FPT algorithms for Token Jumping on Graphs

Takehiro Ito[1,a)] Marcin Kamiński[2,b)] Hirotaka Ono[3,c)] Akira Suzuki[1,d)] Ryuhei Uehara[4,e)]
Katsuhisa Yamanaka[5,f)]

Abstract: Suppose that we are given two independent sets I_0 and I_r of a graph such that $|I_0| = |I_r|$, and imagine that a token is placed on each vertex in I_0. Then, the TOKEN JUMPING problem is to determine whether there exists a sequence of independent sets which transforms I_0 into I_r so that each independent set in the sequence results from the previous one by moving exactly one token to another vertex. Therefore, all independent sets in the sequence must be of the same cardinality. This problem is W[1]-hard when parameterized only by the number of tokens. In this paper, we give FPT algorithms for general graphs when parameterized by both the number of tokens and the maximum degree.

1. Introduction

The TOKEN JUMPING problem was introduced by Kamiński *et al.* [12], which can be seen as a "dynamic" version of independent sets in a graph. Recall that an *independent set* of a graph G is a vertex-subset of G in which no two vertices are adjacent. (See Fig. 1 which depicts six different independent sets of the same graph.) Suppose that we are given two independent sets I_0 and I_r of a graph $G = (V, E)$ such that $|I_0| = |I_r|$, and imagine that a token (coin) is placed on each vertex in I_0. Then, the TOKEN JUMPING problem is to determine whether there exists a sequence $\langle I_0, I_1, \ldots, I_\ell \rangle$ of independent sets of G such that

(a) $I_\ell = I_r$, and $|I_i| = |I_0| = |I_r|$ for all i, $1 \le i \le \ell$; and
(b) for each index i, $1 \le i \le \ell$, I_i can be obtained from I_{i-1} by moving exactly one token on a vertex $u \in I_{i-1}$ to another vertex $v \in V \setminus I_{i-1}$, and hence $I_{i-1} \setminus I_i = \{u\}$ and $I_i \setminus I_{i-1} = \{v\}$.

Figure 1 illustrates a sequence $\langle I_0, I_1, \ldots, I_5 \rangle$ of independent sets which transforms I_0 into $I_r = I_5$.

Recently, this type of problems have been studied extensively in the framework of *reconfiguration problems* [7], which arise when we wish to find a step-by-step transformation between two feasible solutions of a problem such that all intermediate solutions are also feasible and each step abides by a prescribed reconfiguration rule (*i.e.*, an adjacency relation defined on feasible solutions of the original problem). For example, the TOKEN JUMPING problem can be seen as a reconfiguration problem for the (ordinary) INDEPENDENT SET problem: feasible solutions are defined to be all independent sets of the same cardinality in a graph; and the reconfiguration rule is defined to be the condition (b) above. This reconfiguration framework has been applied to several well-known problems, including INDEPENDENT SET [5], [6], [7], [12], [14], SATISFIABILITY [4], [13], SET COVER, CLIQUE, MATCHING [7], VERTEX-COLORING [1], [2], [3], LIST EDGE-COLORING [8], [10], LIST $L(2,1)$-LABELING [9], SHORTEST PATH [11], etc.

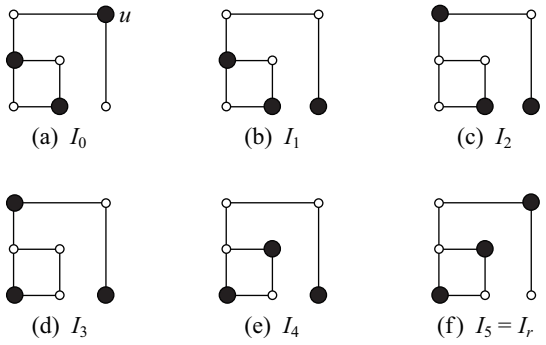

Fig. 1 A sequence $\langle I_0, I_1, \ldots, I_5 \rangle$ of independent sets of the same graph, where the vertices in independent sets are depicted by large black circles (tokens).

1.1 Reconfiguration rules and related results

The original reconfiguration problem for INDEPENDENT SET was introduced by Hearn and Demaine [5], which employs another reconfiguration rule. Indeed, there are three reconfiguration problems for INDEPENDENT SET (ISReconf, for short) under different reconfiguration rules, as follows.

