Reconfiguring Independent Sets in Graphs

A Brief Introduction

Duc A. Hoang

Kyushu Institute of Technology, Japan
hoanganhduc@ces.kyutech.ac.jp

April 07, 2021

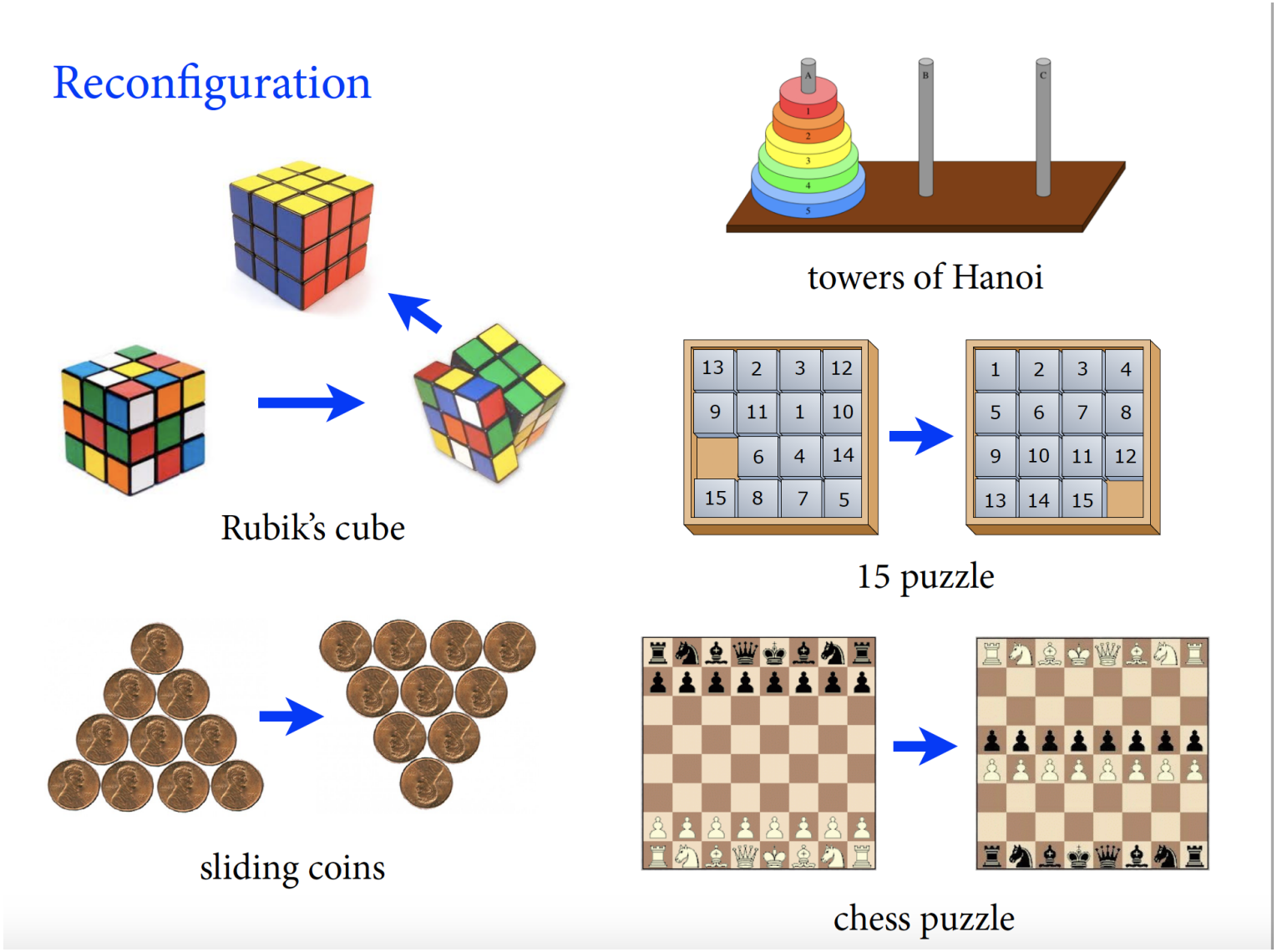
About Duc A. Hoang

- **Nationality:** Vietnamese. (My full name in Vietnamese is Hoàng Anh Đức.)
- **Education:**
  - **B.Math** @ VNU-HUS, Hanoi, Vietnam (2008–2013).
  - **M.S.** and **Ph.D.** @ JAIST, Ishikawa, Japan (2013–2018), supervised by Prof. Ryuhei Uehara.
- **Employment:**
  - **Postdoc** @ Kyutech, Fukuoka, Japan (2019–present), supervised by Prof. Toshiki Saitoh.
- **Research Interests:** Graph Algorithms, Combinatorial Reconfiguration.

