Sliding tokens on a cactus

Duc A. Hoang Ryuhei Uehara

December 12–14, 2016

Japan Advanced Institute of Science and Technology
Asahidai 1-1, Nomi, Ishikawa 923-1292, Japan.
{hoanganhduc, uehara}@jaist.ac.jp

Outline

- Reconfiguration Problems.
- The SLIDING TOKEN problem for a cactus.
- Interesting open questions.

Reconfiguration Problems

Reconfiguration Problems

- INSTANCE:
  1. Collection of configurations.
  2. Allowed transformation rule(s).
- QUESTION: Decide if configuration A can be transformed to configuration B using the given rule(s), while maintaining a configuration throughout.

Reconfiguration Problems

- INSTANCE:
  1. Collection of configurations.
  2. Allowed transformation rule(s).
- QUESTION: Decide if configuration A can be transformed to configuration B using the given rule(s), while maintaining a configuration throughout.

configuration A

8	15	13	3
10		14	7
5	1	2	4
9	12	11	6

configuration B

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

Figure 1: The 15-puzzles. (In general, the $(k^2 - 1)$-puzzles.)

Reconfiguration Problems

- INSTANCE:
  1. Collection of configurations.
  2. Allowed transformation rule(s).
- QUESTION: Decide if configuration A can be transformed to configuration B using the given rule(s), while maintaining a configuration throughout.

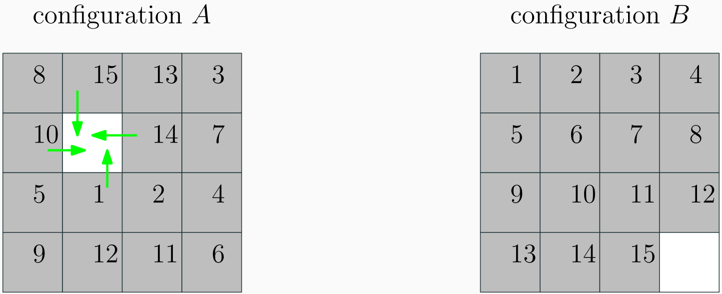


Figure 1: The 15-puzzles. (In general, the $(k^2 - 1)$-puzzles.)

Reconfiguration Problems

- INSTANCE:
  1. Collection of configurations.
  2. Allowed transformation rule(s).
- QUESTION: Decide if configuration A can be transformed to configuration B using the given rule(s), while maintaining a configuration throughout.

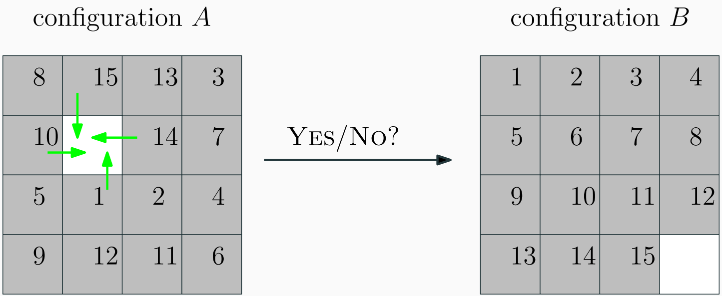
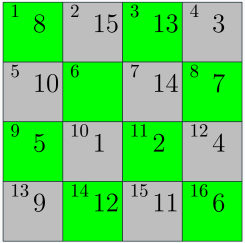


Figure 1: The 15-puzzles. (In general, the $(k^2 - 1)$-puzzles.)

Reconfiguration Problems

- INSTANCE:
  1. Collection of configurations.
  2. Allowed transformation rule(s).
- QUESTION: Decide if configuration A can be transformed to configuration B using the given rule(s), while maintaining a configuration throughout.



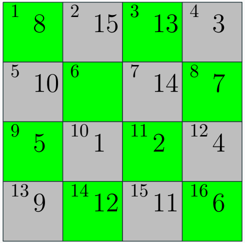
Parity (even/odd) Checking ($O(n)$ time)

$(1, 8, 7, 14, 12, 4, 3, 13, 9, 5, 10)(2, 15, 11)(6, 16)$

Figure 1: The 15-puzzles. (In general, the $(k^2 - 1)$-puzzles.)

