the 2-Colored Token Shifting Problem (i.e., where $\mathrm{Col} = \{c_1, c_2\}$) and the Labeled Token Shifting Problem.

3 Token Shifting on Complete Graphs

3.1 2-Colored Token Shifting on Complete Graphs

In this section, we show that for the 2-Colored Token Shifting Problem on complete graphs, an optimal shift sequence can be constructed in linear time.

Theorem 1. *The 2-Colored Token Shifting Problem on a complete graph $G = (V, E)$ can be solved in linear time by a single shift operation.*

Proof. Let $\mathrm{Col} = \{c_1, c_2\}$ be the color set and let f_0 and f_t be the initial and target token arrangements, respectively. We can construct two sets V_1 and V_2 of vertices as follows:

$$V_1 = \{v \in V \mid f_0(v) = c_1 \text{ and } f_t(v) = c_2\} \text{ and}$$

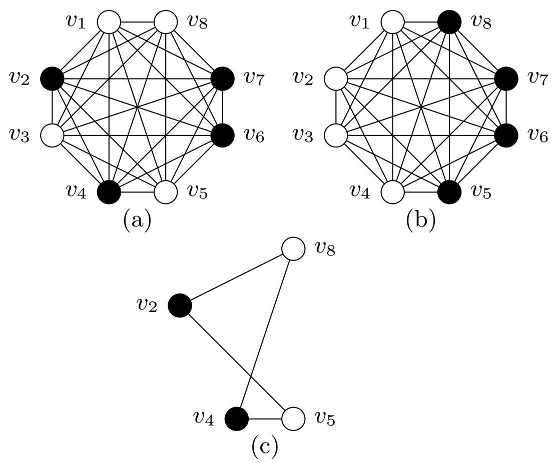

Fig. 1. 2-colored token shifting on a complete graph: (a) an initial token arrangement f_0, (b) a target token arrangement f_t, and (c) an optimal shift cycle

$$V_2 = \{v \in V \mid f_0(v) = c_2 \text{ and } f_t(v) = c_1\}.$$

Given that f_0 is re-configurable to f_t, $|V_1| = |V_2| = m$ for a complete graph with $2m$ misplaced tokens. Thus, we can construct a cycle of length $2m$ that visits each vertex in V_1 and V_2 alternately. For $V_1 = \{x_1, x_2, \ldots, x_m\}$ and $V_2 = \{y_1, y_2, \ldots, y_m\}$, the shift $(x_1, y_1, x_2, y_2, \ldots, x_m, y_m)$ transforms f_0 into f_t. □

For example, in Fig. 1, $V_1 = \{v_5, v_8\}$ and $V_2 = \{v_2, v_4\}$. From V_1 and V_2 the shift cycle (v_2, v_5, v_4, v_8) can be constructed, which transforms f_0 into f_t.

3.2 Labeled Token Shifting on Complete Graphs

In this section, we show that the Labeled Token Shifting Problem on a complete graph can be solved by at most two shift operations.

Theorem 2. *The Labeled Token Shifting Problem on a complete graph $G = (V, E)$ can be solved with a minimum shift sequence $|\mathrm{OPT}| \leq 2$ in linear time.*

Proof. Let f_0 and f_t be the initial and target token arrangements, respectively. We define the *conflict graph* $D(f_a, f_b) = (V', E')$ for two arrangements f_a and f_b as follows [7]:

$$V' = \{v \in V \mid f_a(v) \neq f_b(v)\} \text{ and}$$

$$E' = \{e = (v_i, v_j) \mid f_a(v_i) = f_b(v_j) \text{ and } v_i, v_j \in V'\}.$$

$D(f_0, f_t)$ is a digraph that includes vertices that hold different tokens in the initial and target token arrangements and there is an arc from v_i to v_j if the token on v_i needs to be moved to v_j. A simple example is given in Fig. 2. One way to transform f_0 to f_t would be to perform a token shift along each directed cycle

