


Algorithms for Coloring Reconfiguration Under Recolorability Constraints

Hiroki Osawa

Graduate School of Information Sciences, Tohoku University, Japan
osawa@ecei.tohoku.ac.jp


Akira Suzuki¹

Graduate School of Information Sciences, Tohoku University, Japan
a.suzuki@ecei.tohoku.ac.jp

 <https://orcid.org/0000-0002-5212-0202>

Takehiro Ito²

Graduate School of Information Sciences, Tohoku University, Japan
takehiro@ecei.tohoku.ac.jp

 <https://orcid.org/0000-0002-9912-6898>

Xiao Zhou³

Graduate School of Information Sciences, Tohoku University, Japan
zhou@ecei.tohoku.ac.jp

Abstract

COLORING RECONFIGURATION is one of the most well-studied reconfiguration problems. In the problem, we are given two (vertex-)colorings of a graph using at most k colors, and asked to determine whether there exists a transformation between them by recoloring only a single vertex at a time, while maintaining a k -coloring throughout. It is known that this problem is solvable in linear time for any graph if $k \leq 3$, while is PSPACE-complete for a fixed $k \geq 4$. In this paper, we further investigate the problem from the viewpoint of recolorability constraints, which forbid some pairs of colors to be recolored directly. More specifically, the recolorability constraint is given in terms of an undirected graph R such that each node in R corresponds to a color, and each edge in R represents a pair of colors that can be recolored directly. In this paper, we give a linear-time algorithm to solve the problem under such a recolorability constraint if R is of maximum degree at most two. In addition, we show that the minimum number of recoloring steps required for a desired transformation can be computed in linear time for a yes-instance. We note that our results generalize the known positive ones for COLORING RECONFIGURATION.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases combinatorial reconfiguration, graph algorithm, graph coloring

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2018.37

¹ Partially supported by JST CREST Grant Number JPMJCR1402, and JSPS KAKENHI Grant Numbers JP17K12636 and JP18H04091, Japan.

² Partially supported by JST CREST Grant Number JPMJCR1402, and JSPS KAKENHI Grant Numbers JP16K00004 and JP18H04091, Japan.

³ Partially supported by JSPS KAKENHI Grant Number JP16K00003, Japan.



© Hiroki Osawa, Akira Suzuki, Takehiro Ito, and Xiao Zhou;
licensed under Creative Commons License CC-BY

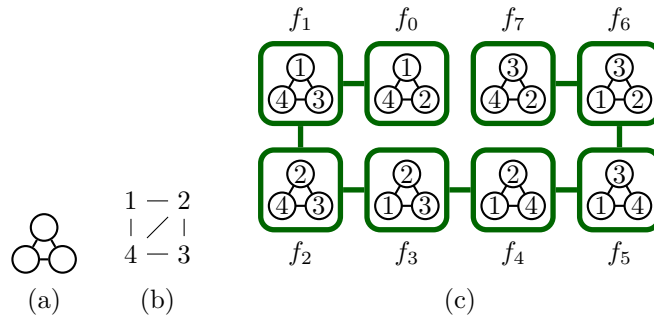
29th International Symposium on Algorithms and Computation (ISAAC 2018).

Editors: Wen-Lian Hsu, Der-Tsai Lee, and Chung-Shou Liao; Article No. 37; pp. 37:1–37:13

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



■ **Figure 1** (a) An input graph G , (b) a recolorability graph R with four colors 1, 2, 3 and 4, and (c) an $(f_0 \rightarrow f_7)$ -reconfiguration sequence.

1 Introduction

Combinatorial reconfiguration [10, 11, 13] has been studied intensively in the field of theoretical computer science. In a typical reconfiguration problem, we are given two feasible solutions of a search problem instance (e.g., graph colorings, independent sets, satisfying truth assignments), and asked to check the existence of a step-by-step transformation between them such that all intermediate results are also feasible and each step conforms to a fixed reconfiguration rule, that is, an adjacency relation defined on feasible solutions of the original search problem instance.

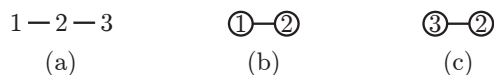
For example, the COLORING RECONFIGURATION problem is one of the most well-studied reconfiguration problems, defined as follows [3, 7]. For an integer $k \geq 1$, we are given two k -colorings f_0 and f_r of the same graph G , and asked to determine whether there exists a sequence $\langle f_0, f_1, \dots, f_\ell \rangle$ of k -colorings of G such that $f_\ell = f_r$ and f_i is obtained from f_{i-1} by recoloring a single vertex of G for each $i \in \{1, 2, \dots, \ell\}$. Figure 1(c) shows an example of a desired sequence $\langle f_0, f_1, \dots, f_7 \rangle$ of 4-colorings, where G is a complete graph K_3 as illustrated in Figure 1(a).

The complexity of COLORING RECONFIGURATION has been clarified based on several “standard” measures (e.g., the number of colors [3, 7, 12] and graph classes [1, 2, 5, 8, 9, 16]) which are used well also for analyzing the original search problem. On the other hand, in [14], we have introduced a new concept, called the *recolorability constraint* on colors, to analyze the complexity of COLORING RECONFIGURATION more precisely. This concept is newly tailored for COLORING RECONFIGURATION, and forbids some pairs of colors to be recolored directly.

1.1 Our problem

For an integer $k \geq 1$, let C be the *color set* of k colors $1, 2, \dots, k$. Let G be a graph with vertex set $V(G)$ and edge set $E(G)$. Recall that a k -coloring of G is a mapping $f : V(G) \rightarrow C$ such that $f(v) \neq f(w)$ holds for any edge $vw \in E(G)$. The *recolorability* on C is given in terms of an undirected graph R , called the *recolorability graph* on C , such that $V(R) = C$; each edge $ij \in E(R)$ represents a “recolorable” pair of colors $i, j \in V(R) = C$. Then, two k -colorings f and f' of G are *adjacent (under R)* if the following two conditions hold:

- (a) $|\{v \in V(G) : f(v) \neq f'(v)\}| = 1$, that is, f' can be obtained from f by *recoloring* a single vertex $v \in V(G)$; and
- (b) if $f(v) \neq f'(v)$ for a vertex $v \in V(G)$, then $f(v)f'(v) \in E(R)$, that is, the colors $f(v)$ and $f'(v)$ form a recolorable pair.