Reconfiguration Problems

- INSTANCE:
  1. Collection of configurations.
  2. Allowed transformation rule(s).
- QUESTION: Decide if configuration A can be transformed to configuration B using the given rule(s), while maintaining a configuration throughout.



Parity (even/odd) Checking ($O(n)$ time)

If YES, need at most $O(n^3)$ moves.

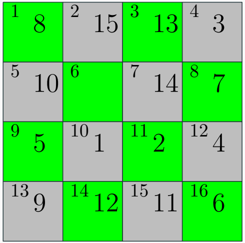
[Kornhauser, Miller, and Spirakis 1984]

$(1, 8, 7, 14, 12, 4, 3, 13, 9, 5, 10)(2, 15, 11)(6, 16)$

Figure 1: The 15-puzzles. (In general, the $(k^2 - 1)$-puzzles.)

Reconfiguration Problems

- INSTANCE:
  1. Collection of configurations.
  2. Allowed transformation rule(s).
- QUESTION: Decide if configuration A can be transformed to configuration B using the given rule(s), while maintaining a configuration throughout.



Parity (even/odd) Checking ($O(n)$ time)

If YES, need at most $O(n^3)$ moves.

[Kornhauser, Miller, and Spirakis 1984]

Find minimum number of moves? - NP-complete

[Ratner and Warmuth 1990]

$(1, 8, 7, 14, 12, 4, 3, 13, 9, 5, 10)(2, 15, 11)(6, 16)$

Figure 1: The 15-puzzles. (In general, the $(k^2 - 1)$-puzzles.)

Reconfiguration Problems

- INSTANCE:
  1. Collection of configurations.
  2. Allowed transformation rule(s).
- QUESTION: Decide if configuration A can be transformed to configuration B using the given rule(s), while maintaining a configuration throughout.

Reconfiguration variants have been studied for several well-known problems:

- SATISFIABLILITY,
- INDEPENDENT SET, VERTEX COVER, CLIQUE,
- VERTEX-COLORING, (LIST) EDGE-COLORING,
- and so on.

Recent Survey on Reconfiguration Problems

Jan van den Heuvel (2013). “The complexity of change”. In: *Surveys in Combinatorics 2013*. Ed. by Simon R. Blackburn et al. Cambridge University Press, pp. 127–160

The Sliding Token problem for a cactus

- Instance:
  - Collection of independent sets of a graph.
  - Allowed transformation rule: Token Sliding (TS).
- Question: Decide if there exists a sequence of independent sets (called at TS-sequence) $\mathcal{S} = \langle \mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_\ell \rangle$ that transforms (reconfigures) $\mathbf{I} = \mathbf{I}_1$ to $\mathbf{J} = \mathbf{I}_\ell$, where $\mathbf{I}_{i+1}$ is obtained from $\mathbf{I}_i$ by sliding a token from a vertex $u \in \mathbf{I}_i \setminus \mathbf{I}_{i+1}$ to its neighbor $v \in \mathbf{I}_{i+1} \setminus \mathbf{I}_i$, $i \in \{1, \dots, \ell - 1\}$.

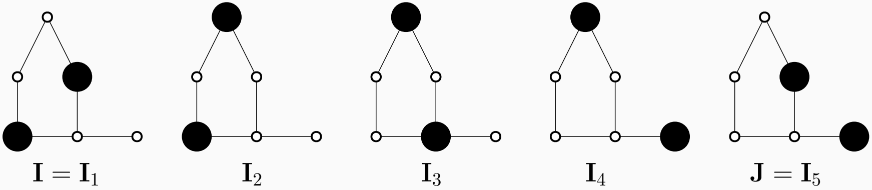


Figure 2: A TS-sequence that reconfigures $\mathbf{I} = \mathbf{I}_1$ to $\mathbf{J} = \mathbf{I}_5$. Vertices of an independent set are marked with black circles (tokens).