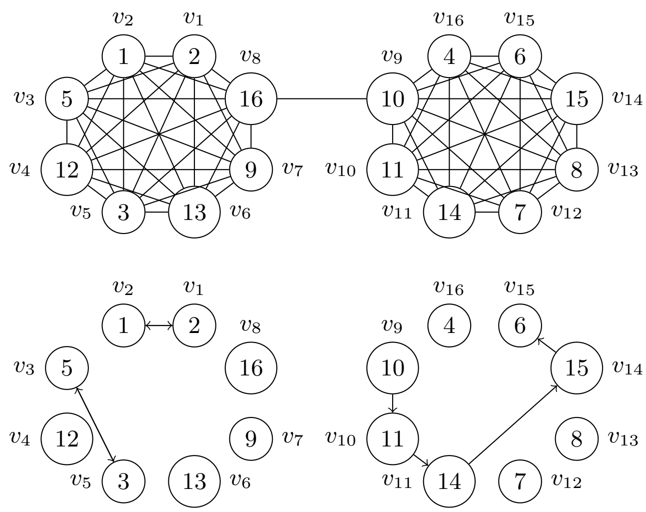

Fig. 2. (a) An initial token arrangement f_0, (b) the conflict graphs $D_A(f_0, f_t)$ and $D_B(f_0, f_t)$

in $D(f_0, f_t)$; if there are only 1 or 2 cycles, this strategy is optimal. However, it is not optimal when the number of cycles is greater than 2.

We consider the disjoint cycles in $D(f_0, f_t)$ as permutation cycles. For example, in Fig. 2(c) we have the three disjoint cycles (v_1, v_4), (v_2, v_6, v_3, v_7), and (v_5, v_8), which collectively correspond to the permutation (14)(2637)(58).

We will use the following general fact: let us be given m disjoint cyclic permutations involving n elements in total; the product of these m disjoint cycles and a length-m cycle consisting of one element from each disjoint cycle is a single length-n cycle that includes all n elements. For example, (**1**4)(**2**637)(**5**8)(**521**) = (18563724). Equivalently, (**1**4)(**2**637)(**5**8) = (18563724)(**125**). In other words, we can express the product of any set of $m > 2$ disjoint cyclic permutations as the product of only two cycles.

Therefore, we construct a first cycle including one vertex from each cycle in $D(f_0, f_t)$, and we shift along this cycle once. This will result in an arrangement f_1 whose conflict graph $D(f_1, f_t)$ consists of a single directed cycle (see Fig. 2(d)). We can then perform a single shift along this cycle to obtain the target token arrangement f_t. □

Corollary 1. *For the k-Colored Token Shifting Problem on a complete graph* $G = (V, E)$*, we have* $|\mathrm{OPT}| \leq 2$.

Proof. Let f_0 and f_t be the initial and final arrangements, respectively. Let $\mathrm{Col}' = V$, and let us define f_0' as an arbitrary bijection $f_0' \colon V \to \mathrm{Col}'$. We then define $f_t' \colon V \to \mathrm{Col}'$ as a bijection that, for all $v_i, v_j \in V$, satisfies $f_0'(v_i) = f_t'(v_j) \implies f_0(v_i) = f_t(v_j)$. Essentially, we assign unique labels to tokens in a way that is consistent with their colors. Thus, we obtain an instance of the Labeled Token Shifting Problem, which we can solve by Theorem 2. The same sequence of moves also solves the original instance, by construction. □

Note that, for the k-Colored Token Shifting Problem with $k > 2$, we do not have an efficient algorithm to determine when $|\mathrm{OPT}| = 1$ and when $|\mathrm{OPT}| = 2$. We leave this as an open problem.

4 Token Shifting on Barbell Graphs and Their Generalizations

In this section, we consider the Labeled Token Shifting Problem on barbell graphs and their generalization. A *barbell graph* is a simple graph obtained by connecting two complete graphs by an edge, which is called its *bar*. Our goal is to find the minimum shift sequence between initial and final token arrangements f_0 and f_t on a barbell graph. Then we extend our result to generalized barbell graphs that have two or more bars.

4.1 Token Shifting on Barbell Graphs

We first show that we can find the minimum shift sequence on a barbell graph in linear time. Let G be a barbell graph composed of two cliques A and B, each of size n, connected by a single edge: the bar.

The two cliques A and B contain n vertices each, from v_1 to v_n and from v_{n+1} to v_{2n}, respectively. The two vertices joined by the bar will be referred as *gate* vertices. Furthermore, we subdivide the tokens into two types, based on their matching vertices in the target arrangement: *local* tokens and *foreign* tokens, as follows. Tokens on vertices in a clique whose target vertices are in the other clique are referred to as *foreign* tokens. Let foreign(A) be the set of foreign tokens in A in f_0 and foreign(B) be the set of foreign tokens in B in f_0, as follows:

$$\mathrm{foreign}(A) = \{v_i \in V \,|\, f_0(v_i) = f_t(v_j) \text{ where } v_i \in A \text{ and } v_j \in B\},$$

$$\mathrm{foreign}(B) = \{v_i \in V \,|\, f_0(v_i) = f_t(v_j) \text{ where } v_i \in B \text{ and } v_j \in A\}.$$