■ **Figure 2** (a) Recolorability graph R with three colors 1, 2 and 3, and (b) and (c) 3-colorings f_0 and f_r of a graph consisting of a single edge, respectively.

For each $i \in \{1, 2, \dots, 7\}$, two 4-colorings f_{i-1} and f_i in Figure 1(c) are adjacent under the recolorability graph R in Figure 1(b). Note that the known adjacency relation for COLORING RECONFIGURATION requires only Condition (a) above, that is, we can recolor a vertex from any color to any color directly. Observe that this corresponds to the case where R is a complete graph of size k , and hence our adjacency relation generalizes the known one.

Given a graph G , two k -colorings f_0 and f_r of G , and a recolorability graph R on C , the COLORING RECONFIGURATION problem UNDER RECOLORABILITY is the decision problem of determining whether there exists a sequence $\langle f_0, f_1, \dots, f_\ell \rangle$ of k -colorings of G such that $f_\ell = f_r$ and f_{i-1} and f_i are adjacent under R for all $i \in \{1, 2, \dots, \ell\}$; such a desired sequence is called an $(f_0 \rightarrow f_r)$ -reconfiguration sequence, and its *length* (i.e., the number of recoloring steps) is defined as ℓ . For example, the sequence $\langle f_0, f_1, \dots, f_7 \rangle$ in Figure 1(c) is an $(f_0 \rightarrow f_7)$ -reconfiguration sequence whose length is seven.

We emphasize that the concept of recolorability constraints changes the reachability of k -colorings drastically. For example, the $(f_0 \rightarrow f_7)$ -reconfiguration sequence in Figure 1(c) is a shortest one between f_0 and f_7 under the recolorability graph R in Figure 1(b). However, in COLORING RECONFIGURATION (in other words, if R would be K_4 and would have the edge joining colors 1 and 3), we can recolor the (top) vertex of G from 1 to 3 directly. As another example, the instance illustrated in Figure 2 is a no-instance for our problem, but is a yes-instance for COLORING RECONFIGURATION with $k = 3$.

1.2 Related and known results

As we mentioned, COLORING RECONFIGURATION has been studied intensively [1, 2, 3, 4, 5, 7, 8, 9, 12, 15, 16]. In particular, a sharp analysis has been obtained from the viewpoint of the number k of colors: Bonsma and Cereceda [3] proved that COLORING RECONFIGURATION is PSPACE-complete even for a fixed $k \geq 4$. On the other hand, Cereceda et al. [7] proved that COLORING RECONFIGURATION is solvable in polynomial time for any graph if $k \in \{1, 2, 3\}$. Brewster et al. [6] generalized this sharp analysis to CIRCULAR COLORING RECONFIGURATION. We also note that Johnson et al. [12] gave a linear-time algorithm to solve COLORING RECONFIGURATION for any graph and $k \in \{1, 2, 3\}$; indeed, their algorithm can determine in linear time whether an $(f_0 \rightarrow f_r)$ -reconfiguration sequence exists or not, and can compute its shortest length in linear time if it exists.

In [14], we introduced the concept of recolorability constraints, and showed the computational hardness of COLORING RECONFIGURATION UNDER RECOLORABILITY based on the graph structure of recolorability graphs R . More specifically, we proved that the problem is PSPACE-complete if (1) R is of maximum degree at least four, or (2) R contains a connected component having at least two cycles. These results are strong in the sense that they show the PSPACE-completeness for *all* recolorability graphs satisfying (1) or (2). Furthermore, the latter result (2) implies that the problem is PSPACE-complete if $R = K_4$. Therefore, the results (1) and (2) generalize the known PSPACE-completeness for COLORING RECONFIGURATION with $k \geq 4$. In this sense, the results in [14] gave a sharper analysis and a better understanding of the computational hardness of COLORING RECONFIGURATION.

1.3 Our contribution

Despite the concept of recolorability graphs R generalized and sharpened the known PSPACE-completeness successfully, there is no algorithmic (positive) result for COLORING RECONFIGURATION UNDER RECOLORABILITY except for the special case of $R = K_3$ obtained from COLORING RECONFIGURATION [7, 12]. In this paper, we thus study the polynomial-time solvability of our problem, and generalize the known algorithmic results from the viewpoint of the graph structure of recolorability graphs. Specifically, our main result can be stated as the following theorem:

► **Theorem 1.** *Suppose that a recolorability graph R is of maximum degree at most two, and let $k = |V(R)|$. For any graph G with n vertices and m edges, COLORING RECONFIGURATION UNDER RECOLORABILITY can be solved in $O(k + n + m)$ time. Furthermore, if an $(f_0 \rightarrow f_r)$ -reconfiguration sequence exists for two k -colorings f_0 and f_r of G , then*

- *its shortest length can be computed in $O(k + n + m)$ time; and*
- *a shortest $(f_0 \rightarrow f_r)$ -reconfiguration sequence can be output in $O(kn(n + m))$ time.*

We emphasize that Theorem 1 holds for any graph G , and only the structure of R is restricted. Since K_3 is of maximum degree two, Theorem 1 generalizes the known positive results for COLORING RECONFIGURATION [7, 12]. Note that k is not always a constant (indeed, can be larger than n).

In this paper, we prove Theorem 1 as follows. We start by giving an observation that a recolorability graph R can be assumed to be connected without loss of generality (Section 2). Then, since the maximum degree of R is two, R is either a path or a cycle. In Section 3, we will prove Theorem 1 for the case where R is a path. Sections 4 and 5 are devoted to the case where R is a cycle; the algorithm in Section 4 only checks whether a given instance is a yes-instance or not, and the one in Section 5 computes the shortest length for a yes-instance.