Outline

Outline

Examples of Reconfiguration



© Anna Lubiw, in Part I of her tutorial "Reconfiguration and geometry" at CoRe2019.

We are given ...

The Input

- A description of *configurations*.
- A pre-defined *reconfiguration rule*.

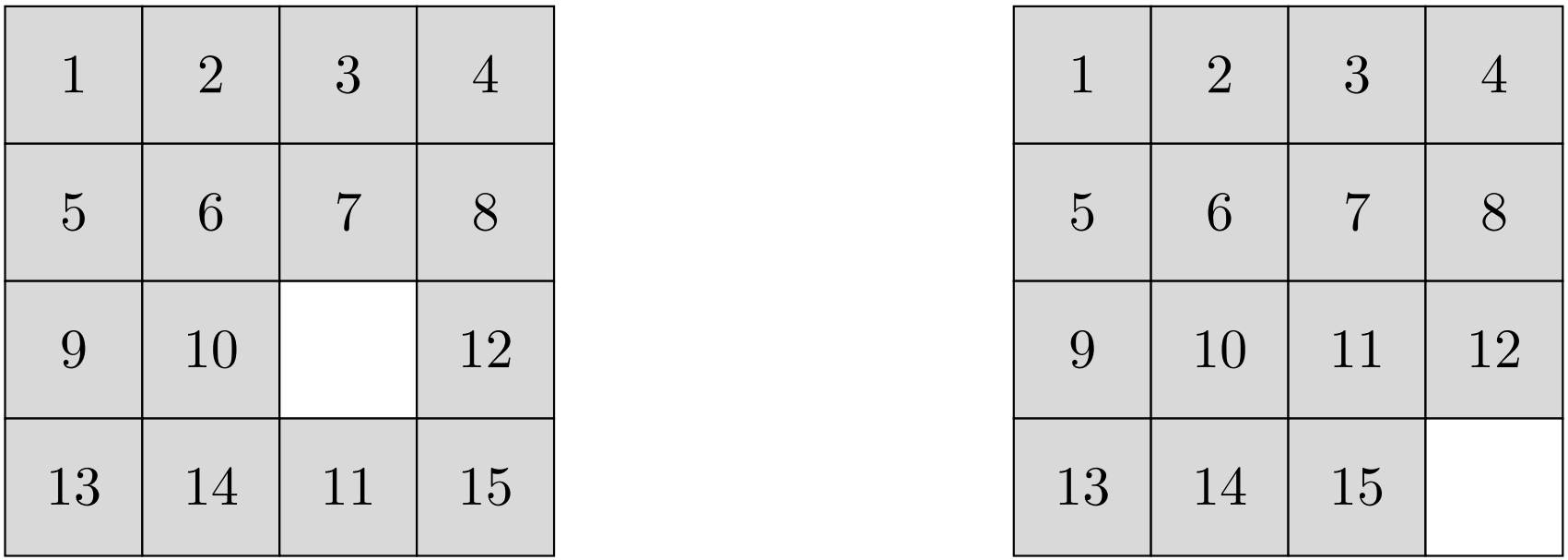
Note

- The *whole set of configurations* is *not* part of the input.
- Each (*valid*) *reconfiguration step* involves transforming one configuration (into another) by applying the given reconfiguration rule exactly *once*. A sequence of valid reconfiguration steps is called a *reconfiguration sequence*.
- There is a *polynomial-time algorithm* to check
  - whether a given object is actually a configuration.
  - given two configurations, whether one can be obtained from the other by a single valid reconfiguration step.

We are given ...

The Input

- A description of *configurations*.