Complexity status of Sliding Token

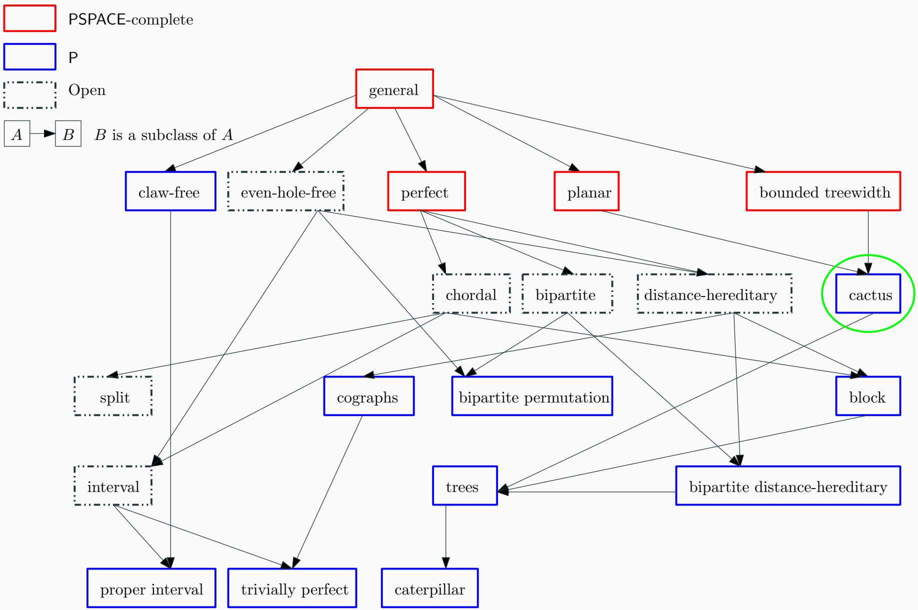


Figure 3: Complexity status of Sliding Token.

A cactus

A cactus is a graph such that every block (i.e., maximal biconnected subgraph) is either an edge or a simple cycle.

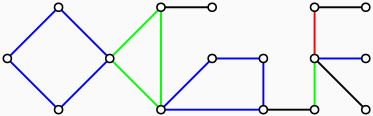


Figure 4: A cactus and its blocks. Two blocks sharing the same vertex are of diffenrent colors.

Why study SLIDING TOKEN for a cactus?

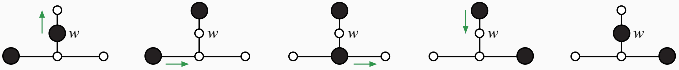
There are a few reasons that motivate our study.

1. We want to understand Intractability vs Polynomial-time tractability of SLIDING TOKEN for bounded-treewidth/planar graphs and their subclasses: Before cacti, the "largest" subclass with polynomial-time tractability is trees.

Why study Sliding Token for a cactus?

There are a few reasons that motivate our study.

1. We want to understand Intractability vs Polynomial-time tractability of Sliding Token for bounded-treewidth/planar graphs and their subclasses: Before cacti, the "largest" subclass with polynomial-time tractability is trees.
2. Even for trees, a token sometimes needs to make "detours" to preserve the independence property. In general, there might be a YES-instance that requires super-polynomial number of token-slides. (see [Demaine et al. 2015])



(a) $I_b = I_1$ (b) I_2 (c) I_3 (d) I_4 (e) $I_r = I_5$

Figure 5: Detours in a tree.

Why study SLIDING TOKEN for a cactus?

There are a few reasons that motivate our study.

1. We want to understand Intractability vs Polynomial-time tractability of SLIDING TOKEN for bounded-treewidth/planar graphs and their subclasses: Before cacti, the "largest" subclass with polynomial-time tractability is trees.
2. Even for trees, a token sometimes needs to make "detours" to preserve the independence property. In general, there might be a YES-instance that requires super-polynomial number of token-slides. (see [Demaine et al. 2015])

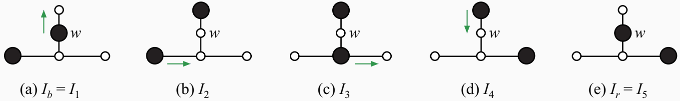


Figure 5: Detours in a tree.

3. In a cactus, there might be more than one path connecting two given vertices. It follows that there might be exponential number of "routes" that a token can be moved.