Due to the page limitation, proofs of the claims marked with (*) are omitted from this extended abstract.

2 Preliminaries

Since we deal with (vertex-)coloring, we may assume without loss of generality that an input graph G is simple, connected and undirected. Let $n = |V(G)|$ and $m = |E(G)|$. For a vertex subset $V' \subseteq V(G)$, we denote by $G[V']$ the subgraph of G induced by V' .

For a graph G and a recolorability graph R on C , we define the R -reconfiguration graph on G , denoted by $\mathcal{C}_R(G)$, as follows: $\mathcal{C}_R(G)$ is an undirected graph such that each node of $\mathcal{C}_R(G)$ corresponds to a k -coloring of G , and two nodes in $\mathcal{C}_R(G)$ are joined by an edge if their corresponding k -colorings are adjacent under R . We sometimes call a node in $\mathcal{C}_R(G)$ simply a k -coloring if it is clear from the context. A path in $\mathcal{C}_R(G)$ from a k -coloring f to another one f' is called an $(f \rightarrow f')$ -reconfiguration sequence. Note that any $(f \rightarrow f')$ -reconfiguration sequence is *reversible*, that is, the path in $\mathcal{C}_R(G)$ forms an $(f' \rightarrow f)$ -reconfiguration sequence, too. Then, the COLORING RECONFIGURATION problem UNDER RECOLORABILITY can be seen as the decision problem of determining whether $\mathcal{C}_R(G)$ contains an $(f_0 \rightarrow f_r)$ -reconfiguration sequence for two given k -colorings f_0 and f_r of G . Note that the problem does not ask for an actual $(f_0 \rightarrow f_r)$ -reconfiguration sequence as the output. We always denote by f_0 and f_r two given k -colorings of G as an input of the problem. For two k -colorings of f and f' in $\mathcal{C}_R(G)$, we denote by $\text{dist}(f, f')$ the shortest length (i.e., the minimum number of edges in $\mathcal{C}_R(G)$) of an $(f \rightarrow f')$ -reconfiguration sequence if it exists; otherwise we let $\text{dist}(f, f') = +\infty$.

We note that a given recolorability graph R can be assumed to be connected without loss of generality. To see this, first observe that no $(f_0 \rightarrow f_r)$ -reconfiguration sequence exists if there is a vertex $u \in V(G)$ such that the colors $f_0(u)$ and $f_r(u)$ belong to different connected components of R . Next, consider any two vertices $v, w \in V(G)$ such that the colors $f_0(v)$ and $f_0(w)$ belong to different connected components R_1 and R_2 of R , respectively. Then, since $V(R_1) \cap V(R_2) = \emptyset$, we can independently recolor vertices v and w . In this way, we can assume without loss of generality that R is connected.

To describe our algorithms, we sometimes use the notion of digraphs (i.e., directed graphs). For an undirected graph G , we denote by \vec{G} a digraph whose underlying graph is G , and also denote by $A(\vec{G})$ the arc set of \vec{G} . We denote by vw an edge joining two vertices v and w in an undirected graph, while by (v, w) an arc from v to w in a digraph. In this paper, we say that a digraph \vec{G} is *connected* if \vec{G} is weakly connected, that is, the underlying graph G is connected. A vertex v in a digraph \vec{G} is called a *source* vertex if the in-degree of v is zero, while it is called a *sink* vertex if the out-degree of v is zero. A sequence $v_0 a_1 v_1 a_2 v_2 \dots a_l v_l$ of vertices v_0, v_1, \dots, v_l and arcs a_1, a_2, \dots, a_l in \vec{G} is called a *forward walk from v_0 on \vec{G}* if it forms a directed walk from v_0 to v_l (with repeated arcs and vertices allowed), that is, a_i is the arc from v_{i-1} to v_i for all $i \in \{1, 2, \dots, l\}$; while it is called a *backward walk to v_0 on \vec{G}* if it is a directed walk from v_l to v_0 , that is, a_i is the arc from v_i to v_{i-1} for all $i \in \{l, l-1, \dots, 1\}$.

3 Algorithms for Path Recolorability

In this section, we consider the case where R is a path. We first prove that the existence of an $(f_0 \rightarrow f_r)$ -reconfiguration sequence can be checked in linear time, as follows.

► **Theorem 2.** COLORING RECONFIGURATION UNDER RECOLORABILITY *for any graph G can be solved in $O(k + n + m)$ time if a recolorability graph R is a path.*

We prove Theorem 2 by giving such an algorithm. We first rename the colors in R so that the colors $1, 2, \dots, k$ appear in a numerical order along the path R , and modify two k -colorings f_0 and f_r accordingly; this can be done in $O(k + n)$ time. Then, the most important property for the path recolorability is that any recoloring step preserves the “order” of colors assigned to two adjacent vertices in G : If a k -coloring f of G assigns colors to two adjacent vertices $v, w \in V(G)$ such that $f(v) < f(w)$, then $f'(v) < f'(w)$ holds for any k -coloring f' such that an $(f \rightarrow f')$ -reconfiguration sequence exists. Indeed, this property yields the following necessary and sufficient condition, which can be checked in $O(m)$ time; and hence Theorem 2 holds.

► **Lemma 3 (*)**. *An $(f_0 \rightarrow f_r)$ -reconfiguration sequence exists on $\mathcal{C}_R(G)$ if and only if $f_r(v) < f_r(w)$ holds for any $vw \in E(G)$ such that $f_0(v) < f_0(w)$.*

We next give a linear-time algorithm to compute $\text{dist}(f_0, f_r)$; together with Theorem 2, this completes the proof of Theorem 1 for the path recolorability.