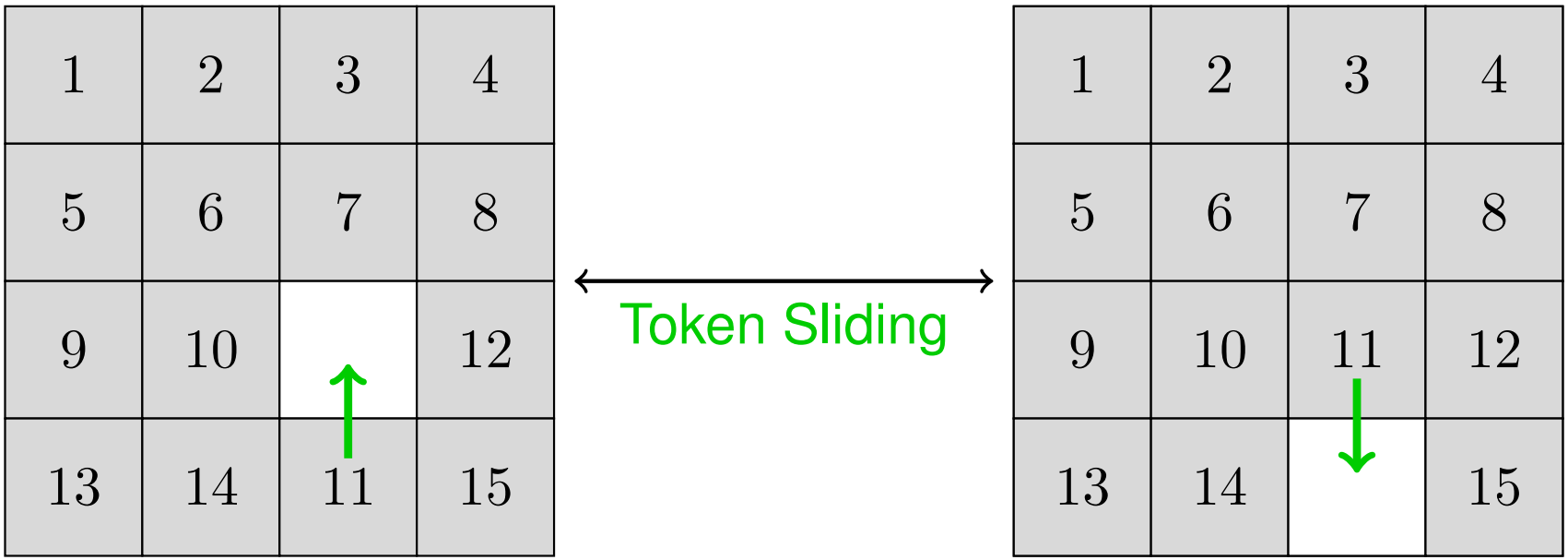


15-PUZZLE [Story 1879].

We are given ...

The Input

- A description of *configurations*.
- A pre-defined *reconfiguration rule*.