Let $F = |\mathrm{foreign}(A)| = |\mathrm{foreign}(B)|$. In the following, we will prove that $3F - 2 \leq |\mathrm{OPT}| \leq 3F + 4$. Note that $|\mathrm{foreign}(A)| = |\mathrm{foreign}(B)| = F$ must hold in order for f_0 to be re-configurable to f_t. Let S_F be a shortest sequence of shifts that moves all $2F$ foreign tokens to their matching vertices. Note that this may still leave some non-foreign tokens on incorrect vertices; we will deal with re-configuring these tokens later.

Lemma 1. *In the Labeled Token Shifting Problem on a barbell graph, we have* $3F - 2 \leq |S_F| \leq 3F + 2$.

Proof. To transform f_0 to f_t, it is required for every foreign token on A and B to cross the bar at least once. Note that we can move two foreign tokens by performing a token exchange across the bar. In the worst case, a foreign token needs to be moved three times: from the current vertex to the nearest gate vertex, then across the bar to the gate vertex of the target clique, and then to the target vertex. Firstly, a foreign token on each clique must be moved to the gate vertex of that clique, which takes 2 shifts in total. Then, the actual exchange of tokens on gate vertices in a shift cycle (v_n, v_{n+1}) of length 2 occurs. Next, in each clique, the token on the gate vertex, say v_n, is moved to its target vertex v_i, while a new foreign token is moved from v_j to the gate. This is done with the single cycle (v_n, v_i, v_j). After the Fth exchange, we need one more shift in each clique to move the token from the gate vertex to its target vertex. Therefore, in the worst case we do F exchanging shifts and $2F+2$ local shifts, which is $3F+2$ shifts in total. However, we also need to consider the following special cases.

Condition 1. *A gate vertex already holds a foreign token in the initial arrangement f_0.*

If a gate vertex already holds a foreign token in the initial arrangement, then the initial shift for moving a foreign token to that gate vertex is not necessary. Hence, in the cases where A or B (or both) satisfy Condition 1, we need one (or two) fewer shift than $2F+2$.

Condition 2. *The target token of a gate vertex (i.e., the token that is on a gate vertex in f_t) is in the opposite clique in f_0.*

If this condition is satisfied, we can move that gate's final token across the bar in the Fth exchange. This way it is already in place when it enters the clique, and we can spare the final shift in that clique. Thus, in the extreme case where both gate vertices satisfy Conditions 1 and 2, and only $3F-2$ shifts are necessary. □

As for the local tokens, their target vertices are within the same clique. Hence, by Theorem 2, at most 2 shifts are necessary to solve the problem in each clique. We can now present this section's main result (for a proof, see the Appendix).

Theorem 3. *The Labeled Token Shifting Problem on a barbell graph $G=(V,E)$ can be solved with an optimal shift sequence in linear time, satisfying $3F-2 \leq |\mathrm{OPT}| \leq 3F+4$.* □

4.2 Token Shifting on Generalized Barbell Graphs with Two Bars

In this section, we extend our previous result to generalized barbell graphs. That is, we join two cliques by two bars instead of one, and this allows us to more effectively exploit the cyclic shift operation.

Let G be a generalized barbell graph with $2n$ vertices, with cliques A and B consisting of vertices from v_1 to v_n and v_{n+1} to v_{2n}, respectively. Two *bars* e_1 and e_2 connect A and B such that e_1 is incident to v_n and v_{n+1} and e_2 is incident to v_{n-1} and v_{n+2}. Let $F = |\text{foreign}(A)| = |\text{foreign}(B)|$, defined as in the previous section.

The proof of the next theorem is found in the Appendix.