► **Theorem 4.** *Suppose that a recolorability graph R is a path, and let f_0 and f_r be two k -colorings of a graph G such that an $(f_0 \rightarrow f_r)$ -reconfiguration sequence exists on $\mathcal{C}_R(G)$. Then,*

- (a) $\text{dist}(f_0, f_r) = \sum_{v \in V(G)} |f_r(v) - f_0(v)|$;
- (b) $\text{dist}(f_0, f_r)$ can be computed in $O(k + n + m)$ time; and
- (c) a shortest $(f_0 \rightarrow f_r)$ -reconfiguration sequence can be output in $O(kn(n + m))$ time.

By Theorem 2 we can check in $O(k + n + m)$ time if an $(f_0 \rightarrow f_r)$ -reconfiguration sequence exists on $\mathcal{C}_R(G)$. Then, Theorem 4(b) immediately follows from Theorem 4(a). Therefore, we will prove Theorem 4(a) and (c), as follows: Observe that $\text{dist}(f_0, f_r) \geq \sum_{v \in V(G)} |f_r(v) - f_0(v)|$ holds, because each recoloring step can change the current color of a vertex $v \in V(G)$ to its adjacent color in R , and hence each vertex $v \in V(G)$ requires at least $|f_r(v) - f_0(v)|$ recoloring steps. Therefore, the following lemma completes the proof of Theorem 4.

► **Lemma 5 (*)**. *There exists an $(f_0 \rightarrow f_r)$ -reconfiguration sequence on $\mathcal{C}_R(G)$ of length $\sum_{v \in V(G)} |f_r(v) - f_0(v)|$. Furthermore, it can be output in $O(kn(n + m))$ time.*

4 Algorithm for Reachability on Cycle Recolorability

In this section, we consider the case where R is a cycle, and show that the existence of an $(f_0 \rightarrow f_r)$ -reconfiguration sequence can be checked in linear time; the shortest length will be discussed in the next section. We prove the following theorem in this section.

► **Theorem 6**. COLORING RECONFIGURATION UNDER RECOLORABILITY *for any graph G can be solved in $O(k + n + m)$ time if a recolorability graph R is a cycle.*

Since K_3 is a cycle, Theorem 6 immediately implies the following corollary.

► **Corollary 7** ([12]). COLORING RECONFIGURATION *with $k = 3$ can be solved in linear time.*

We will prove Theorem 6 by giving such an algorithm, as follows. In Section 4.1, we give a simple necessary condition for a yes-instance based on the concept of “frozen” vertices; the idea is simple, but we need a nice characterization of frozen vertices for checking the condition in linear time. In Section 4.2, we then give a necessary and sufficient condition for a yes-instance by defining a potential function which appropriately characterizes the reconfigurability of k -colorings; however, this condition cannot be checked in linear time by a naive way. In Section 4.3, we thus explain how to check the condition in linear time.

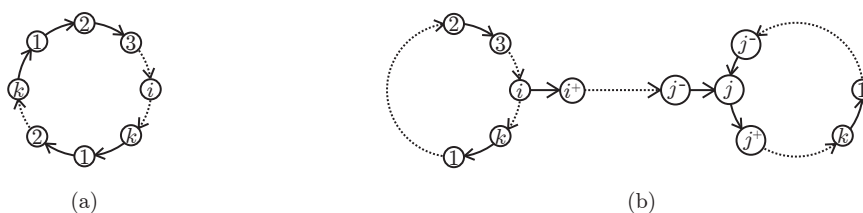
We rename the colors in R so that the colors $1, 2, \dots, k$ appear in a numerical order along the cycle R , and modify two k -colorings f_0 and f_r accordingly; this can be done in $O(k + n)$ time. For notational convenience, we define the *successor* color c^+ and the *predecessor* color c^- for a color $c \in V(R)$, as follows:

$$c^+ = \begin{cases} c + 1 & \text{if } c < k; \\ 1 & \text{if } c = k, \end{cases} \quad \text{and} \quad c^- = \begin{cases} c - 1 & \text{if } c > 1; \\ k & \text{if } c = 1. \end{cases}$$

We use this notation also for a color assigned by a k -coloring: For a k -coloring f of a graph G and a vertex v in G , we denote by $f(v)^+$ and $f(v)^-$ the successor and predecessor colors for $f(v)$, respectively. In this and later sections, we call a k -coloring of G simply a *coloring*.

4.1 Frozen vertices

We now define the concept of “frozen” vertices [7] from the viewpoint of recoloring, which plays an important role in our algorithm. For a coloring f of a graph G and a recolorability graph R on C , a vertex $v \in V(G)$ is said to be *frozen on f (under R)* if $f(v) = f'(v)$ holds for any coloring f' of G such that $\mathcal{C}_R(G)$ has an $(f \rightarrow f')$ -reconfiguration sequence. For a coloring f of G , we denote by $\text{Frozen}(f)$ the set of all vertices in G that are frozen on f . The following lemma gives a simple necessary condition, which immediately follows from the definition of frozen vertices.



■ **Figure 3** Characterization of frozen vertices.

► **Lemma 8.** *Suppose that there exists an $(f \rightarrow f')$ -reconfiguration sequence for two colorings f and f' of a graph G . Then, $\text{Frozen}(f) = \text{Frozen}(f')$, and $f(v) = f'(v)$ holds for every vertex v in $\text{Frozen}(f)$.*

Note that it is not trivial to compute $\text{Frozen}(f)$ for a coloring f in linear time. However, we will give a characterization of frozen vertices (in Lemma 9), which enables us to compute all of them in linear time (as proved in Lemma 10). We note that Lemma 9 generalizes the characterization of frozen vertices on COLORING RECONFIGURATION with $k = 3$ given by Cereceda et al. [7].