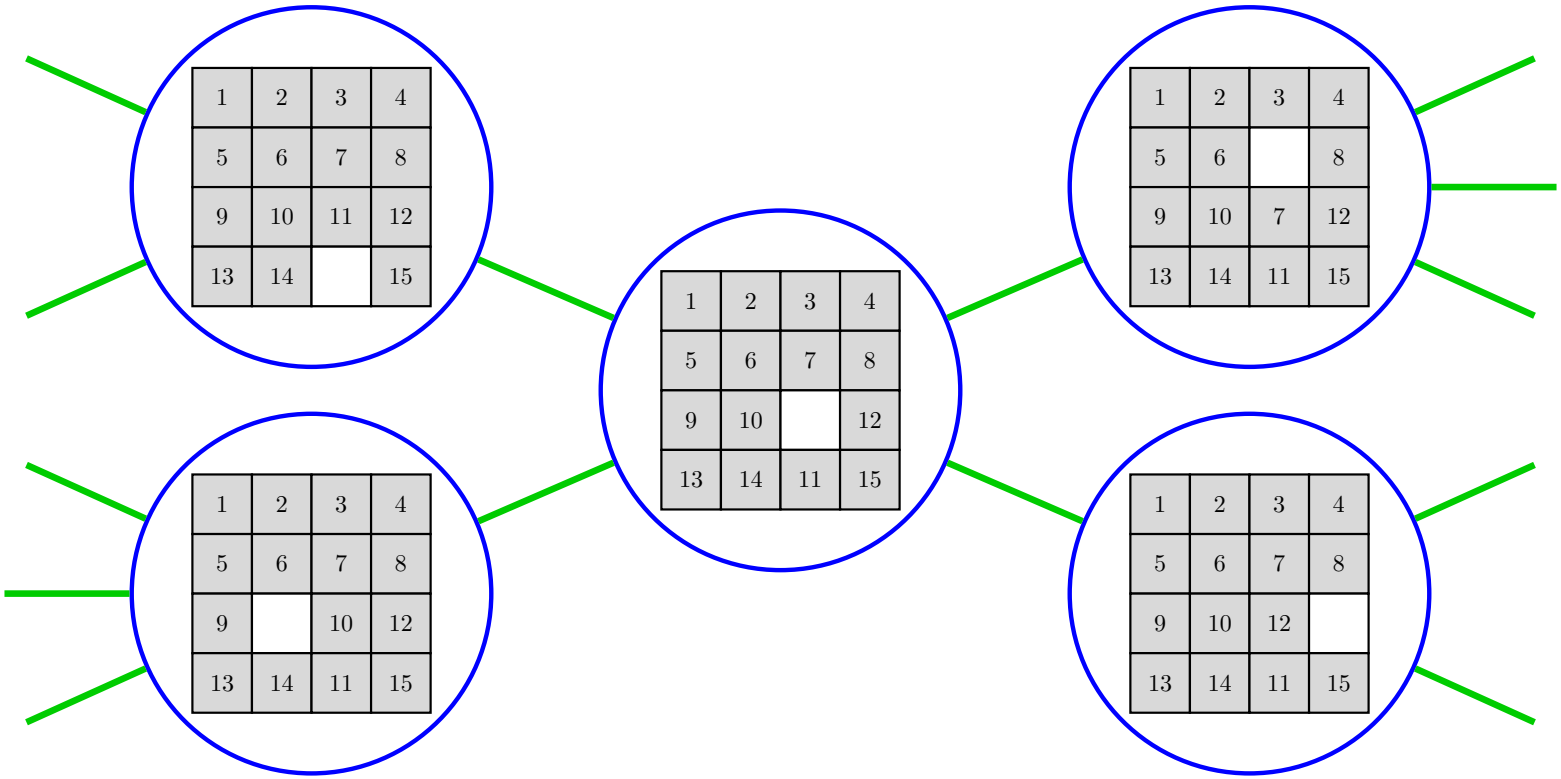


15-PUZZLE [Story 1879].

We are given ...

Reconfiguration Graphs

- Each *configuration* corresponds to a *vertex/node*.
- The *reconfiguration rule* defines *adjacent nodes*.



Part of the reconfiguration graph for 15-PUZZLE.

Given a reconfiguration graph, we may ask ...

- REACHABILITY: Is there a *path* between two given nodes?
- SHORTEST TRANSFORMATION: If REACHABILITY is yes, can we find a *shortest path*?
- BOUNDED TRANSFORMATION: Is there a *path of length at most some given positive integer* ℓ between two given nodes?
- CONNECTIVITY: Is there a *path* between *any* two given nodes?
- DIAMETER: Is the *maximum distance* between any two nodes bounded?
- and so on.

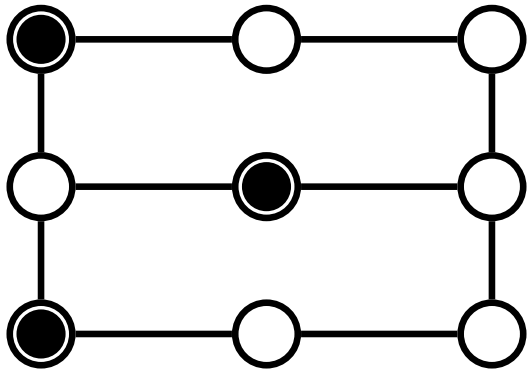
Note: Equivalence of Notations

A *path* between two nodes of the reconfiguration graphs is also a *reconfiguration sequence* between the corresponding configurations.