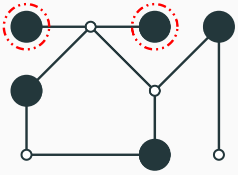
Given an instance $(G, \mathbf{I}, \mathbf{J})$ of SLIDING TOKEN, where $\mathbf{I}$ and $\mathbf{J}$ are independent sets of a cactus G, we can

1. Characterize all structures that forbid the existence of a TS-sequence between $\mathbf{I}$ and $\mathbf{J}$ in polynomial time.
   - A token that cannot be slid at all (called a $(G, \mathbf{I})$-rigid token).
   - A cycle whose inside-tokens form a maximum independent set of it and no token can be slid "out" or "in" (called a $(G, \mathbf{I})$-confined cycle).
2. Prove the existence of a TS-sequence between $\mathbf{I}$ and $\mathbf{J}$ when no such structures exist.

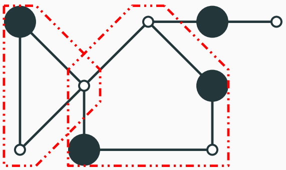
The general idea

Lemma 1

One can find all $(G, \mathbf{I})$-rigid tokens in $O(n^2)$ time, where $n = |V(G)|$. Without $(G, \mathbf{I})$-rigid tokens, one can find all $(G, \mathbf{I})$-confined cycles in $O(n^2)$ time.



(a) $(G, \mathbf{I})$-rigid tokens



(b) $(G, \mathbf{I})$-confined cycles

Figure 6: Examples of the forbidden structures.

Lemma 2

If the set of $(G, \mathbf{I})$-rigid tokens and $(G, \mathbf{J})$-rigid tokens are different, then it is a NO-instance. Without $(G, \mathbf{I})$-rigid and $(G, \mathbf{J})$-rigid tokens, if the set of $(G, \mathbf{I})$-confined cycles and $(G, \mathbf{J})$-confined cycles are different, then it is a NO-instance.

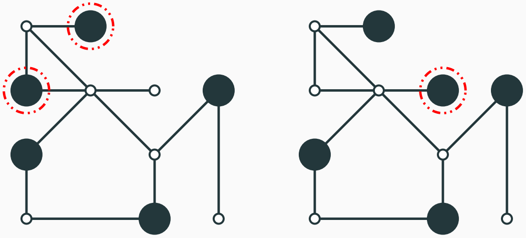


Figure 7: The set of $(G, \mathbf{I})$-rigid tokens and $(G, \mathbf{J})$-rigid tokens are different.

Lemma 2

If the set of $(G, \mathbf{I})$-rigid tokens and $(G, \mathbf{J})$-rigid tokens are different, then it is a NO-instance. Without $(G, \mathbf{I})$-rigid and $(G, \mathbf{J})$-rigid tokens, if the set of $(G, \mathbf{I})$-confined cycles and $(G, \mathbf{J})$-confined cycles are different, then it is a NO-instance.

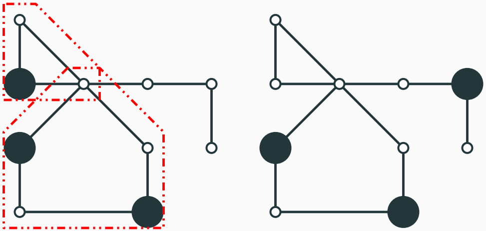


Figure 8: The set of $(G, \mathbf{I})$-confined cycles and $(G, \mathbf{J})$-confined cycles are different.

Lemma 3

Without rigid tokens and confined cycles (for both $\mathbf{I}$ and $\mathbf{J}$), $\mathbf{I}$ can be reconfigured to $\mathbf{J}$ if and only if $|\mathbf{I}| = |\mathbf{J}|$.

Proof Idea: Construct an "intermediate" independent set $\mathbf{I}^*$ such that both $\mathbf{I}$ and $\mathbf{J}$ can be reconfigured to $\mathbf{I}^*$.

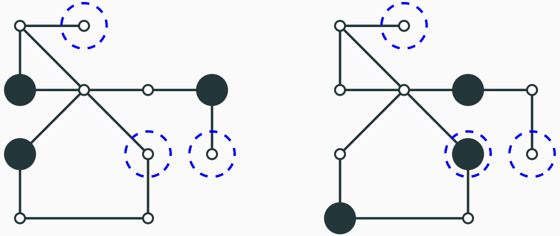


Figure 9: Illustration of **Lemma 3**.