To characterize the frozen vertices, we introduce some notation and terms. For a graph G and its coloring f , let \vec{H}_f be the digraph with vertex set $V(\vec{H}_f) = V(G)$ and arc set

$$A(\vec{H}_f) = \{(v, w) : vw \in E(G) \text{ and } f(v)^+ = f(w)\}.$$

Notice that an arc $(v, w) \in A(\vec{H}_f)$ implies that $f(v) = f(w)^-$, and represents that, if we wish to recolor v from $f(v)$ to $f(v)^+$, we need to recolor w from $f(w)$ ($= f(v)^+$) to $f(w)^+$ in advance. The *forward blocking graph from v on a coloring f* , denoted by $\vec{B}^+(v, f)$, is the subgraph of \vec{H}_f consisting of all forward walks from v on \vec{H}_f . Similarly, the *backward blocking graph to v on a coloring f* , denoted by $\vec{B}^-(v, f)$, is the subgraph of \vec{H}_f consisting of all backward walks to v on \vec{H}_f . Then, we have the following lemma. (See also Figure 3.)

► **Lemma 9 (*)**. *A vertex $v \in V(G)$ is frozen on f if and only if it satisfies the following conditions (a) or (b):*

- (a) *v is contained in a directed cycle in \vec{H}_f ; or*
- (b) *\vec{H}_f has a forward walk from v to a vertex in a directed cycle, and also has a backward walk from a vertex in a directed cycle to v .*

Based on Lemma 9, we have the following lemma.

► **Lemma 10 (*)**. *$\text{Frozen}(f)$ can be computed in $O(m)$ time for any coloring f of a graph G .*

4.2 Necessary and sufficient condition

In the remainder of this section, by Lemma 8 we assume that $\text{Frozen}(f_0) = \text{Frozen}(f_r)$ and $f_0(v) = f_r(v)$ for each vertex $v \in \text{Frozen}(f_0)$; otherwise it is a no-instance. In this subsection, we will give a necessary and sufficient condition for a yes-instance.

We define some notation to describe the condition. Let G be an undirected graph, and let \vec{H} be any digraph whose underlying graph is a subgraph of G . For a coloring f of G and each arc $(u, v) \in A(\vec{H})$, we define the *potential* $p_f((u, v))$ of (u, v) on f , as follows:

$$p_f((u, v)) = \begin{cases} f(v) - f(u) & \text{if } f(v) > f(u); \\ f(v) - f(u) + k & \text{if } f(v) < f(u). \end{cases}$$

Note that $f(u) \neq f(v)$ holds since $uv \in E(G)$. In addition, observe that

$$\mathfrak{p}_f((u, v)) + \mathfrak{p}_f((v, u)) = k \quad (1)$$

holds for any pair of parallel arcs (u, v) and (v, u) if such a pair exists. The *potential* $\mathfrak{p}_f(\vec{H})$ of \vec{H} on f is defined to be the sum of potentials of all arcs of \vec{H} on f , that is, $\mathfrak{p}_f(\vec{H}) = \sum_{(u,v) \in A(\vec{H})} \mathfrak{p}_f((u, v))$.

Let C be a cycle in an undirected graph G . Then, there are only two possible orientations of C such that they form directed cycles, that is, either the clockwise direction or the anticlockwise direction; we always denote by \vec{C} and \overleftarrow{C} such the two possible orientations of C . The following lemma immediately follows from Eq. (1).

► **Lemma 11.** *Let f be a coloring of an undirected graph G . Then, $\mathfrak{p}_f(\vec{C}) + \mathfrak{p}_f(\overleftarrow{C}) = k|E(C)|$ for every cycle C in G .*

For a coloring f of an undirected graph G , we define a supergraph G^f of G as follows⁴: let $V(G^f) = V(G)$, and we arbitrarily add new edges between frozen vertices on G so that $\text{Frozen}(f)$ induces a connected subgraph in the resulting graph. Then, since there are at most $|V(G)|$ frozen vertices, G^f has $|V(G)|$ vertices and at most $|E(G)| + |V(G)| - 1$ edges. Note that $G^f = G$ if $\text{Frozen}(f) = \emptyset$. Recall that two given colorings f_0 and f_r of G are assumed to satisfy $\text{Frozen}(f_0) = \text{Frozen}(f_r)$ and $f_0(v) = f_r(v)$ for every vertex v in $\text{Frozen}(f_0)$. We can thus assume $G^{f_0} = G^{f_r}$, and hence simply denote it by G^f . Furthermore, since newly added edges join only frozen vertices, we have the following lemma.

► **Lemma 12.** *There exists an $(f_0 \rightarrow f_r)$ -reconfiguration sequence on $\mathcal{C}_R(G)$ if and only if there exists an $(f_0 \rightarrow f_r)$ -reconfiguration sequence on $\mathcal{C}_R(G^f)$.*

We are now ready to claim our necessary and sufficient condition, as follows.

► **Theorem 13.** *Let f_0 and f_r be two colorings of a graph G such that $\text{Frozen}(f_0) = \text{Frozen}(f_r)$, and $f_0(v) = f_r(v)$ for all vertices $v \in \text{Frozen}(f_0)$. Then, an $(f_0 \rightarrow f_r)$ -reconfiguration sequence exists on $\mathcal{C}_R(G)$ if and only if $\mathfrak{p}_{f_0}(\vec{C}) = \mathfrak{p}_{f_r}(\vec{C})$ holds for every cycle C in G^f .*

Before proving the theorem, we note that Theorem 13 is independent from the choice of the two orientations of a cycle C , because Lemma 11 implies that $\mathfrak{p}_{f_0}(\vec{C}) = \mathfrak{p}_{f_r}(\vec{C})$ holds if and only if $\mathfrak{p}_{f_0}(\overleftarrow{C}) = \mathfrak{p}_{f_r}(\overleftarrow{C})$ holds. We also note that Theorem 13 does not directly yield a linear-time algorithm.

We first prove the only-if direction of Theorem 13. Suppose that there exists an $(f_0 \rightarrow f_r)$ -reconfiguration sequence on $\mathcal{C}_R(G)$. Then, Lemma 12 implies that $\mathcal{C}_R(G^f)$ contains an $(f_0 \rightarrow f_r)$ -reconfiguration sequence $\langle f_0, f_1, \dots, f_\ell \rangle$, where $f_\ell = f_r$, and hence the only-if direction of Theorem 13 can be obtained from the following lemma.