- **Token Sliding (TS)** [2], [5], [6], [12]: We can slide a single token only *along an edge* of a graph. In other words, each token can be moved only to its adjacent vertex. This

1 Graduate School of Information Sciences, Tohoku University, Aobayama 6-6-05, Sendai, 980-8579, Japan.
2 Dept. of Mathematics, Computer Science and Mechanics, University of Warsaw, Banacha 2, 02-097, Warsaw, Poland.
3 Faculty of Economics, Kyushu University, Hakozaki 6-19-1, Higashi-ku, Fukuoka, 812-8581, Japan.
4 School of Information Science, JAIST, Asahidai 1-1, Nomi, Ishikawa, 923-1292, Japan.
5 Dept. of Electrical Engineering and Computer Science, Iwate University, Ueda 4-3-5, Morioka, Iwate 020-8551, Japan.
a) takehiro@ecei.tohoku.ac.jp
b) mjk@mimuw.edu.pl
c) hirotaka@en.kyushu-u.ac.jp
d) a.suzuki@ecei.tohoku.ac.jp
e) uehara@jaist.ac.jp
f) yamanaka@cis.iwate-u.ac.jp

rule corresponds to the original one introduced by Hearn and Demaine [5].

- **Token Jumping (TJ)** [12]: This rule corresponds to TOKEN JUMPING, that is, we can move a single token to any vertex.
- **Token Addition and Removal (TAR)** [7], [12], [14]: We can either add or remove a single token at a time if it results in an independent set of cardinality at least a given threshold. Therefore, independent sets in the sequence do not have the same cardinality.

We remark that the existence of a desired sequence depends deeply on the reconfiguration rules. For example, Fig. 1 is an yes-instance for TOKEN JUMPING, but it is a no-instance for ISReconf under the TS rule.

We here explain only the results which are strongly related to TOKEN JUMPING; see the references above for the other results.

Hearn and Demaine [5], [6] proved that ISReconf under the TS rule is PSPACE-complete for planar graphs of maximum degree three. Then, Bonsma and Cereceda [2] showed that this problem remains PSPACE-complete even for very restricted instances. Indeed, their result implies that TOKEN JUMPING is PSPACE-complete for planar graphs with maximum degree three.

Kamiński *et al.* [12] proved that ISReconf is PSPACE-complete for perfect graphs under any of the three reconfiguration rules. As the positive results for TOKEN JUMPING, they gave a linear-time algorithm for even-hole-free graphs. Furthermore, their algorithm can find an actual sequence of independent sets with the minimum number of token movements.

1.2 Our contributions

In this paper, we investigate the parameterized complexity of the TOKEN JUMPING problem. The problem is W[1]-hard when parameterized only by the number t of tokens. (Details are omitted from this extended abstract.) Therefore, the problem admits no FPT algorithm when parameterized only by t unless FPT = W[1].

We thus consider the problem with two parameters, and give an FPT algorithm for general graphs when parameterized by both the number of tokens and the maximum degree. Recall that the problem remains PSPACE-complete even if the maximum degree is three. Therefore, it is very unlikely that the problem can be solved in polynomial time even for graphs with bounded maximum degree.

In addition, we show that our FPT algorithm for general graphs can be modified so that it finds an actual sequence of independent sets between I_0 and I_r with the minimum number of token movements. We remark that the sequence of independent sets in Fig. 1 has the minimum length. It is interesting that the token on the vertex u in Fig. 1(a) must be moved twice even though $u \in I_0 \cap I_r$.

2. Preliminaries

In this section, we first introduce some basic terms and notations which will be used throughout the paper.

In TOKEN JUMPING, we may assume without loss of generality that graphs are simple. For a graph G, we sometimes denote by $V(G)$ and $E(G)$ the vertex set and the edge set of G, respectively. Let $n(G) = |V(G)|$ and $m(G) = |E(G)|$. We denote by $\Delta(G)$ the maximum degree of G.

For a vertex v of a graph G, we denote by $N(G; v)$ the set of all neighbors of v in G (which does not include v itself), that is, $N(G; v) = \{w \in V(G) \mid (v, w) \in E(G)\}$. Let $N[G; v] = N(G; v) \cup \{v\}$, and let $N[G; V'] = \bigcup_{v \in V'} N[G; v]$ for a vertex-subset $V' \subseteq V(G)$.

Let I_i and I_j be two independent sets of the same cardinality in a graph $G = (V, E)$. We say that I_i and I_j are *adjacent* if there exists exactly one pair of vertices u and v such that $I_i \setminus I_j = \{u\}$ and $I_j \setminus I_i = \{v\}$, that is I_j can be obtained from I_i by *moving* the token on a vertex $u \in I_i$ to another vertex $v \in V \setminus I_i$. We remark that the tokens are unlabeled, while the vertices in a graph are labeled.