Outline

Remind: Independent Set and Vertex Cover

An *independent set* of a graph $G = (V, E)$ is a set I of vertices such that for any pair $u, v \in I \subseteq V$, $uv \notin E$.
A *vertex cover* of G is a set J of vertices such that for every edge $uv \in E$, the set $\{u, v\} \cap J$ is not empty.



Example of an independent set whose members are depicted with black tokens.

Lemma 1: (folklore)

I is an independent set $\Leftrightarrow$ $V \setminus I$ is a vertex cover.

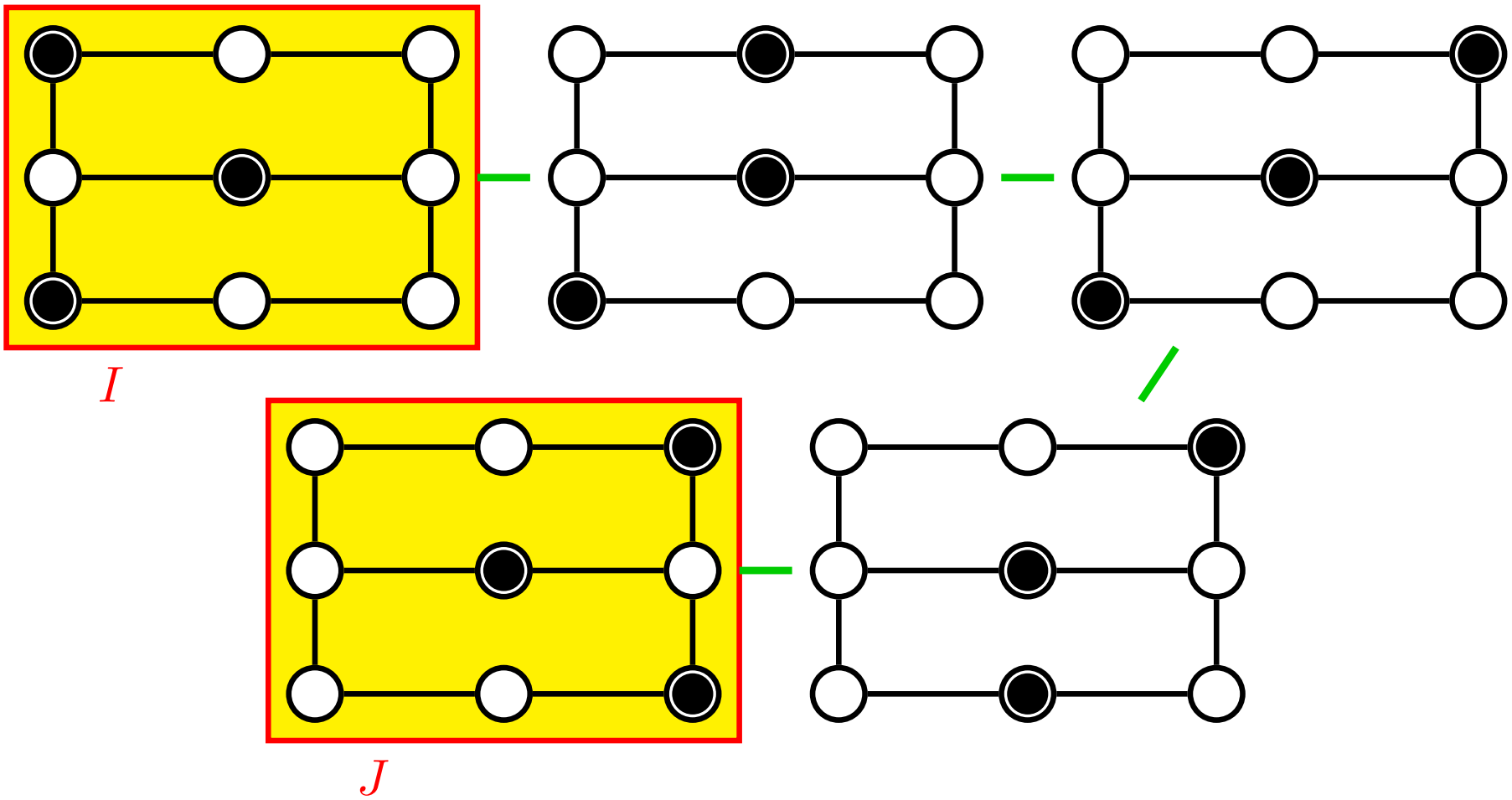
Outline

Reconfiguration Rules

- For any reconfiguration rule R, a *valid* R-move transforms one independent set into another.
- Three main *reconfiguration rules* have been studied:
  - Token Sliding (TS).
  - Token Jumping (TJ).
  - Token Addition/Removal (TAR(u)).