► **Lemma 14 (*)**. *Suppose that two colorings f and f' are adjacent on $\mathcal{C}_R(G^f)$. Then, $\mathfrak{p}_f(\vec{C}) = \mathfrak{p}_{f'}(\vec{C})$ holds for every cycle C in G^f .*

We then prove the if direction of Theorem 13: If $\mathfrak{p}_{f_0}(\vec{C}) = \mathfrak{p}_{f_r}(\vec{C})$ holds for every cycle C in G^f , then an $(f_0 \rightarrow f_r)$ -reconfiguration sequence exists on $\mathcal{C}_R(G^f)$; Lemma 12 then implies that $\mathcal{C}_R(G)$ contains an $(f_0 \rightarrow f_r)$ -reconfiguration sequence.

⁴ We note that our construction of G^f is different from that by Cereceda et al. [7] so that the running time of our algorithm does not depend on k .

Our proof is constructive, that is, we give an algorithm which indeed finds an $(f_0 \rightarrow f_r)$ -reconfiguration sequence on $\mathcal{C}_R(G^f)$. We say that a vertex v is *fixed* if it is colored with $f_r(v)$ and our algorithm decides not to recolor v anymore. Thus, all frozen vertices are fixed. Our algorithm maintains the set of fixed vertices, denoted by F . The following Algorithm 1 transforms f_0 into a coloring f'_0 of G^f so that $F \neq \emptyset$, as the initialization.

Algorithm 1 Initialization for Algorithm 2.

1. If $\text{Frozen}(f_0) \neq \emptyset$, then let $F = \text{Frozen}(f_0)$ and $f'_0 = f_0$.
 2. Otherwise let $F = \{v\}$ for an arbitrarily chosen vertex $v \in V(G)$. Let $f = f_0$, and obtain f'_0 such that $f'_0(v) = f_r(v)$, as follows:
 - 2-1. If $f(v) = f_r(v)$, then let $f'_0 = f$ and stop the algorithm.
 - 2-2. Otherwise recolor a sink vertex w (possibly v itself) of $\vec{B}^+(v, f)$ to $f(w)^+$. Let f be the resulting coloring, and go to Step 2-1.
-

Note that we can always find a sink vertex w in Step 2-2 of Algorithm 1, because otherwise $\vec{B}^+(v, f)$ contains a directed cycle; by Lemma 9 the vertices in the directed cycle are frozen, and hence this contradicts the assumption that $\text{Frozen}(f_0) = \emptyset$ holds in Step 2. We note the following properties.

► **Lemma 15.** *Let $F \subseteq V(G^f)$ be the vertex subset obtained by Algorithm 1, and let f'_0 be the coloring of G^f obtained by Algorithm 1. Then, the induced subgraph $G^f[F]$ is connected, and $\mathfrak{p}_{f'_0}(\vec{C}) = \mathfrak{p}_{f_0}(\vec{C}) = \mathfrak{p}_{f_r}(\vec{C})$ for any cycle C in G^f .*

Proof. Recall that G^f was obtained by adding new edges to G so that $G^f[\text{Frozen}(f_0)]$ is connected. Thus, $G^f[F] = G^f[\text{Frozen}(f_0)]$ is connected if $\text{Frozen}(f_0) \neq \emptyset$. If $\text{Frozen}(f_0) = \emptyset$, then F consists of a single vertex v ; and hence $G^f[F]$ is connected also in this case.

Notice that Algorithm 1 constructs an $(f_0 \rightarrow f'_0)$ -reconfiguration sequence on $\mathcal{C}_R(G^f)$. Then, Lemma 14 implies that $\mathfrak{p}_{f'_0}(\vec{C}) = \mathfrak{p}_{f_0}(\vec{C}) = \mathfrak{p}_{f_r}(\vec{C})$ for any cycle C in G^f . ◀

We now give our main procedure, called Algorithm 2, which finds an $(f'_0 \rightarrow f_r)$ -reconfiguration sequence on $\mathcal{C}_R(G^f)$. The algorithm attempts to extend the vertex set F to $V(G^f)$ so that each vertex v in F is fixed (and hence is colored with $f_r(v)$); we eventually obtain the target coloring f_r when $F = V(G^f)$. Recall that our algorithm never recolors any vertex v in F , and all frozen vertices are contained in F . Let $f = f'_0$, and apply the following procedure.

Algorithm 2 Finding an $(f'_0 \rightarrow f_r)$ -reconfiguration sequence on $\mathcal{C}_R(G^f)$.

1. If $F = V(G^f)$ holds, then stop the algorithm.
 2. Otherwise pick an arbitrary vertex $v \in V(G^f) \setminus F$ which is adjacent with at least one vertex $u \in F$.
 - 2-1. If $f(v) = f_r(v)$, then add v to F and go to Step 1.
 - 2-2. Otherwise
 - if $\mathfrak{p}_f((u, v)) < \mathfrak{p}_{f_r}((u, v))$, then recolor a sink vertex w (possibly v itself) of $\vec{B}^+(v, f)$ to $f(w)^+$; and
 - if $\mathfrak{p}_f((u, v)) > \mathfrak{p}_{f_r}((u, v))$, then recolor a source vertex w (possibly v itself) of $\vec{B}^-(v, f)$ to $f(w)^-$.
 Let f be the resulting coloring, and go to Step 2-1.
-

To prove that Algorithm 2 correctly finds an $(f'_0 \rightarrow f_r)$ -reconfiguration sequence on $\mathcal{C}_R(G^f)$, it suffices to show that there always exists a non-fixed sink/source vertex in Step 2-2 under the condition that $\mathbf{p}_{f'_0}(\vec{C}) = \mathbf{p}_{f_0}(\vec{C}) = \mathbf{p}_{f_r}(\vec{C})$ holds for any cycle C in G^f . Therefore, the following lemma completes the proof of the if direction of Theorem 13.