A *reconfiguration sequence* between two independent sets I and I' of G is a sequence $\langle I_1, I_2, \ldots, I_\ell \rangle$ of independent sets of G such that $I_1 = I$, $I_\ell = I'$, and I_{i-1} and I_i are adjacent for $i = 2, 3, \ldots, \ell$. We say that two independent sets I and I' are *reconfigurable* each other if there exists a reconfiguration sequence between I and I'. Clearly, any two adjacent independent sets are reconfigurable each other. The *length* of a reconfiguration sequence $\mathcal{S}$ is defined as the number of independent sets contained in $\mathcal{S}$. For example, the length of the reconfiguration sequence in Fig. 1 is 6.

The TOKEN JUMPING problem is to determine whether two given independent sets I_0 and I_r of a graph G are reconfigurable each other. We may assume without loss of generality that $|I_0| = |I_r|$; otherwise the answer is clearly "no." Note that TOKEN JUMPING is a decision problem asking the existence of a reconfiguration sequence between I_0 and I_r, and hence it does not ask an actual reconfiguration sequence. We always denote by I_0 and I_r the *initial* and *target* independent sets of G, respectively, as an instance of TOKEN JUMPING.

3. FPT algorithms

In this section, we give an FPT algorithm for general graphs when parameterized by both the number of tokens and the maximum degree. Recall that TOKEN JUMPING remains PSPACE-complete even for planar graphs with bounded maximum degree.

In Section 3.1, we first give an FPT algorithm which simply solves TOKEN JUMPING for general graphs. We then show in Section 3.2 that our FPT algorithm can be modified so that it finds an actual reconfiguration sequence with the minimum length.

3.1 TOKEN JUMPING

The main result of this subsection is the following theorem.

Theorem 3.1 Let G be a graph whose maximum degree is bounded by a fixed constant d. Let I_0 and I_r be two independent sets of G such that $|I_0| = |I_r| \leq t$ for a fixed constant t. Then, one can determine whether I_0 and I_r are reconfigurable each other in time $O\big((3td)^{2t}\big)$.

In this subsection, we give such an algorithm as a proof of Theorem 3.1. We first show in Lemma 3.2 that, if a graph G has at least $3t(d + 1)$ vertices, then I_0 and I_r are always reconfigurable each other. Therefore, one can know that the answer is always "yes" if $n(G) \geq 3t(d + 1)$, and hence it suffices to deal with a graph having less than $3t(d + 1)$ vertices. For such a graph, we then show in Lemma 3.3 that there is an $O\big((3td)^{2t}\big)$-time algo-

rithm that determines whether I_0 and I_r are reconfigurable each other.

We first show that any two independent sets are reconfigurable each other if the graph has a sufficiently large number of vertices, as in the following lemma.

Lemma 3.2 Let G be a graph with $\Delta(G) \leq d$, and let I_i and I_j be an arbitrary pair of independent sets of G such that $|I_i| = |I_j| \leq t$. Then, I_i and I_j are reconfigurable each other if $n(G) \geq 3t(d+1)$.

Proof. Suppose that $n(G) \geq 3t(d+1)$. To prove the lemma, we show that there exists a reconfiguration sequence between I_i and I_j.

Let G^- be the graph obtained from G by deleting all vertices in $N[G; I_i] \cup N[G; I_j]$. Since all neighbors of the vertices in $I_i \cup I_j$ have been deleted from G, no vertex in G^- is adjacent with any vertex in $I_i \cup I_j$. Therefore, if G^- has an independent set I_k with $|I_k| \geq t$, then there is a reconfiguration sequence between I_i and I_j, as follows: move all tokens on the vertices in I_i to the vertices in I_k one by one; and move all tokens on the vertices in I_k to the vertices in I_j one by one.