Token Sliding (TS)

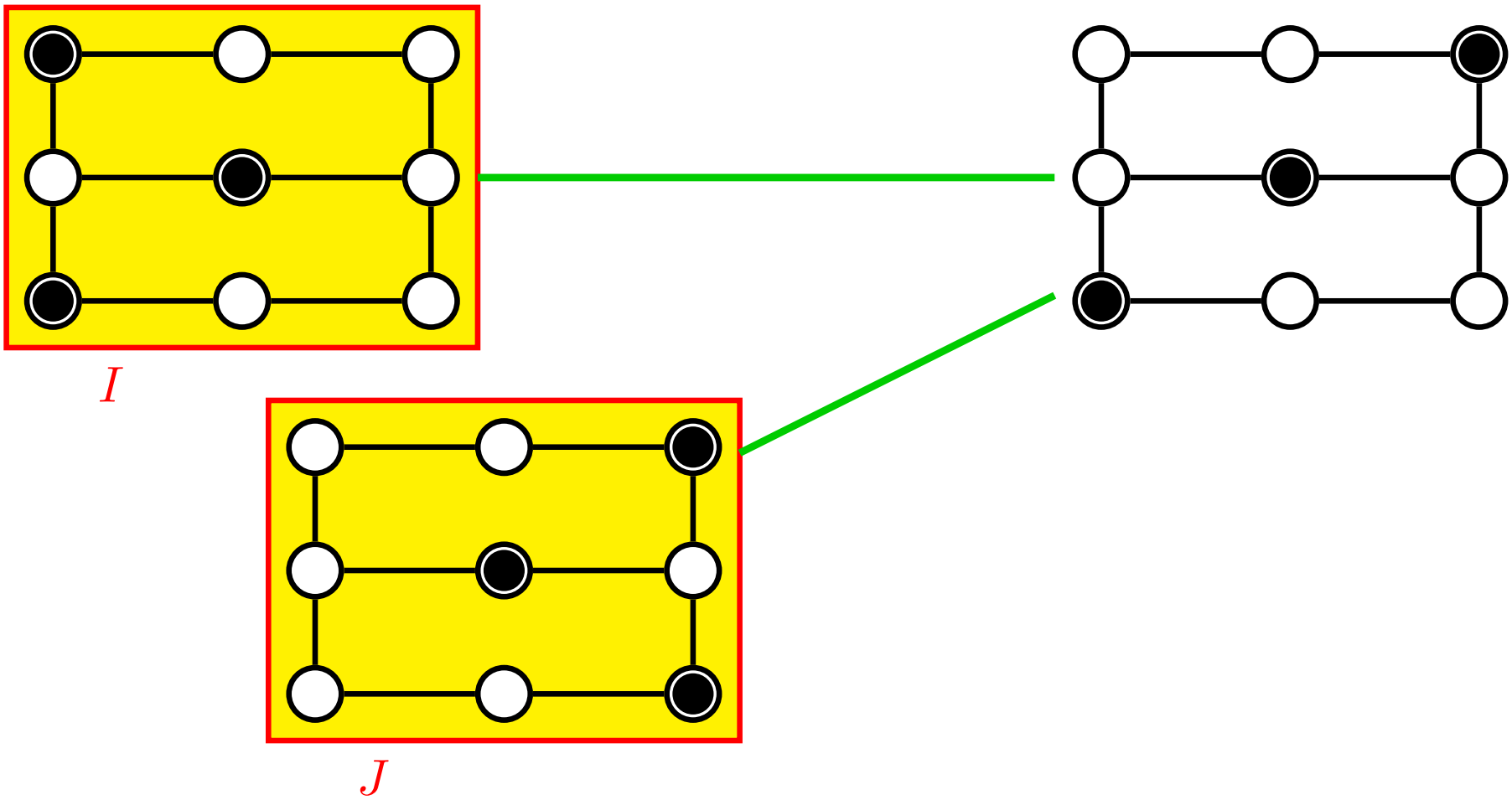
- First introduced in [Hearn and Demaine 2005].
- Each TS-move involves removing a member *and* adding one of its neighbors.



