Shortest Reconfiguration Sequence for Sliding Tokens on Spiders

Duc A. Hoang[1] Amanj Khorramian[2] Ryuhei Uehara[1]

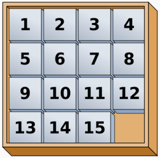
August 26–27, 2018

[1]School of Information Science, JAIST, Japan

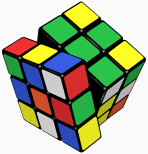
[2]University of Kurdistan, Sanandaj, Iran

Reconfiguration and Sliding Tokens

Reconfiguration: An Overview



15-PUZZLE



RUBIK'S CUBE

RUSH-HOUR

They are all examples of Reconfiguration Problems:

Given two configurations, and a specific rule describing how a configuration can be transformed into a (slightly) different one

Ask whether one can transform one configuration into another by applying the given rule repeatedly

The figures were originally downloaded from various online sources, especially Wikipedia

Reconfiguration: An Overview

New insights into the computational complexity theory

Given	Two configurations A, B, and a transformation rule
Decision	Decide if A can be transformed into B
Find	A transformation sequence between them?
Shortest	A shortest transformation sequence between them?

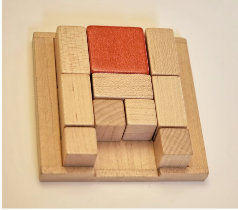
See also the "Masterclass Talk: Algorithms and Complexity for Japanese Puzzles" by R. Uehara at ICALP 2015

The figures were originally downloaded from various online sources, especially Wikipedia

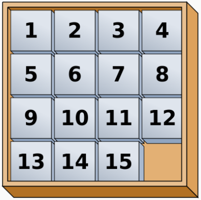
Reconfiguration: An Overview

New insights into the computational complexity theory

	SLIDING-BLOCK PUZZLE	15-PUZZLE
Decision	PSPACE-complete	Linear
Find	PSPACE-complete	Poly-time
Shortest	PSPACE-complete	NP-complete



SLIDING-BLOCK PUZZLE



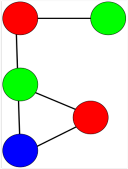
15-PUZZLE

See also the "Masterclass Talk: Algorithms and Complexity for Japanese Puzzles" by R. Uehara at ICALP 2015

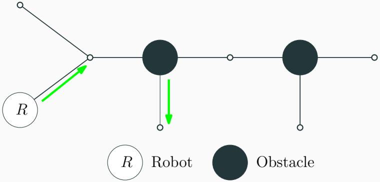
The figures were originally downloaded from various online sources, especially Wikipedia

Reconfiguration: An Overview

Real-world situations involving movement and change



Frequency Re-Assignment



Robot Motion

Assignment $\equiv$ Vertex-Coloring

Re-assign $\equiv$ Re-color Vertices

Robots & Obstacles $\equiv$ Tokens

Moving Robots $\equiv$ Sliding Tokens

Surveys on Reconfiguration

Jan van den Heuvel (2013). "The Complexity of Change". In: *Surveys in Combinatorics*. Vol. 409. London Mathematical Society Lecture Note Series. Cambridge University Press, pp. 127–160. DOI: 10.1017/CBO9781139506748.005

Naomi Nishimura (2018). "Introduction to Reconfiguration". In: *Algorithms* 11.4. (article 52). DOI: 10.3390/a11040052

Online Web Portal

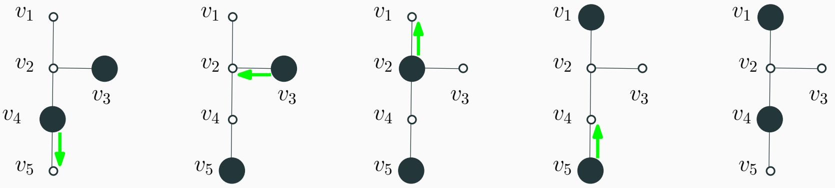
http://www.ecei.tohoku.ac.jp/alg/core/

The Sliding Token problem

Sliding Token [Hearn and Demaine 2005]

Given two independent sets (token sets) I, J of a graph G, and the Token Sliding (TS) rule

Ask whether there is a TS-sequence that transforms I into J (and vice versa)



$I = I_1$ I_2 I_3 I_4 $J = I_5$

A TS-sequence that transforms $I = I_1$ into $J = I_5$. Vertices of an independent set are marked with black circles (tokens).