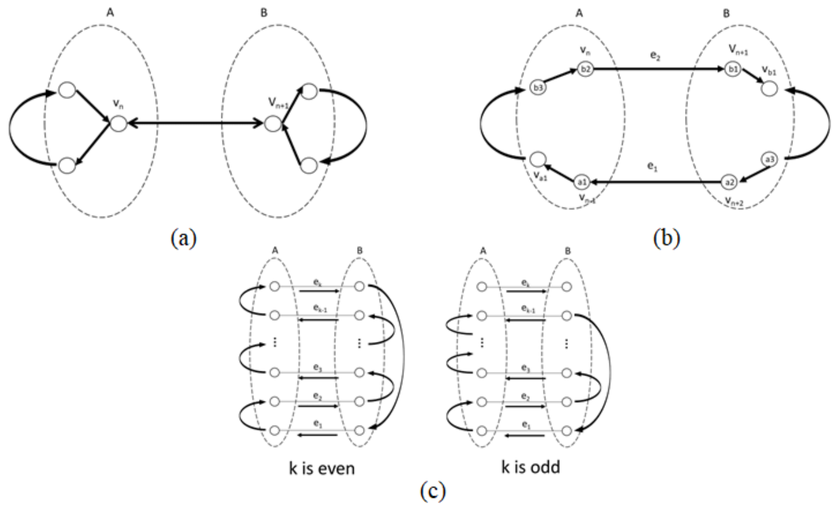

Fig. 3. Representation of token shifting on (a) a barbell graph, (b) a generalized barbell graph with 2 bars, and (c) a generalized barbell graph with $k > 2$ bars

Theorem 4. *The Labeled Token Shifting Problem on a generalized barbell graph* $G = (V, E)$ *with 2 bars can be solved with an optimal shift sequence in linear time, satisfying* $F \leq |\text{OPT}| \leq F + 4$. □

4.3 Token Shifting on Generalized Barbell Graphs with $k \geq 2$ Bars

For the next step, we discuss the Labeled Token Shifting Problem on generalized barbell graphs with $k > 2$ bars. Here, G is a graph consisting of two equal cliques A and B connected by k edges, called *bars*, such that no two bars are incident to the same vertex. Let $F = \text{foreign}(A) = \text{foreign}(B)$, defined as usual.

Theorem 5. *The Labeled Token Shifting Problem on a generalized barbell graph* $G = (V, E)$ *with* $k \geq 2$ *bars can be solved with an optimal shift sequence that satisfies* $F/\lfloor k/2 \rfloor \leq |\text{OPT}| \leq F/\lfloor k/2 \rfloor + 4$.

Proof. In the previous section, we proved that token shifting on a barbell graph with 2 connecting edges for $2F$ foreign tokens uses $F + 4$ shifts: 2 local shifts for moving foreign tokens on gate vertices at the start, F shifts for exchanging foreign tokens between cliques, and 2 local shifts to rearrange tokens within cliques. Now, while the number of local shifts remains the same, the number of exchanging shifts decreases as k increases.

Half of the k edges can be used to move the foreign tokens from A to B and another half of the k edges can be used to move foreign tokens from B to A. In one shift, we can exchange k tokens for even k and $k - 1$ tokens for odd k (see Figure 3(c)). Thus, for F tokens, we only need $F/\lfloor k/2 \rfloor$ shifts. □

5 2-Colored Token Shifting on Block Graphs

In this section, we discuss the 2-Colored Token Shifting Problem on block graphs. A block graph (or a clique tree) is a graph in which every bi-connected component (block) is a clique (see Fig. 4).

Definitions. In order to state this section's result, we need some definitions. Given a block graph $G = (V, E)$, where a block is a maximal clique, an *articulation point* is a vertex that belongs to more than one block. Let $P \subseteq V$ be the set of articulation points of G, and let K be the set of blocks of G. We define the *tree representation* of G (see [3]) as the undirected graph $T(G) = (V', E')$, where $V' = P \cup K$ and

$$E' = \{\{k, p\} \mid \text{the articulation point } p \in P \text{ lies in the block } k \in K\}.$$

When referring to $T(G)$, the nodes in P are called *articulation nodes*, and the nodes in K are called *clique nodes*. Fig. 4(c) shows an example of a tree representation. For a clique node $k \in K$, we write $I(k)$ to indicate the vertices of G that are in the block k but are not articulation points, i.e., $I(k) = k \setminus P$. Note that $I(k)$ induces a (possibly empty) clique in G.

Now, let $G = (V, E)$ be a block graph with n vertices, let $\mathrm{Col} = \{c_1, c_2\}$ be the color set, and let f_0 and f_t be the initial and target token arrangements on G. We say that an articulation node $p \in P$ *holds* color $c \in \mathrm{Col}$ if $f_0(p) = c$. Also, if f is an arbitrary arrangement, we write $n_c(f(p)) = 1$ if $f(p) = c$, and $n_c(f(p)) = 0$ otherwise. Similarly, for a clique node $k \in K$, let $n_c(f(k))$ be the number of c-colored tokens in $I(k) \subseteq V$ in the arrangement f. Then, we say that a clique node k of $T(G)$ *holds* color c if $n_c(f_0(k)) > n_c(f_t(k))$.