To complete the proof, we thus show that G^- has an independent set I_k with $|I_k| \geq t$ if $n(G) \geq 3t(d+1)$. Since $\Delta(G) \leq d$, we clearly have $\left|N[G; v]\right| \leq d+1$ for every vertex v in G. Since $|I_i| \leq t$, we thus have

$$\left|N[G; I_i]\right| \leq \sum_{v \in I_i} \left|N[G; v]\right| \leq t(d+1).$$

Similarly, we have $\left|N[G; I_j]\right| \leq t(d+1)$. Therefore,

$$n(G^-) \geq n(G) - \left|N[G; I_i]\right| - \left|N[G; I_j]\right| \geq t(d+1). \tag{1}$$

We now suppose for a contradiction that $|I_{\max}| < t$ holds for a maximum independent set $I_{\max}$ of G^-. Then, we have

$$\left|N[G^-; I_{\max}]\right| \leq \sum_{v \in I_{\max}} \left|N[G; v]\right| < t(d+1),$$

and hence by Eq. (1)

$$n(G^-) - \left|N[G^-; I_{\max}]\right| \geq 1.$$

Therefore, the graph obtained from G^- by deleting all vertices in $N[G^-; I_{\max}]$ is non-empty, and hence we can add at least one vertex to $I_{\max}$. This contradicts the assumption that $I_{\max}$ is a maximum independent set of G^-. Therefore, $|I_{\max}| \geq t$, and hence G^- has an independent set I_k with $|I_k| \geq t$. □

We then give an FPT algorithm for the case where a given graph G has only a constant number of vertices, as in the following lemma.

Lemma 3.3 Suppose that $n(G) < 3t(d+1)$. Then, there is an $O\left((3td)^{2t}\right)$-time algorithm which determines whether I_0 and I_r are reconfigurable each other.

Proof. We give such an algorithm. For a graph G and a constant $t' = |I_0| = |I_r|$ ($\leq t$), we construct a *configuration graph* $C = (\mathcal{V}, \mathcal{E})$, as follows:

(i) each node in C corresponds to an independent set of G with cardinality exactly t'; and

(ii) two nodes in C are joined by an edge if and only if the corresponding two independent sets are adjacent.

For an independent set I of G with $|I| = t'$, we always denote by w_I the node of C corresponding to I. Clearly, two independent sets I_0 and I_r are reconfigurable each other if and only if there is a path in C between w_{I_0} and w_{I_r}.

Notice that G has at most the number $\binom{n(G)}{t'}$ of distinct independent sets with cardinality exactly t'. Since $t' \leq t$, we thus have

$$|\mathcal{V}| \leq \binom{n(G)}{t'} < \binom{3t(d+1)}{t'} = O\left((3td)^t\right).$$

The configuration graph C above can be constructed in time $O(|\mathcal{V}|^2)$. Furthermore, by the breadth-first search on C starting from the node w_{I_0}, one can determine whether C has a path from w_{I_0} to w_{I_r} in time $O(|\mathcal{V}| + |\mathcal{E}|) = O(|\mathcal{V}|^2)$. In this way, our algorithm runs in time $O(|\mathcal{V}|^2) = O\left((3td)^{2t}\right)$ in total. □

Lemmas 3.2 and 3.3 complete the proof of Theorem 3.1. □

3.2 Shortest reconfiguration sequence

We now give an FPT algorithm which finds an actual reconfiguration sequence with the minimum length.

Theorem 3.4 Let G be a graph whose maximum degree is bounded by a fixed constant d. Let I_0 and I_r be two independent sets of G such that $|I_0| = |I_r| \leq t$ for a fixed constant t. Then, one can find a shortest reconfiguration sequence between I_0 and I_r in time $O\left((4td)^{2t} + n(G) + m(G)\right)$ if there exists.

We give such an algorithm as a proof of Theorem 3.4. Let $t' = |I_0| = |I_r| \leq t$. Although our algorithm is based on the proofs in Section 3.1, the number of vertices for the graph classification is slightly changed from $3t(d+1)$ to $4t(d+1)$; this yields that the base of the running time becomes 4 in Theorem 3.4.

We first consider the case where $n(G) < 4t(d+1)$.

Lemma 3.5 Suppose that $n(G) < 4t(d+1)$. Then, one can find a shortest reconfiguration sequence between I_0 and I_r in time $O\left((4td)^{2t}\right)$ if there exists.

Proof. As in the proof of Lemma 3.3, we construct the configuration graph $C = (\mathcal{V}, \mathcal{E})$ for G and t' in time

$$O(|\mathcal{V}|^2) = O\left(\binom{4t(d+1)}{t'}^2\right) = O\left((4td)^{2t}\right).$$

Recall that the node set of C corresponds to *all* independent sets in G of cardinality exactly t'. Therefore, a shortest reconfiguration sequence between two independent sets I_0 and I_r corresponds to a shortest path in C between the two nodes w_{I_0} and w_{I_r}. By the breadth-first search on C starting from w_{I_0}, one can find a shortest path in C in time $O(|\mathcal{V}| + |\mathcal{E}|) = O(|\mathcal{V}|^2)$ if there exists. Therefore, if $n(G) < 4t(d+1)$, one can find a shortest reconfiguration sequence in time $O(|\mathcal{V}|^2) = O\left((4td)^{2t}\right)$. □

We then consider the case where $n(G) \geq 4t(d+1)$. Notice that, since $n(G)$ is not bounded by a fixed constant, we cannot directly construct the configuration graph C for G and t' in this case. However, we will prove that only a subgraph of C having a constant number of nodes is sufficient to find a shortest reconfiguration

sequence.