Lemmas 1–3 give rise to the following polynomial-time algorithm. For an instance $(G, \mathbf{I}, \mathbf{J})$ of SLIDING TOKEN, where G is a cactus and $\mathbf{I}$, $\mathbf{J}$ are two independent sets of G.

- **Step 1:**
  - **Step 1-1:** If the set of $(G, \mathbf{I})$-rigid tokens and $(G, \mathbf{J})$-rigid tokens are different, return NO.
  - **Step 1-2:** Otherwise, remove all vertices where $(G, \mathbf{I})$-rigid tokens are placed and its neighbors, and go to **Step 2**. Let G' be the resulting graph.
- **Step 2:**
  - **Step 2-1:** If the set of $(G', \mathbf{I} \cap G')$-confined cycles and $(G', \mathbf{J} \cap G')$-confined cycless are different, return NO.
  - **Step 2-2:** Otherwise, remove all $(G', \mathbf{I} \cap G')$-confined cycles, and go to **Step 3**. Let G'' be the resulting graph.
- **Step 3:** If $|\mathbf{I} \cap F| \neq |\mathbf{J} \cap F|$ for some component F of G'' then return NO. Otherwise, return YES.

Our algorithm

Lemmas 1–3 give rise to the following polynomial-time algorithm. For an instance $(G, \mathbf{I}, \mathbf{J})$ of SLIDING TOKEN, where G is a cactus and $\mathbf{I}$, $\mathbf{J}$ are two independent sets of G.

- **Step 1:**
  - **Step 1-1:** If the set of $(G, \mathbf{I})$-rigid tokens and $(G, \mathbf{J})$-rigid tokens are different, return NO.
  - **Step 1-2:** Otherwise, remove all vertices where $(G, \mathbf{I})$-rigid tokens are placed and its neighbors, and go to **Step 2**. Let G' be the resulting graph.
- **Step 2:**
  - **Step 2-1:** If the set of $(G', \mathbf{I} \cap G')$-confined cycles and $(G', \mathbf{J} \cap G')$-confined cycless are different, return NO.
  - **Step 2-2:** Otherwise, remove all $(G', \mathbf{I} \cap G')$-confined cycles, and go to **Step 3**. Let G'' be the resulting graph.
- **Step 3:** If $|\mathbf{I} \cap F| \neq |\mathbf{J} \cap F|$ for some component F of G'' then return NO. Otherwise, return YES.

Our algorithm

Lemmas 1–3 give rise to the following polynomial-time algorithm. For an instance $(G, \mathbf{I}, \mathbf{J})$ of Sliding Token, where G is a cactus and $\mathbf{I}$, $\mathbf{J}$ are two independent sets of G.

- **Step 1:**
  - **Step 1-1:** If the set of $(G, \mathbf{I})$-rigid tokens and $(G, \mathbf{J})$-rigid tokens are different, return NO.
  - **Step 1-2:** Otherwise, remove all vertices where $(G, \mathbf{I})$-rigid tokens are placed and its neighbors, and go to **Step 2**. Let G' be the resulting graph.
- **Step 2:**
  - **Step 2-1:** If the set of $(G', \mathbf{I} \cap G')$-confined cycles and $(G', \mathbf{J} \cap G')$-confined cycless are different, return NO.
  - **Step 2-2:** Otherwise, remove all $(G', \mathbf{I} \cap G')$-confined cycles, and go to **Step 3**. Let G'' be the resulting graph.
- **Step 3:** If $|\mathbf{I} \cap F| \neq |\mathbf{J} \cap F|$ for some component F of G'' then return NO. Otherwise, return YES.

Lemmas 1–3 give rise to the following polynomial-time algorithm. For an instance $(G, \mathbf{I}, \mathbf{J})$ of Sliding Token, where G is a cactus and $\mathbf{I}$, $\mathbf{J}$ are two independent sets of G.

- **Step 1:**
  - **Step 1-1:** If the set of $(G, \mathbf{I})$-rigid tokens and $(G, \mathbf{J})$-rigid tokens are different, return NO.
  - **Step 1-2:** Otherwise, remove all vertices where $(G, \mathbf{I})$-rigid tokens are placed and its neighbors, and go to **Step 2**. Let G' be the resulting graph.
- **Step 2:**
  - **Step 2-1:** If the set of $(G', \mathbf{I} \cap G')$-confined cycles and $(G', \mathbf{J} \cap G')$-confined cycless are different, return NO.
  - **Step 2-2:** Otherwise, remove all $(G', \mathbf{I} \cap G')$-confined cycles, and go to **Step 3**. Let G'' be the resulting graph.
- **Step 3:** If $|\mathbf{I} \cap F| \neq |\mathbf{J} \cap F|$ for some component F of G'' then return NO. Otherwise, return YES.