For each node x in $T(G)$, x has a *value* of $n_{c_1}(f_0(x)) - n_{c_1}(f_t(x))$. For each edge e in E' connecting two nodes $k \in K$ and $p \in P$, we define the number $\mathrm{diff}(e)$ as follows (cf. [8]). Let T_k be the subtree including node k resulted by the removal of e from $T(G)$. $n_{c_1}(f(T'))$ is the number of c_1 tokens on the set of vertices of G represented by T' in arrangement f. Then, $\mathrm{diff}(e) = n_{c_1}(f_t(T_k)) - n_{c_1}(f_0(T_k))$, i.e., the difference in number of c_1 tokens on T' between f_0 and f_t. For simplicity, $\mathrm{diff}(e)$ can be defined as the number of c_1 tokens (and, symmetrically, also c_2 tokens) that we must move along e to transform f_0 into f_t. If $\mathrm{diff}(e) = d > 0$, it means we need to move d tokens of color c_1 to k. If $\mathrm{diff}(e) = -d < 0$, it means we need to move d tokens of color c_2 to k.

Finally, we define $E'_k \subseteq E'$ to be the set of edge of $T(G)$ that are incident to the clique node k.

Theorem 6. *For the 2-Colored Token Shifting Problem on a block graph $G = (V, E)$, we have*

$$\sum_{k \in K} \max_{e \in E'_k} \{|\mathrm{diff}(e)|\} \le |\mathrm{OPT}| \le \sum_{k \in K} \max \left\{ \sum_{\substack{e \in E'_k \\ \mathrm{diff}(e) > 0}} \mathrm{diff}(e),\ \sum_{\substack{e \in E'_k \\ \mathrm{diff}(e) < 0}} |\mathrm{diff}(e)|,\ 1 \right\},$$

and a shift sequence within these bounds can be computed in $O(n^2)$ time.

Proof. For the upper bound, we will give a procedure for finding a shift sequence. We first construct the tree representation $T(G)$ in $O(n^2)$ time. From $T(G)$, we determine the sequence of shifts by deciding on which clique the shift must be performed in each step (note that, in a block graph, every cycle is included in a single clique).

For a clique k with an excess of c_1 tokens connected to an articulation vertex p, some c_1 tokens in k must be moved out and some c_2 tokens must be moved in through p. We need to perform a shift that moves the extra c_1 token in k to the articulation vertex p and the c_2 tokens on p to the target vertex in k. On $T(G)$, it will be a token exchange between a clique node k that holds color c_1 and the articulation node p that holds color c_2 along the edge $e = \{k, p\} \in E'$. This exchange will decrease $|\text{diff}(e)|$ and change the color of p to c_1. However, in the case where the p holds the same color c_1 as k, it is pointless to perform a shift between them. The same goes for a clique with $n_{c_2}(f_0(k)) > n_{c_2}(f_t(k))$. If $\text{diff}(e) = 0$, no token needs to be moved across e, and e can be removed from $T(G)$. For G to achieve the target arrangement f_t, all the edges in $T(G)$ must be removed. Thus, we can construct the shift sequence for G from $T(G)$ by determining the clique nodes for an exchange in each step.

We now discuss how to choose a feasible clique node for token exchange. There are three types of clique nodes in $T(G)$: (1) leaf node, (2) non-leaf node, and (3) isolated node.

A leaf node is a clique node with an articulation node, the removal of which will disconnect the clique node from the other clique nodes in $T(G)$. When we look for a clique for token exchange, we start with the leaf nodes and go up the tree $T(G)$. A leaf node k connected to node p by edge e is feasible for an exchange if k and p hold different colors and $|\text{diff}(e)| > 0$.

Non-leaf nodes are those with multiple articulation nodes connecting them to other clique nodes in $T(G)$. In non-leaf nodes, we can exchange one or more pairs of different color tokens in one shift. For a non-leaf node k with m articulation nodes $p_1, p_2, \ldots, p_m$, k is feasible for an exchange (1) if there are one or more edges $e = (k, p)$ with $|\text{diff}(e)| > 0$, and k and p hold different colors, where $p \in \{p_1, p_2, \ldots, p_m\}$ and k has non-zero value or (2) if k is connected to one or more pairs of articulation nodes p_i and $p_j \in \{p_1, p_2, \ldots, p_m\}$ where p_i and p_j hold different colors, and $\text{diff}(e_i = \{k, p_i\})$ and $\text{diff}(e_j = \{k, p_j\})$ have opposite sign (one positive, one negative).

