FPT algorithms for Token Jumping on Graphs

TAKEHIRO ITO\footnote{Graduate School of Information Sciences, Tohoku University, Aoba-yama 6-6-05, Sendai, 980-8579, Japan.}, MARCIN KAMIŃSKI\footnote{Dept. of Mathematics, Computer Science and Mechanics, University of Warsaw, Banacha 2, 02-097, Warsaw, Poland.}, HIROTAKA ONO\footnote{Faculty of Economics, Kyushu University, Hakoizaki 6-19-1, Higashi-ku, Fukuoka, 812-8581, Japan.}, AKIRA SUZUKI\footnote{School of Information Science, JAIST, Asahidai 1-1, Nomi, Ishikawa, 923-1292, Japan.}, RYUHEI UEHARA\footnote{Dept. of Electrical Engineering and Computer Science, Iwate University, Ueda 4-3-5, Morioka, Iwate 020-8551, Japan.}, KATSUHISA YAMANAKA\footnote{takehiro@ecei.tohoku.ac.jp \hspace{1cm} nmk@mimuw.edu.pl \hspace{1cm} hirotaka@en.kyushu-u.ac.jp \hspace{1cm} a.suzuki@ecei.tohoku.ac.jp \hspace{1cm} uehara@jaist.ac.jp \hspace{1cm} yamanaka@cis.iwate-u.ac.jp}

Abstract: Suppose that we are given two independent sets $I_0$ and $I_i$ of a graph such that $|I_0| = |I_i|$, 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 $I_i$ 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. \cite{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 $I_i$ of a graph $G = (V, E)$ such that $|I_0| = |I_i|$, 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 $(I_{0}, I_{1}, \ldots, I_{i})$ of independent sets of $G$ such that
(a) $I_{i} = I_{i}$ and $|I_{i}| = |I_{0}| = |I_{i}|$ for all $i, 1 \leq i \leq \ell$; and
(b) for each index $i$, $1 \leq i \leq \ell$, $I_{i}$ 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 $(I_{0}, I_{1}, \ldots, I_{i})$ of independent sets which transforms $I_0$ into $I_i$.

Recently, this type of problems have been studied extensively in the framework of reconfiguration problems \cite{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 configuration 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 \cite{5, 6, 7, 12, 14}, satisfiability \cite{4, 13}, set cover, clique, matching \cite{7}, vertex-coloring \cite{1, 2, 3}, list edge-coloring \cite{8, 10}, list $L(2,1)$-labeling \cite{9}, shortest path \cite{11}, etc.

1.1 Reconfiguration rules and related results

The original reconfiguration problem for independent set was introduced by Hearn and Demaine \cite{5}, which employs another reconfiguration rule. Indeed, there are three reconfiguration problems for independent set (ISRReconf for short) under different reconfiguration rules, as follows.

- **Token Sliding (TS)** \cite{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
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 ISRConf 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 ISRConf under the TS rule is PSPACE-complete for planar graphs of maximum degree three. Then, Bonsma and Cerereca [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 ISRConf 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 moves.

### 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 $I$ and $I_j$ be two independent sets of the same cardinality in a graph $G = (V, E)$. We say that $I$ and $I_j$ are adjacent if there exists exactly one pair of vertices $u$ and $v$ such that $I_v \setminus I_j = \{u\}$ and $I \setminus I_j = \{v\}$, that is, $I$ can be obtained from $I_j$ by moving the token on a vertex $u \in I$, to another vertex $v \in V \setminus I$. 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 $(I_1, I_2, \ldots, I_l)$ of independent sets of $G$ such that $I_1 = I$, $I_l = I'$, and $I_{r-1}$ and $I_r$ are adjacent for $r = 2, 3, \ldots, \ell$. 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 $S$ is defined as the number of independent sets contained in $S$. 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 $\Delta$. 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 $O((3td)^5)$.

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 $O((3td)^5)$-time algo-
Algorithm that determines whether $I_0$ and $I_1$ 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 $I_0$ and $I_1$ be an arbitrary pair of independent sets of $G$ such that $|I_0| = |I_1| \leq t$. Then, $I_0$ and $I_1$ 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 $I_0$ and $I_1$.

Let $G'$ be the graph obtained from $G$ by deleting all vertices in $N(G; I_0) \cup N(G; I_1)$. Since all neighbors of the vertices in $I_0 \cup I_1$ have been deleted from $G$, no vertex in $G'$ is adjacent to any vertex in $I_0 \cup I_1$. Therefore, if $G'$ has an independent set $I_0'$ with $|I_0'| \geq t$, then there is a reconfiguration sequence between $I_0$ and $I_1$, as follows: move all tokens on the vertices in $I_0$ to the vertices in $I_1$ one by one; and move all tokens on the vertices in $I_1$ to the vertices in $I_0$ one by one.