Lemma 3.2 ensures that there always exists a reconfiguration sequence between I_0 and I_r in this case. Furthermore, in the proof of Lemma 3.2, we proposed a reconfiguration sequence $\mathcal{S}'$ between I_0 and I_r such that every token is moved exactly twice. Although this is not always a shortest reconfiguration sequence, the minimum length of a reconfiguration sequence between I_0 and I_r can be bounded by the length of $\mathcal{S}'$, that is, $2t'$.

Let G^- be the graph obtained from G by deleting all vertices in $N[G; I_0] \cup N[G; I_r]$. Then, by the counterpart of Eq. (1) we have $n(G^-) \geq 2t(d+1)$, and hence G^- has an independent set I'_k such that $|I'_k| = 2t'$ ($\leq 2t$). We now give the following lemma.

Lemma 3.6 There exists a shortest reconfiguration sequence $\mathcal{S}$ between I_0 and I_r such that $I \subseteq I_0 \cup I'_k \cup I_r$ for all independent sets I in $\mathcal{S}$.

Proof. Let $\mathcal{S}^* = \langle I^*_0, I^*_1, \ldots, I^*_\ell \rangle$ be an arbitrary shortest reconfiguration sequence between $I_0 = I^*_0$ and $I_r = I^*_\ell$. Then, the proof of Lemma 3.2 implies that $\ell \leq 2t'$, as we have mentioned above. Note that some independent sets in $\mathcal{S}^*$ may contain vertices in $V(G) \setminus \big(I_0 \cup I'_k \cup I_r\big)$. Let

$$V(I_0, I_r; \mathcal{S}^*) = \bigcup_{1 \leq i \leq \ell - 1} \Big(I^*_i \setminus \big(I_0 \cup I_r\big) \Big),$$

that is, $V(I_0, I_r; \mathcal{S}^*)$ is the set of all vertices that are not in $I_0 \cup I_r$ but appear in the reconfiguration sequence $\mathcal{S}^*$. Since $\ell \leq 2t'$ and $|I^*_{i+1} \setminus I^*_i| = 1$ for all i, $0 \leq i \leq \ell - 1$, we have $|V(I_0, I_r; \mathcal{S}^*)| < \ell \leq 2t'$.

Therefore, since $|I'_k| = 2t'$, one can replace all vertices in $V(I_0, I_r; \mathcal{S}^*)$ with distinct vertices in I'_k; let $\mathcal{S}$ be the resulting sequence. Recall that I'_k is an independent set of G^-, and hence no vertex in I'_k is adjacent with any vertex in $I_0 \cup I_r$. Therefore, $\mathcal{S}$ is a reconfiguration sequence between I_0 and I_r. Note that any independent set I in $\mathcal{S}$ satisfies $I \subseteq I_0 \cup I'_k \cup I_r$. Furthermore, the length of $\mathcal{S}$ is equal to that of $\mathcal{S}^*$, and hence $\mathcal{S}$ is a shortest reconfiguration sequence. □

We now give the following lemma, which completes the proof of Theorem 3.4.

Lemma 3.7 Suppose that $n(G) \geq 4t(d+1)$. Then, one can find a shortest reconfiguration sequence between I_0 and I_r in time $O\big((4t)^{2t} + n(G) + m(G)\big)$.

Proof. We first remark that an independent set I'_k of G^- with $|I'_k| = 2t'$ ($\leq 2t$) can be found in time $O\big(n(G) + m(G)\big)$ by the following simple greedy algorithm: initially, let $I'_k = \emptyset$; choose an arbitrary vertex v in G^-, and add v to I'_k; delete all vertices in $N[G^-; v]$ from G^-, and repeat. Recall that $n(G^-) \geq 2t(d+1)$ and $\big|N[G^-; v]\big| \leq d+1$ for every vertex v in G^-. Therefore, this greedy algorithm always finds an independent set I'_k with $|I'_k| = 2t'$.