An isolated node is already disconnected from other clique nodes in $T(G)$ and the amount of both c_1 and c_2 tokens in it is the same for f_0 and f_t. For each isolated node k with no edge in $T(G)$, if $f_0(k) \neq f_t(k)$, then one shift suffices to reach the target arrangement as $n_c(f_0(k)) = n_c(f_t(k))$, $c \in \{c_1, c_2\}$.

As for the lower bound, we observe that, for each clique node k, we can only move one token to or from each articulation point in a shift and decrease the $|\text{diff}(e)|$ of each edge by one. Therefore, if k is incident to an edge e with $|\text{diff}(e)| = d$, then at least d shifts must be performed in the clique corresponding to k. Thus, to remove all the edges incident to a clique node k in $T(G)$, at least $\max_{e \in E'_k} \{|\text{diff}(e)|\}$ shifts are necessary. □

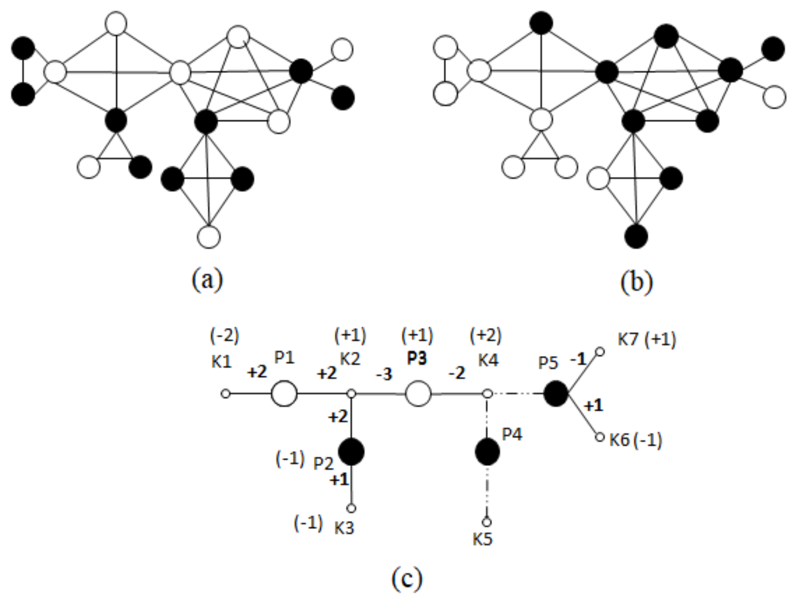

Fig. 4. (a) Initial arrangement f_0, (b) target arrangement f_t, and (c) tree representation $T(G)$ of block graph G with positive values over nodes that need black tokens, negative values over nodes that need white tokens, $\text{diff}(e)$ values over each edge e, and dotted lines for removed edges

6 Hardness of 2-Colored Token Shifting

In this section, we show that a shortest shift sequence for the 2-Colored Token Shifting Problem is not only NP-hard to compute, but also NP-hard to approximate within a factor of $2 - \varepsilon$, for any $\varepsilon > 0$. This is true even if the graph G is a grid graph, hence planar and with maximum degree 4. We will prove it by a reduction from the NP-complete problem of deciding if a grid graph has a Hamiltonian cycle, i.e., a cycle involving all vertices [4].

Theorem 7. *The optimal shifting sequence for the 2-Colored Token Shifting Problem is NP-hard to approximate within a factor of* $2 - \varepsilon$*, for any* $\varepsilon > 0$*, even for grid graphs.*

Proof. Let $G = (V, E)$ be a connected grid graph (i.e., a vertex-induced finite subgraph of the infinite grid), and let a *checkered arrangement* be an arrangement of two-colored tokens on G such that tokens on any two adjacent vertices have different colors. Note that, for any given G, there are exactly two different checkerboard arrangements.

Our reduction maps the grid graph G to the 2-Colored Token Shifting Problem on the same graph G, where the initial arrangement f_0 and the target arrangement f_t are the two distinct checkerboard arrangements (see Figure 5).