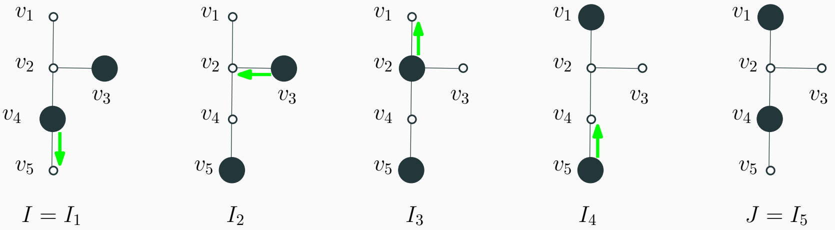
Note: This is a variant of Sliding-block Puzzle

The SHORTEST SLIDING TOKEN problem

SHORTEST SLIDING TOKEN [Yamada and Uehara 2016]

Given a yes-instance (G, I, J) of SLIDING TOKEN, where I, J are independent sets of a graph G

Ask find a shortest TS-sequence that transforms I into J (and vice versa)



A shortest TS-sequence that transforms $I = I_1$ into $J = I_5$. Vertices of an independent set are marked with black circles (tokens).

Note: This is a variant of SLIDING-BLOCK PUZZLE

The Shortest Sliding Token problem

Theorem (Kamiński et al. 2012)

It is is NP-complete to decide if there is a TS-sequence having at most ℓ token-slides between two independent sets I, J of a perfect graph G even when ℓ is polynomial in $|V(G)|$.

Theorem (Kamiński et al. 2012)

Shortest Sliding Token *can be solved in linear time for cographs (P_4-free graphs).*

Theorem (Yamada and Uehara 2016)

Shortest Sliding Token *can be solved in polynomial time for proper interval graphs, trivially perfect graphs, and caterpillars.*

The Shortest Sliding Token problem

Very recently, it has been announced that

Theorem (Sugimori, AAAC 2018)

Shortest Sliding Token *can be solved in* $O(poly(n))$ *when the input graph is a tree* T *on* n *vertices.*

- Sugimori-san's algorithm uses a dynamic programming approach. (We believe that it is correct.)
- The order of $poly(n)$ seems to be large.

The Shortest Sliding Token problem

Very recently, it has been announced that

Theorem (Sugimori, AAAC 2018)

Shortest Sliding Token *can be solved in* $O(poly(n))$ *when the input graph is a tree* T *on* n *vertices.*

- Sugimori-san's algorithm uses a dynamic programming approach. (We believe that it is correct.)
- The order of $poly(n)$ seems to be large.

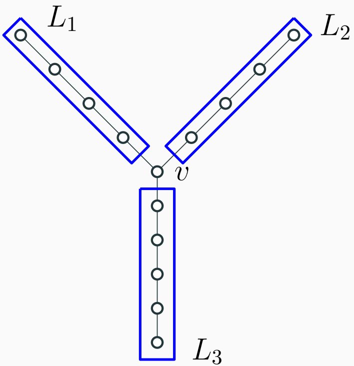
Theorem (Our Result)

Shortest Sliding Token *can be solved in* $O(n^2)$ *when the input graph is a spider* G *(i.e., a tree having exactly one vertex of degree at least* 3*) on* n *vertices.*

- We hope that our algorithm provides new insights into improving Sugimori-san's algorithm.

Shortest Sliding Token for Spiders

Spider Graphs



A spider graph

A spider G is specified in terms of

- a body vertex v whose degree is at least 3; and
- $d = \deg_G(v)$ legs $L_1, L_2, \ldots, L_d$ attached to v

Our Approach

The body vertex v is crucial. Roughly speaking, we explicitly construct a shortest TS-sequence when

- $\boxed{\max\{|I \cap N_G(v)|, |J \cap N_G(v)|\} = 0}$
  No token is in the neighbor $N_G(v)$ of v
- $\boxed{\max\{|I \cap N_G(v)|, |J \cap N_G(v)|\} \leq 1}$
  At most one token (from either I or J) is in the neighbor $N_G(v)$ of v
- $\boxed{\max\{|I \cap N_G(v)|, |J \cap N_G(v)|\} \geq 2}$
  At least two tokens (from either I or J) are in the neighbor $N_G(v)$ of v

Target assignments

A target assignment is simply a bijective mapping $f : I \to J$.

Observe that

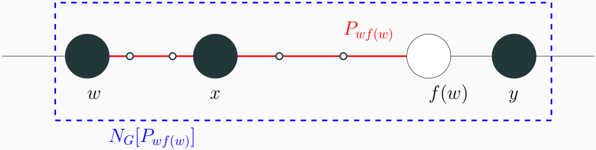
- Any TS-sequence S induces a target assignment f_S.
- Thus, each S uses at least $\sum_{w \in I} \mathrm{dist}_G(w, f_S(w))$ token-slides.