► **Lemma 16 (*)**. *Every application of Step 2 of Algorithm 2 produces a set F of fixed vertices and a coloring f of G^f satisfying the following (a) and (b): For each edge uv in G^f such that $u \in F$ and $v \notin F$,*

- (a) *if $\mathbf{p}_f((u, v)) < \mathbf{p}_{f_r}((u, v))$, then $\vec{B}^+(v, f)$ is a directed acyclic graph such that no vertex in $\vec{B}^+(v, f)$ is contained in F ; and*
- (b) *if $\mathbf{p}_f((u, v)) > \mathbf{p}_{f_r}((u, v))$, then $\vec{B}^-(v, f)$ is a directed acyclic graph such that no vertex in $\vec{B}^-(v, f)$ is contained in F .*

4.3 Proof of Theorem 6

We finally prove Theorem 6 by giving such an algorithm. Our algorithm first checks the simple necessary condition described in Lemma 8. By Lemma 10 this step can be done in $O(m)$ time. Note that we can obtain the vertex subsets $\text{Frozen}(f_0)$ and $\text{Frozen}(f_r)$ in this running time. Then, we determine whether a given instance is a yes-instance or not, based on the necessary and sufficient condition described in Theorem 13. However, recall that the condition in Theorem 13 cannot be checked in linear time by a naive way. Below, we give a linear-time algorithm to check the condition.

Let T be an arbitrary spanning tree of the graph G^f . For an edge $e \in E(G^f) \setminus E(T)$, we denote by $C_{T,e}$ the unique cycle obtained by adding the edge e to T . The following lemma shows that it suffices to check the necessary and sufficient condition only for the number $|E(G^f) \setminus E(T)|$ of cycles.

► **Lemma 17 (*)**. *Let T be any spanning tree of G^f . Then, $\mathbf{p}_{f_0}(\vec{C}) = \mathbf{p}_{f_r}(\vec{C})$ holds for every cycle C of G^f if and only if $\mathbf{p}_{f_0}(\vec{C}_{T,e}) = \mathbf{p}_{f_r}(\vec{C}_{T,e})$ holds for every edge $e \in E(G^f) \setminus E(T)$.*

Lemma 17 and the following lemma imply that there is a linear-time algorithm to check the necessary and sufficient condition described in Theorem 13. Therefore, the following lemma completes the proof of Theorem 6.

► **Lemma 18 (*)**. *Let T be any spanning tree of G^f . Then, $\mathbf{p}_{f_0}(\vec{C}_{T,e})$ and $\mathbf{p}_{f_r}(\vec{C}_{T,e})$ for all $e \in E(G^f) \setminus E(T)$ can be computed in $O(n + m)$ time in total.*

5 Algorithm for Shortest Sequence on Cycle Recolorability

In this section, we consider the case where R is a cycle, and explain how to compute the length of a shortest reconfiguration sequence.

Let $P_{u,v}$ be a path in an undirected graph G connecting vertices u and v . We denote by $\vec{P}_{u,v}$ the directed path from u to v . The following theorem characterizes the shortest length of an $(f_0 \rightarrow f_r)$ -reconfiguration sequence, which generalizes the characterization for COLORING RECONFIGURATION with $k = 3$ [7, 12].

► **Theorem 19 (*)**. *Suppose that a recolorability graph R is a cycle, and let f_0 and f_r be two colorings of a graph G such that an $(f_0 \rightarrow f_r)$ -reconfiguration sequence exists on $\mathcal{C}_R(G)$. Then, the following (a) and (b) hold:*

(a) If $\text{Frozen}(f_0) \neq \emptyset$, then it holds for an arbitrary chosen vertex $u \in \text{Frozen}(f_0)$ that

$$\text{dist}(f_0, f_r) = \sum_{v \in V(G)} |\mathfrak{p}_{f_r}(\vec{P}_{u,v}) - \mathfrak{p}_{f_0}(\vec{P}_{u,v})|,$$

where $P_{u,v}$ is an arbitrary chosen path in G connecting u and v .

(b) If $\text{Frozen}(f_0) = \emptyset$, then there exist two integers $\rho_{u,1}$ and $\rho_{u,2}$ for an arbitrary chosen vertex $u \in V(G)$ such that

$$\text{dist}(f_0, f_r) = \min \left\{ \sum_{v \in V(G)} |\mathfrak{p}_{f_r}(\vec{P}_{u,v}) - \mathfrak{p}_{f_0}(\vec{P}_{u,v}) + \rho_{u,1}|, \sum_{v \in V(G)} |\mathfrak{p}_{f_r}(\vec{P}_{u,v}) - \mathfrak{p}_{f_0}(\vec{P}_{u,v}) + \rho_{u,2}| \right\},$$

where $P_{u,v}$ is an arbitrary chosen path in G connecting u and v .

We finally claim that $\text{dist}(f_0, f_r)$ can be computed in linear time, based on Theorem 19, and that a shortest $(f_0 \rightarrow f_r)$ -reconfiguration sequence can be output in polynomial time.

► **Lemma 20 (*)**. For any vertex $u \in V(G)$, two integers $\rho_{u,1}$ and $\rho_{u,2}$ of Theorem 19(b) can be obtained in $O(n + m)$ time. Furthermore,

(a) $\text{dist}(f_0, f_r)$ can be computed in $O(n + m)$ time; and

(b) a shortest $(f_0 \rightarrow f_r)$ -reconfiguration sequence can be output in $O(kn(n + m))$ time.

6 Concluding Remarks

In this paper, we have generalized and sharpened the positive results [7, 12] obtained for COLORING RECONFIGURATION, from the viewpoint of recolorability constraints. We emphasize that our algorithms run in linear time to simply answer the decision problem COLORING RECONFIGURATION UNDER RECOLORABILITY, or to compute the shortest length of $(f_0 \rightarrow f_r)$ -reconfiguration sequences.

One may expect that a shortest $(f_0 \rightarrow f_r)$ -reconfiguration sequence can be output also in linear time. However, Cereceda et al. [7] showed that there exists an infinite family of yes-instances for COLORING RECONFIGURATION with $k = 3$ whose shortest $(f_0 \rightarrow f_r)$ -reconfiguration sequence requires $\Omega(n^2)$ length.