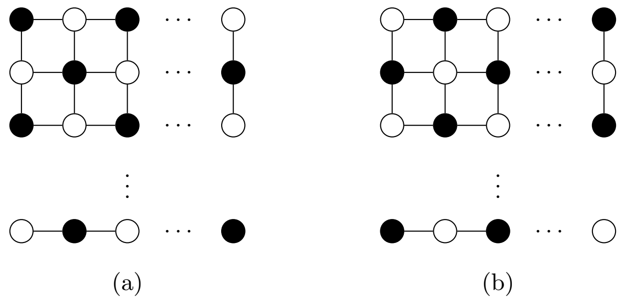

Fig. 5. (a) Initial arrangement f_0 and (b) target arrangement f_t

Observe that $f_0(v) \neq f_t(v)$ for all $v \in V$, and thus a sequence of shift operations that transforms f_0 into f_t must move every token at least once. More precisely, f_t is reached if and only if every token takes part in an odd number of shift operations. If G has a Hamiltonian cycle C, then the shift operation along C immediately transforms f_0 into f_t, and hence $|\mathrm{OPT}| = 1$. Conversely, if $|\mathrm{OPT}| = 1$, the single shift operation that transforms f_0 into f_t must involve every vertex, and thus it must be a Hamiltonian cycle.

We have proved that, if G has a Hamiltonian cycle, then $|\mathrm{OPT}| = 1$, and that if G does not have a Hamiltonian cycle, then $|\mathrm{OPT}| \geq 2$. Thus, if we could compute an approximation of $|\mathrm{OPT}|$ within a factor of $2-\varepsilon$ in polynomial time, we would also be able to decide if G has a Hamiltonian cycle. Since the latter problem is NP-hard [4], then so is the former problem. □

References

1. Amano, K., Kojima, Y., Kurabayashi, T., Kurihara, K., Nakamura, M., Omi, A., Tanaka, T., Yamazaki, K.: How to solve the torus puzzle. Algorithms **5**(1), 18–29 (2012)
2. Biniaz, A., Jain, K., Lubiw, A., Masárová, Z., Miltzow, T., Mondal, D., Naredla, A.M., Tkadlec, J., Turcotte, A.: Token swapping on trees. arXiv preprint arXiv:1903.06981 (2019)
3. Brimkov, B., Hicks, I.V.: Memory efficient algorithms for cactus graphs and block graphs. Discrete Applied Mathematics **216**, 393–407 (2017)
4. Itai, A., Papadimitriou, C.H., Szwarcfiter, J.L.: Hamilton paths in grid graphs. SIAM Journal on Computing **11**(4), 676–686 (1982)
5. Nishimura, N.: Introduction to reconfiguration. Algorithms **11**(4), 52 (2018)
6. Sai, K.K., Uehara, R., Viglietta, G.: Cyclic shift problems on graphs. In: WALCOM: Algorithms and Computation. pp. 308–320. Lecture Notes in Computer Science Vol. 12635, Springer-Verlag (2021)
7. Yamanaka, K., Demaine, E.D., Ito, T., Kawahara, J., Kiyomi, M., Okamoto, Y., Saitoh, T., Suzuki, A., Uchizawa, K., Uno, T.: Swapping labeled tokens on graphs. Theoretical Computer Science **27**, 81–94 (June 2015)
8. Yamanaka, K., Horiyama, T., Kirkpatrick, D., Otachi, Y., Saitoh, T., Uehara, R., Uno, Y.: Swapping colored tokens on graphs. In: Workshop on Algorithms and Data Structures. pp. 619–628. Springer (2015)

Appendix: Missing Proofs

Theorem 3. *The Labeled Token Shifting Problem on a barbell graph $G = (V, E)$ can be solved with an optimal shift sequence in linear time, satisfying $3F - 2 \leq |\mathrm{OPT}| \leq 3F + 4$.*

Proof. We can classify each vertex into one of three types by constructing conflict graphs for A and B. A vertex either (1) already holds its target token, (2) belongs to a directed cycle such as $(v_i, v_j, \ldots, v_k)$ where v_k holds token i, v_i holds token j or (3) belongs to a chain of vertices that cannot form a cycle such as $v_i, v_j, \ldots, v_k$ where v_i holds token j, v_k holds a foreign token j, and token i belongs to another clique.