Indeed,

Lemma (Key Lemma)

One can construct in linear time a target assignment f that minimizes $\sum_{w \in I} \mathrm{dist}_G(w, f(w))$, where $\mathrm{dist}_G(x, y)$ denotes the distance between two vertices x, y of a spider G.

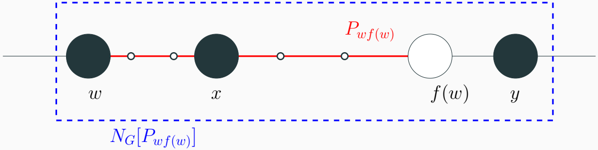
Case 1: $\max\{|I \cap N_G(v)|, |J \cap N_G(v)|\} = 0$



Observation

In the figure above, w can be moved to $f(w)$ along the shortest path $P_{wf(w)}$ between them only after both x and y are moved.

Case 1: $\max\{|I \cap N_G(v)|, |J \cap N_G(v)|\} = 0$



Observation

In the figure above, w can be moved to $f(w)$ along the shortest path $P_{wf(w)}$ between them only after both x and y are moved.

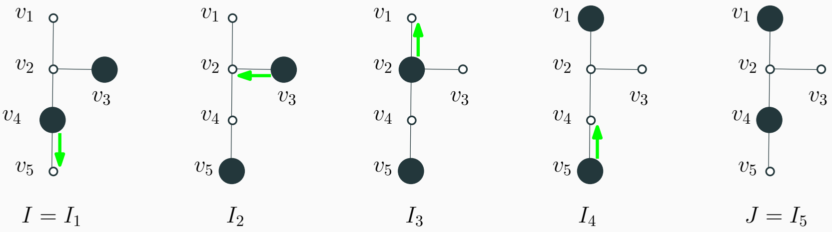
Theorem

When $\max\{|I \cap N_G(v)|, |J \cap N_G(v)|\} = 0$, *one can construct a (shortest)* TS-*sequence using* M^* *token-slides between* I *and* J, *where* $M^* = \min_{\textit{target assignment } f} \sum_{w \in I} \mathrm{dist}_G(w, f(w))$. *Moreover, this construction takes* $O(|V(G)|^2)$ *time.*

Hint: The Key Lemma allows us to pick a "good" target assignment, and the above observation tells us which token should be moved first.

Detour

We say that a TS-sequence S makes detour over an edge $e = xy \in E(G)$ if S at some time moves a token from x to y, and at some other time moves a token from y to x.



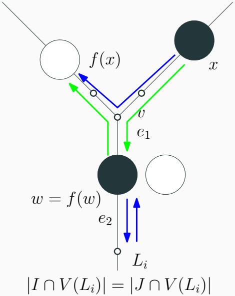
S makes detour over $e = v_4v_5$

Case 2: $\max\{|I \cap N_G(v)|, |J \cap N_G(v)|\} \le 1$

Special Case

- w and $f(w)$ are both placed in $N_G(v) \cap V(L_i)$;
- the number of I-tokens and J-tokens in L_i are equal.

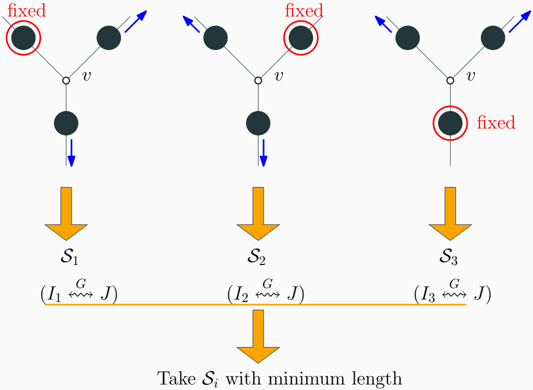
In this case, any TS-sequence must (at least) make detour over either e_1 or e_2.



- To handle this case, simply move both w and $f(w)$ to v. The problem now reduces to **Case 1**.
- This is not true when each leg of G contains the same number of I-tokens and J-tokens. However, this case is easy and can be handled separately.
- When the above case does not happen, slightly modify the instance to reduce to **Case 1**.

Case 3: $\max\{|I \cap N_G(v)|, |J \cap N_G(v)|\} \geq 2$

We consider only the case $|I \cap N_G(v)| \geq 2$ and $|J \cap N_G(v)| \leq 1$. Other cases are similar.