Our algorithm

Lemmas 1–3 give rise to the following polynomial-time algorithm. For an instance $(G, \mathbf{I}, \mathbf{J})$ of SLIDING TOKEN, where G is a cactus and $\mathbf{I}$, $\mathbf{J}$ are two independent sets of G.

- **Step 1:**
  - **Step 1-1:** If the set of $(G, \mathbf{I})$-rigid tokens and $(G, \mathbf{J})$-rigid tokens are different, return NO.
  - **Step 1-2:** Otherwise, remove all vertices where $(G, \mathbf{I})$-rigid tokens are placed and its neighbors, and go to **Step 2**. Let G' be the resulting graph.
- **Step 2:**
  - **Step 2-1:** If the set of $(G', \mathbf{I} \cap G')$-confined cycles and $(G', \mathbf{J} \cap G')$-confined cycless are different, return NO.
  - **Step 2-2:** Otherwise, remove all $(G', \mathbf{I} \cap G')$-confined cycles, and go to **Step 3**. Let G'' be the resulting graph.
- **Step 3:** If $|\mathbf{I} \cap F| \neq |\mathbf{J} \cap F|$ for some component F of G'' then return NO. Otherwise, return YES.

Interesting open questions

Interesting open questions

1. SLIDING TOKEN for bipartite graphs is still open. Unlike a cactus, two cycles of a bipartite graph may have more than one vertex in common. Polynomial results are known for bipartite permutation graphs [Fox-Epstein et al. 2015].
2. Given a YES-instance, finding a shortest TS-sequence is open even for trees. The only known polynomial result regarding this problem is the case for caterpillars [Yamada and Uehara 2016].
3. It is interesting to find a graph class $\mathcal{G}$ with the property that SLIDING TOKEN is polynomial-time solvable for $\mathcal{G}$, and finding a shortest TS-sequence for $\mathcal{G}$ is NP-hard. We conjecture that $\mathcal{G}$ might be cacti.

Bibliography I

- Demaine, Erik D., Martin L. Demaine, Eli Fox-Epstein, Duc A. Hoang, Takehiro Ito, Hirotaka Ono, Yota Otachi, Ryuhei Uehara, and Takeshi Yamada (2015). "Linear-time algorithm for sliding tokens on trees". In: *Theoretical Computer Science* 600, pp. 132–142.
- Fox-Epstein, Eli, Duc A. Hoang, Yota Otachi, and Ryuhei Uehara (2015). "Sliding Token on Bipartite Permutation Graphs". In: *Algorithms and Computation - ISAAC 2015*. Ed. by Khaled Elbassioni and Kazuhisa Makino. Vol. 9472. LNCS. Springer, pp. 237–247.
- Kornhauser, Daniel, Gary L. Miller, and Paul Spirakis (1984). "Coordinating Pebble Motion on Graphs, The Diameter of Permutation Groups, and Applications". In: *25th Annual Symposium on Foundations of Computer Science*, pp. 241–250.
- Ratner, Daniel and Manfred Warmuth (1990). "The $(n^2 - 1)$-puzzle and related relocation problems". In: *Journal of Symbolic Computation* 10.2, pp. 111–137.
- van den Heuvel, Jan (2013). "The complexity of change". In: *Surveys in Combinatorics 2013*. Ed. by Simon R. Blackburn, Stefanie Gerke, and Mark Wildon. Cambridge University Press, pp. 127–160.

JAPAN ADVANCED INSTITUTE OF SCIENCE AND TECHNOLOGY 1990

Yamada, Takeshi and Ryuhei Uehara (2016). “Shortest Reconfiguration of Sliding Tokens on a Caterpillar”. In: *Algorithms and Computation - WALCOM 2016*. Ed. by Mohammad Kaykobad and Rossella Petreschi. Vol. 9627. LNCS. Springer, pp. 236–248.