To complete the proof, we thus show that $G'$ has an independent set $I_0'$ with $|I_0'| \geq t$ if $n(G) \geq 3t(d + 1)$. Since $\Delta(G) \leq d$, we clearly have $|N(G; v)| \leq d + 1$ for every vertex $v$ in $G$. Since $|I_0| \leq t$, we thus have

$$|N(G; I_0)| \leq \sum_{v \in I_0} |N(G; v)| \leq t(d + 1).$$

Similarly, we have $|N(G; I_1)| \leq t(d + 1)$. Therefore,

$$n(G') \geq n(G) - |N(G; I_0)| - |N(G; I_1)| \geq t(d + 1).$$

We now suppose for a contradiction that $|I_{\text{max}}| < t$ holds for a maximum independent set $I_{\text{max}}$ of $G'$. Then, we have

$$|N(G'; I_{\text{max}})| \leq \sum_{v \in I_{\text{max}}} |N(G; v)| < t(d + 1),$$

and hence by Eq. (1)

$$n(G') - |N(G'; I_{\text{max}})| \geq 1.$$

Therefore, the graph obtained from $G'$ by deleting all vertices in $N(G'; I_{\text{max}})$ is non-empty, and hence we can add at least one vertex to $I_{\text{max}}$. This contradicts the assumption that $I_{\text{max}}$ is a maximum independent set of $G'$. Therefore, $|I_{\text{max}}| \geq t$, and hence $G'$ has an independent set $I_0'$ with $|I_0'| \geq t$.

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 $O((3td)^2)$-time algorithm which determines whether $I_0$ and $I_1$ are reconfigurable each other.

**Proof.** We give such an algorithm. For a graph $G$ and a constant $t' = |I_0| = |I_1| \leq t$, we construct a configuration graph $C = (V, E)$, 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 $I_1$ are reconfigurable each other if and only if there is a path in $C$ between $w_{I_0}$ and $w_{I_1}$.

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

$$|V| \leq (n(G)^{t'}) < (3td + 1) < (3td + 1) = O((3td)^2).$$

The configuration graph $C$ above can be constructed in time $O(|V|^2)$. 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 $w_{I_1}$ in time $O(|V| + |E|) = O(|V|^2)$. In this way, our algorithm runs in time $O(|V|^2) = O((3td)^2)$ in total. □

**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 $I_1$ be two independent sets of $G$ such that $|I_0| = |I_1| \leq t$ for a fixed constant $t$. Then, one can find a shortest reconfiguration sequence between $I_0$ and $I_1$ in time $O((4td)^2 + n(G) + m(G))$ if there exists.

We give such an algorithm as a proof of Theorem 3.4. Let $t' = |I_0| = |I_1| \leq t$. 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 $I_1$ in time $O((4td)^2)$ if there exists.

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

$$O(|V|^2) = O\left(\left(4td + 1\right)^2\right) = O(4td)^2.$$

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 $I_1$ corresponds to a shortest path in $C$ between the two nodes $w_{I_0}$ and $w_{I_1}$. By the breadth-first search on $C$ starting from $w_{I_0}$, one can find a shortest path in $C$ in time $O(|V| + |E|) = O(|V|^2)$ if there exists. Therefore, if $n(G) < 4t(d + 1)$, one can find a shortest reconfiguration sequence in time $O(|V|^2) = O(4td)^2$. □

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.
Lemma 3.6 There exists a shortest reconfiguration sequence \( S \) between \( I_0 \) and \( I_t \) for any independent sets \( I_0 \) and \( I_t \) that is, \( 2t' \).

Let \( G' \) be the graph obtained from \( G \) by deleting all vertices in \( N(G; I_0) \cup N(G; I_t) \). Then, by the counterpart of Eq. (1) we have \( |G' (\mathcal{v})| \geq 2(t + d + 1) \). Hence, \( G' \) has a shortest reconfiguration sequence \( |I'_{(t+1)}| \geq 2d \). Let \( G_{okr} \) be the configuration graph for the reconfiguration sequence \( I_0 \) and \( I_t \), such that every independent set in \( G_{okr} \) can be found as a shortest path between the two nodes \( w_{I_0} \) and \( w_{I_t} \). This can be done in time \( O((d + 1) \cdot 4) \), because the number of nodes in \( G_{okr} \) can be bounded by \( n_{okr} = O(4t') \).

In this case, if \( n(G) \geq 4d + 1 \), one can find a shortest reconfiguration sequence between \( I_0 \) and \( I_t \), in time \( O((d + 1) \cdot 4t' \cdot m(G) + m(G)) \) in total.

\[ \square \]

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