A sequence of valid TS-moves between I into J.

Token Jumping (TJ)

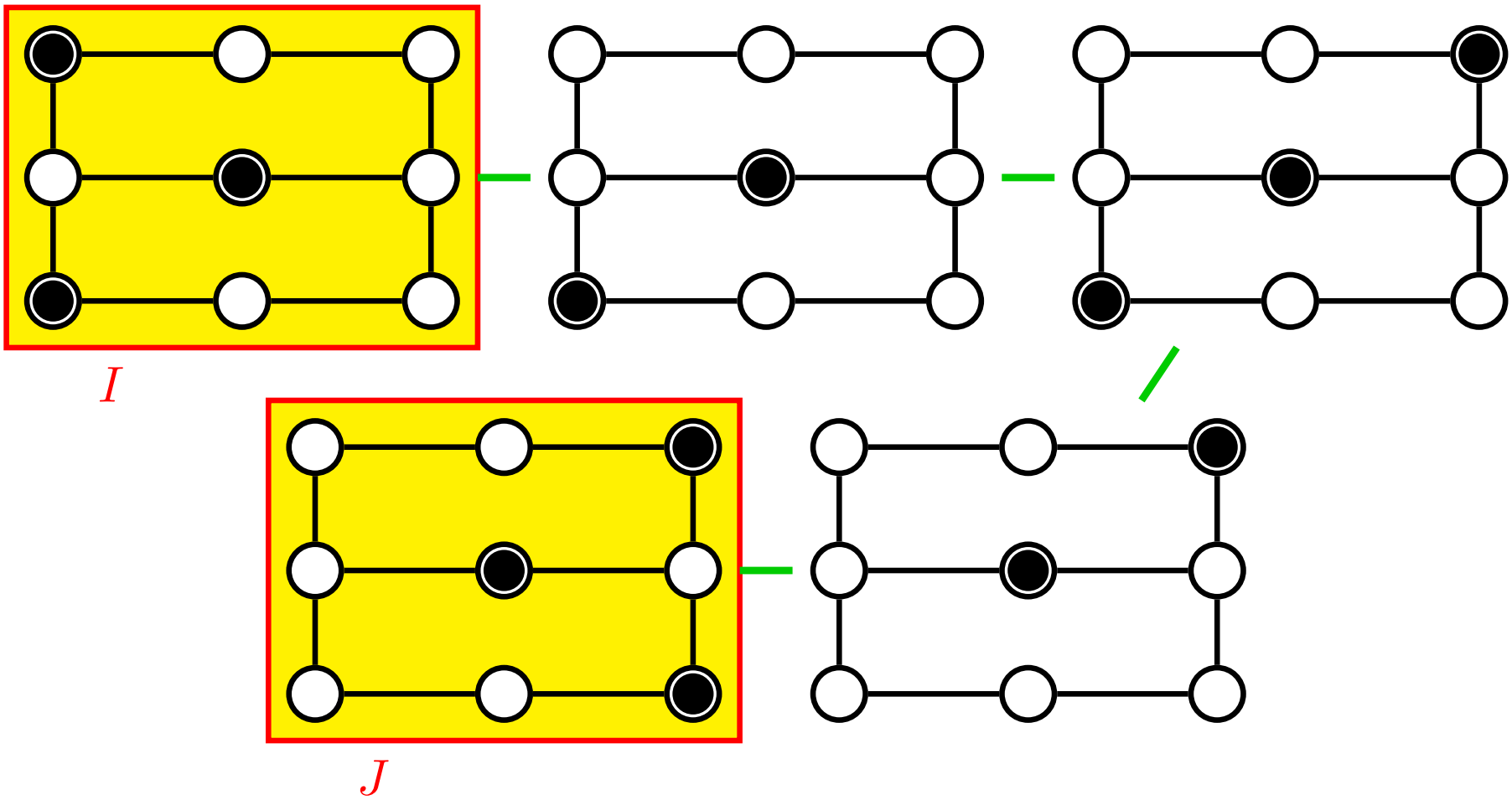
- First introduced in [Kamiński et al. 2012].
- Each TJ-move involves removing a member *and* adding one non-member vertex.