Together with our sister paper [14], we have clarified several tractable/intractable cases of COLORING RECONFIGURATION UNDER RECOLORABILITY. Our analyses are summarized in Table 1, and give a better understanding of the complexity of COLORING RECONFIGURATION. However, the complexity status remains open for the case where a connected recolorability graph R is of maximum degree three and has at most one cycle.

■ **Table 1** Complexity of COLORING RECONFIGURATION UNDER RECOLORABILITY, where a recolorability graph R is assumed to be connected without loss of generality.

Maximum degree of R	R contains at most one cycle	R contains at least two cycles
two	Linear time [this paper]	(no such R exists)
three	?	PSPACE-complete [14]
at least four	PSPACE-complete [14]	PSPACE-complete [14]

References

- 1 Marthe Bonamy and Nicolas Bousquet. Recoloring graphs via tree decompositions. *Eur. J. Comb.*, 69:200–213, 2018. doi:10.1016/j.ejc.2017.10.010.
- 2 Marthe Bonamy, Matthew Johnson, Ioannis Lignos, Viresh Patel, and Daniël Paulusma. Reconfiguration graphs for vertex colourings of chordal and chordal bipartite graphs. *J. Comb. Optim.*, 27(1):132–143, 2014. doi:10.1007/s10878-012-9490-y.
- 3 Paul S. Bonsma and Luis Cereceda. Finding Paths between graph colourings: PSPACE-completeness and superpolynomial distances. *Theor. Comput. Sci.*, 410(50):5215–5226, 2009. doi:10.1016/j.tcs.2009.08.023.
- 4 Paul S. Bonsma, Amer E. Mouawad, Naomi Nishimura, and Venkatesh Raman. The Complexity of Bounded Length Graph Recoloring and CSP Reconfiguration. In Marek Cygan and Pinar Heggernes, editors, *Parameterized and Exact Computation - 9th International Symposium, IPEC 2014, Wroclaw, Poland, September 10-12, 2014. Revised Selected Papers*, volume 8894 of *Lecture Notes in Computer Science*, pages 110–121. Springer, 2014. doi:10.1007/978-3-319-13524-3_10.
- 5 Paul S. Bonsma and Daniël Paulusma. Using Contracted Solution Graphs for Solving Reconfiguration Problems. In Piotr Faliszewski, Anca Muscholl, and Rolf Niedermeier, editors, *41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016, August 22-26, 2016 - Kraków, Poland*, volume 58 of *LIPICs*, pages 20:1–20:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPICs.MFCS.2016.20.
- 6 Richard C. Brewster, Sean McGuinness, Benjamin Moore, and Jonathan A. Noel. A dichotomy theorem for circular colouring reconfiguration. *Theor. Comput. Sci.*, 639:1–13, 2016. doi:10.1016/j.tcs.2016.05.015.
- 7 Luis Cereceda, Jan van den Heuvel, and Matthew Johnson. Finding paths between 3-colorings. *Journal of Graph Theory*, 67(1):69–82, 2011. doi:10.1002/jgt.20514.
- 8 Tatsuhiko Hatanaka, Takehiro Ito, and Xiao Zhou. The List Coloring Reconfiguration Problem for Bounded Pathwidth Graphs. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 98-A(6):1168–1178, 2015. URL: http://search.ieice.org/bin/summary.php?id=e98-a_6_1168, doi:10.1587/transfun.E98.A.1168.
- 9 Tatsuhiko Hatanaka, Takehiro Ito, and Xiao Zhou. Parameterized complexity of the list coloring reconfiguration problem with graph parameters. *Theor. Comput. Sci.*, 739:65–79, 2018. doi:10.1016/j.tcs.2018.05.005.
- 10 Jan van den Heuvel. The complexity of change. In Simon R. Blackburn, Stefanie Gerke, and Mark Wildon, editors, *Surveys in Combinatorics 2013*, volume 409 of *London Mathematical Society Lecture Note Series*, pages 127–160. Cambridge University Press, 2013. doi:10.1017/CB09781139506748.005.
- 11 Takehiro Ito, Erik D. Demaine, Nicholas J. A. Harvey, Christos H. Papadimitriou, Martha Sideri, Ryuhei Uehara, and Yushi Uno. On the complexity of reconfiguration problems. *Theor. Comput. Sci.*, 412(12–14):1054–1065, 2011. doi:10.1016/j.tcs.2010.12.005.
- 12 Matthew Johnson, Dieter Kratsch, Stefan Kratsch, Viresh Patel, and Daniël Paulusma. Finding Shortest Paths Between Graph Colourings. *Algorithmica*, 75(2):295–321, 2016. doi:10.1007/s00453-015-0009-7.
- 13 Naomi Nishimura. Introduction to Reconfiguration. *Algorithms*, 11(4):52, 2018. doi:10.3390/a11040052.
- 14 Hiroki Osawa, Akira Suzuki, Takehiro Ito, and Xiao Zhou. Complexity of Coloring Reconfiguration under Recolorability Constraints. In Yoshio Okamoto and Takeshi Tokuyama, editors, *28th International Symposium on Algorithms and Computation, ISAAC 2017, December 9-12, 2017, Phuket, Thailand*, volume 92 of *LIPICs*, pages 62:1–62:12. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPICs.ISAAC.2017.62.

- 15 Marcin Wrochna. Homomorphism Reconfiguration via Homotopy. In Ernst W. Mayr and Nicolas Ollinger, editors, *32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015, March 4-7, 2015, Garching, Germany*, volume 30 of *LIPICs*, pages 730–742. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPICs.STACS.2015.730.
- 16 Marcin Wrochna. Reconfiguration in bounded bandwidth and tree-depth. *J. Comput. Syst. Sci.*, 93:1–10, 2018. doi:10.1016/j.jcss.2017.11.003.