Type-1 vertices need no consideration. As for type-3 vertices, they can be solved while exchanging foreign tokens. Since token i must reach gate v_n after an exchange at some point, we can then perform the cyclic shift $(v_i, v_j, \ldots, v_k, v_n)$. This will move the token i to v_i, token j to v_j, token k to v_k and lastly the foreign token on v_k to v_n. This not only matches the vertices $v_i, v_j, \ldots, v_k$ with their tokens but also moves a foreign token to v_n for the next exchange. We now consider how to deal with the type-2 vertices. As they are isolated from the type-3 vertices, they cannot be solved while exchanging the foreign tokens. Hence, to avoid additional shifts, we connect the directed cycles to a chain of type-3 tokens. We can do that by performing a shift that includes a type-2 vertex from each directed cycle and a type-3 vertex while moving a foreign token to the gate vertex. This way, we can handle the local tokens while exchanging foreign tokens and $|\mathrm{OPT}| = |S_F|$. However, this is true only when $|S_F| \geq 5$.

Let us now discuss the exceptional case where the minimum shift sequence required for exchanging foreign tokens satisfies $|S_F| < 5$. In this case, fewer than than two shifts are performed on one of the cliques during the exchange. When $F = 0$, the problem becomes two independent token shifting problems on two complete graphs, which may need 4 shifts in total. Thus, $3F + 2$ is no longer an upper bound.

We can conclude that the shortest shift sequence for token shifting on a barbell graph is $|\mathrm{OPT}| = |S_F| = 3F+2$ in the general case without Conditions 1, 2, and excluding the exceptional case discussed above. We can now compute optimal bounds on the minimum shift sequence from the extreme cases as follows. For a case with $F = 0$, we need at most 4 shifts for solving the problem on two complete graphs independently, and so $|\mathrm{OPT}| = 3F + 4$ holds. For a case with Conditions 1 and 2 on both sides, we have the minimum sequence of $|\mathrm{OPT}| = 3F-2$. We can easily determine whether those conditions hold in linear time. □

Theorem 4. *The Labeled Token Shifting Problem on a generalized barbell graph $G = (V, E)$ with 2 bars can be solved with an optimal shift sequence in linear time, satisfying $F \leq |\mathrm{OPT}| \leq F + 4$.*

Proof. As discussed before, an exchange needs two steps: moving foreign tokens on each clique to the gate vertices and the actual exchange of tokens on gate vertices. In a barbell graph with 2 bars, we can combine the two steps into one by exchanging foreign tokens and bringing the foreign tokens to the gate vertices for the next exchange in a single shift. Since each clique now has two gate vertices, one vertex acts as the entry gate vertex where the incoming tokens pass through and another acts like the exit gate vertex through which the foreign tokens leave. Between the cliques A and B, the two bars e_1 and e_2 act like two lanes going in opposite directions.

Let v_n and v_{n-1} be the gate vertices of A and v_{n+1} and v_{n+2} be the gate vertices of B such that v_n and v_{n+1} are connected by e_1 and v_{n-1} and v_{n+2} are connected be e_2. In a single shift, we can move a foreign token b_3 inside A to v_n (exit of A), b_2 on v_n to v_{n+1} (entry of B), and b_1 on v_{n+1} to v_{b_1} inside B. Also, move a foreign token a_3 inside B to v_{n+2} (exit of B), a_2 on v_{n+2} to v_{n-1} (entry of A), and a_1 on v_{n-1} to v_{a_1} inside A (see Figure 3(b)).

Therefore, $|S_F|$ is reduced to $F + 4$ (F exchanging shifts, 2 pre-exchange shifts, and 2 post-exchange shifts). The generalized barbell graphs with 2 bars also have two exceptional conditions, corresponding to Conditions 1 and 2 in Lemma 1.

In this case of Condition 1, we need one less shift than $F + 4$. If the gate vertices v_a of A and v_b of B are not adjacent and both vertices hold foreign tokens, we can start exchanging tokens immediately and need 2 fewer shifts.

In the case of Condition 2, the target token of a gate vertex lies in the opposite clique. This token can be exchanged last to save 1 shift. If both A and B satisfy Conditions 1 and 2, then $|S_F| = F$.

We can deal with the local tokens in a similar way as in Section 4.1, so that no additional shift is necessary for moving local tokens when $|S_F| \geq 2$.

In the exceptional case where the minimum shifts required for exchanging foreign tokens is $|S_F| < 2$, local tokens cannot be handled by S_F.

We can now work out exact bounds on the minimum shift sequence from the extreme cases as follows. For a case with $F = 0$, we need at most 4 shifts for solving the two cliques separately, and $|\text{OPT}| = F + 4$ holds. For a case with Conditions 1 and 2 on both sides, we have the minimum shift sequence of $|\text{OPT}| = F$. □