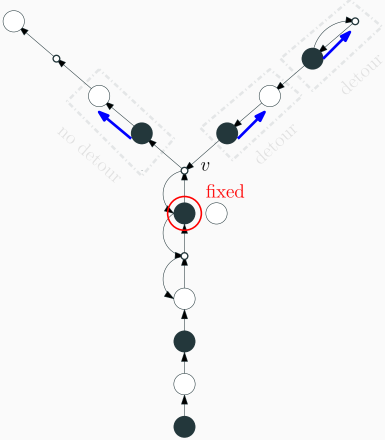
- For any TS-sequence S, exactly one of the $d = \deg_G(v)$ situations (as in the above example) must happen.
- Applying the above trick (regardless of J-tokens) reduces the problem to known cases (either **Case 1** or **Case 2**).

Case 3: $\max\{|I \cap N_G(v)|, |J \cap N_G(v)|\} \geq 2$

Issue

We don't know exactly how many detours the constructed sequence S performs.

Involve the following directed auxiliary graph $A(G, I, J)$.



- $V(A(G, I, J)) = V(G)$; and
- $E(A(G, I, J)) = \left\{(x, y) : xy \in E(G) \text{ and } \left|I \cap V(G^x_y)\right| \leq \left|J \cap V(G^x_y)\right|\right\}$,

where G^x_y is the subtree induced by y and its descendants when regarding x as root.

Case 3: $\max\{|I \cap N_G(v)|, |J \cap N_G(v)|\} \geq 2$

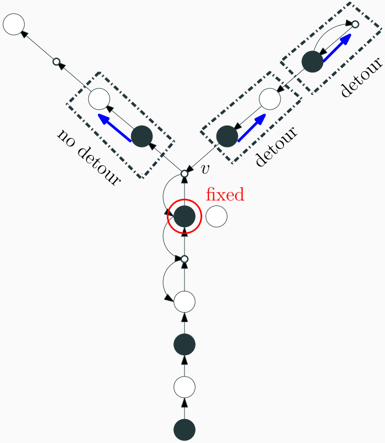
Issue

We don't know exactly how many detours the constructed sequence S performs.

Involve the following directed auxiliary graph $A(G, I, J)$.

- $V(A(G,I,J)) = V(G)$; and
- $E(A(G,I,J)) = \left\{(x,y) : xy \in E(G) \text{ and } \left|I \cap V(G_y^x)\right| \leq \left|J \cap V(G_y^x)\right|\right\}$,

where G_y^x is the subtree induced by y and its descendants when regarding x as root.



Conclusion

Conclusion

- We provided a $O(n^2)$-time algorithm for solving Shortest Sliding Token for spiders on n vertices.
- A shortest TS-sequence is explicitly constructed, along with the number of detours it makes.
- Our algorithm is optimal in the number of token-slides, as there exists a TS-sequence having $\Omega(n^2)$ token-slides (see Demaine et al. 2015).

Future Work

Extend the framework in order to improve Sugimori-san's algorithm for Shortest Sliding Token for trees.

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. DOI: 10.1016/j.tcs.2015.07.037.

Hearn, Robert A. and Erik D. Demaine (2005). "PSPACE-Completeness of Sliding-Block Puzzles and Other Problems through the Nondeterministic Constraint Logic Model of Computation". In: *Theoretical Computer Science* 343.1-2, pp. 72–96. DOI: 10.1016/j.tcs.2005.05.008.

Heuvel, Jan van den (2013). "The Complexity of Change". In: *Surveys in Combinatorics*. Vol. 409. London Mathematical Society Lecture Note Series. Cambridge University Press, pp. 127–160. DOI: 10.1017/CBO9781139506748.005.

Kamiński, Marcin, Paul Medvedev, and Martin Milanič (2012). "Complexity of independent set reconfigurability problems". In: *Theoretical Computer Science* 439, pp. 9–15. DOI: 10.1016/j.tcs.2012.03.004.

Nishimura, Naomi (2018). "Introduction to Reconfiguration". In: *Algorithms* 11.4. (article 52). DOI: 10.3390/a11040052.

Yamada, Takeshi and Ryuhei Uehara (2016). "Shortest reconfiguration of sliding tokens on a caterpillar". In: *Proceedings of WALCOM 2016*. Ed. by Mohammad Kaykobad and Rossella Petreschi. Vol. 9627. LNCS. Springer, pp. 236–248. DOI: 10.1007/978-3-319-30139-6_19.