Let G_{0kr} be the subgraph of G induced by the vertex-subset $I_0 \cup I'_k \cup I_r$. Notice that $n(G_{0kr}) = |I_0 \cup I'_k \cup I_r| \leq 4t'$. Let C_{0kr} be the configuration graph for G_{0kr} and the constant t'. Since G_{0kr} is an induced subgraph of G, any independent set I of G_{0kr} is an independent set of G. Then, Lemma 3.6 ensures that there exists a shortest reconfiguration sequence $\mathcal{S}$ between I_0 and I_r such that every independent set I in $\mathcal{S}$ is an independent set of G_{0kr}. Therefore, such a shortest reconfiguration sequence $\mathcal{S}$ between I_0 and I_r can be found as a shortest path in C_{0kr} between the two nodes w_{I_0} and w_{I_r}. This can be done in time $O\big((4t)^{2t}\big)$, because the number of nodes in C_{0kr} can be bounded by $\binom{n(G_{0kr})}{t'} = O\big((4t)^t\big)$.

In this way, if $n(G) \geq 4t(d+1)$, one can find a shortest reconfiguration sequence between I_0 and I_r in time $O\big((4t)^{2t} + n(G) + m(G)\big)$ in total. □

4. Concluding Remark

We remark that the running time of each of our FPT algorithms is just a single exponential with respect to the number of tokens; furthermore, the parameter d of maximum degree does not appear in the exponent.

Acknowledgments This work is partially supported by JSPS KAKENHI Grant Numbers 25106504 and 25330003 (Ito), 25104521 (Ono), 24106004 (Ono and Uehara), 26730001 (Suzuki) and 25106502 (Yamanaka).

References

[1] Bonamy, M., Johnson, M., Lignos, I., Patel, V., Paulusma, D.: On the diameter of reconfiguration graphs for vertex colourings. Electronic Notes in Discrete Mathematics 38, pp. 161–166 (2011)

[2] Bonsma, P., Cereceda, L.: Finding paths between graph colourings: PSPACE-completeness and superpolynomial distances. Theoretical Computer Science 410, pp. 5215–5226 (2009)

[3] Cereceda, L., van den Heuvel, J., Johnson, M.: Finding paths between 3-colourings. J. Graph Theory 67, pp. 69–82 (2011)

[4] Gopalan, P., Kolaitis, P.G., Maneva, E.N., Papadimitriou, C.H.: The connectivity of Boolean satisfiability: computational and structural dichotomies. SIAM J. Computing 38, pp. 2330–2355 (2009)

[5] Hearn, R.A., Demaine, E.D.: PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. Theoretical Computer Science 343, pp. 72–96 (2005)

[6] Hearn, R.A., Demaine, E.D.: Games, Puzzles, and Computation. A K Peters (2009)

[7] Ito, T., Demaine, E.D., Harvey, N.J.A., Papadimitriou, C.H., Sideri, M., Uehara, R., Uno, Y.: On the complexity of reconfiguration problems. Theoretical Computer Science 412, pp. 1054–1065 (2011)

[8] Ito, T., Kamiński, M., Demaine, E.D.: Reconfiguration of list edge-colorings in a graph. Discrete Applied Mathematics 160, pp. 2199–2207 (2012)

[9] Ito, T., Kawamura, K., Ono, H., Zhou, X.: Reconfiguration of list $L(2,1)$-labelings in a graph. In: Proc. of ISAAC 2012, Lecture Notes in Computer Science vol. 7676, pp. 34–43 (2012)

[10] Ito, T., Kawamura, K., Zhou, X.: An improved sufficient condition for reconfiguration of list edge-colorings in a tree. IEICE Trans. on Information and Systems E95-D, pp. 737–745 (2012)

[11] Kamiński, M., Medvedev, P., Milanič, M.: Shortest paths between shortest paths. Theoretical Computer Science 412, pp. 5205–5210 (2011)

[12] Kamiński, M., Medvedev, P., Milanič, M.: Complexity of independent set reconfigurability problems. Theoretical Computer Science 439, pp. 9–15 (2012)

[13] Makino, K., Tamaki, S., Yamamoto, M.: An exact algorithm for the Boolean connectivity problem for k-CNF. Theoretical Computer Science 412, pp. 4613–4618 (2011)

[14] Mouawad, A.E., Nishimura, N., Raman, V., Simjour, N., Suzuki, A.: On the parameterized complexity of reconfiguration problems. In Proc. of IPEC 2013, Lecture Notes in Computer Science vol. 8246, pp. 281–294 (2013)