A sequence of valid TJ-moves between I into J.

Token Addition/Removal (TAR(u))

- First introduced in [Ito et al. 2011].
- Each TAR(u)-move involves adding a non-member *or* removing a member s.t. each set has $\geq u$ *members*.



A sequence of valid TAR(2)-moves between I into J.

Outline

Known Research Directions

- Classic Complexity.
- Parameterized Complexity.
- Structure of Reconfiguration Graphs.
- Optimization.
- Distributed Reconfiguration.

Classic Complexity

Under any of the described rules, one may ask

- REACHABILITY [main question in this talk]
- SHORTEST TRANSFORMATION
- BOUNDED TRANSFORMATION
- CONNECTIVITY
- DIAMETER
- and so on.

Note

The *classic complexity* results on reconfiguring independent sets and vertex covers *can be interchanged*. See Section 4 of [Nishimura 2018] for more details.

Parameterized Complexity

- Given one or more *parameter(s)*, in addition to the original input:
  - A bound s on the *size* of the independent sets;
  - The *length* ℓ of a reconfiguration sequence;
  - The *treewidth* tw of the input graph;
  - The *maximum degree* Δ of the input graph;
  - and so on.
- A problem is *fixed-parameter tractable (*FPT*)* if it can be solved in *poly(original input size)* $\star$ f*(parameter)* time, where f is a computable function.
- Complexity hierarchy:
  FPT $\subseteq$ W[1] $\subseteq$ W[2] $\subseteq \cdots \subseteq$ W[P] $\subseteq$ XP.
- FPT (W[1]*-hard*) is the *parameterized complexity analog* of P (NP*-hard*).

Note

See Section 5 of [Nishimura 2018] for more details.

Structure of Reconfiguration Graphs

Under any of the described rules, one may ask

- Which *graph* can be a *reconfiguration graph* for INDEPENDENT SET RECONFIGURATION?
- Which *reconfiguration graph* for INDEPENDENT SET RECONFIGURATION is *isomorphic* to the input graph itself?
- Which *reconfiguration graph* for INDEPENDENT SET RECONFIGURATION are *in the same class* with the input graph itself?
- Can we find a graph that is *not* a reconfiguration graph for INDEPENDENT SET RECONFIGURATION in *any* input graph?
- and so on.

Note

Similar questions have been studied extensively for DOMINATING SET RECONFIGURATION and VERTEX-COLOR RECONFIGURATION [Mynhardt and Nasserasr 2019].

Optimization

Under any of the described rules, one may ask

- If I can be reconfigured into J, can we find a reconfiguration sequence between them with *smallest* number of moves? ($\equiv$ SHORTEST TRANSFORMATION)
- If p tokens can be moved *simultaneously*, what is the *smallest* value of p such that one can *always* transform one independent set into another? (e.g., see [de Berg et al. 2018].)
- **(Only under** TAR(u)**)** Starting from an independent set I, what is the *largest* independent set one can reach? (e.g., see [Ito et al. 2019].)
- and so on.

Distributed Reconfiguration

- First introduced in [Censor-Hillel and Rabie 2019].
- Their work employs the *LOCAL model of computation*.
  - Simple, undirected, unweighted, n-node graph $G = (V, E)$.
  - The algorithms work in *synchronous rounds*.
  - *Per round*, each node can
    - *send* a message to its neighbors;
    - *receive* messages sent from its neighbors;
    - *do* some computation.
  - Each node knows whether it is *selected*. The set of selected nodes forms a *maximal* independent set.
- Reconfiguration rule: TAR.
  - *No* lower bound on the number of tokens (i.e., selected nodes).
  - In each TAR-step, *both* adding and removing *multiple tokens* are allowed.
  - *No* two neighbors change their membership status at the same step: if J is obtained from I by a single TAR-step, their symmetric difference $I \Delta J = (I \setminus J) \cup (J \setminus I)$ must be independent.

Outline

Reconfiguring Independent Sets under Token Sliding

In this talk, we shall mention the following questions:

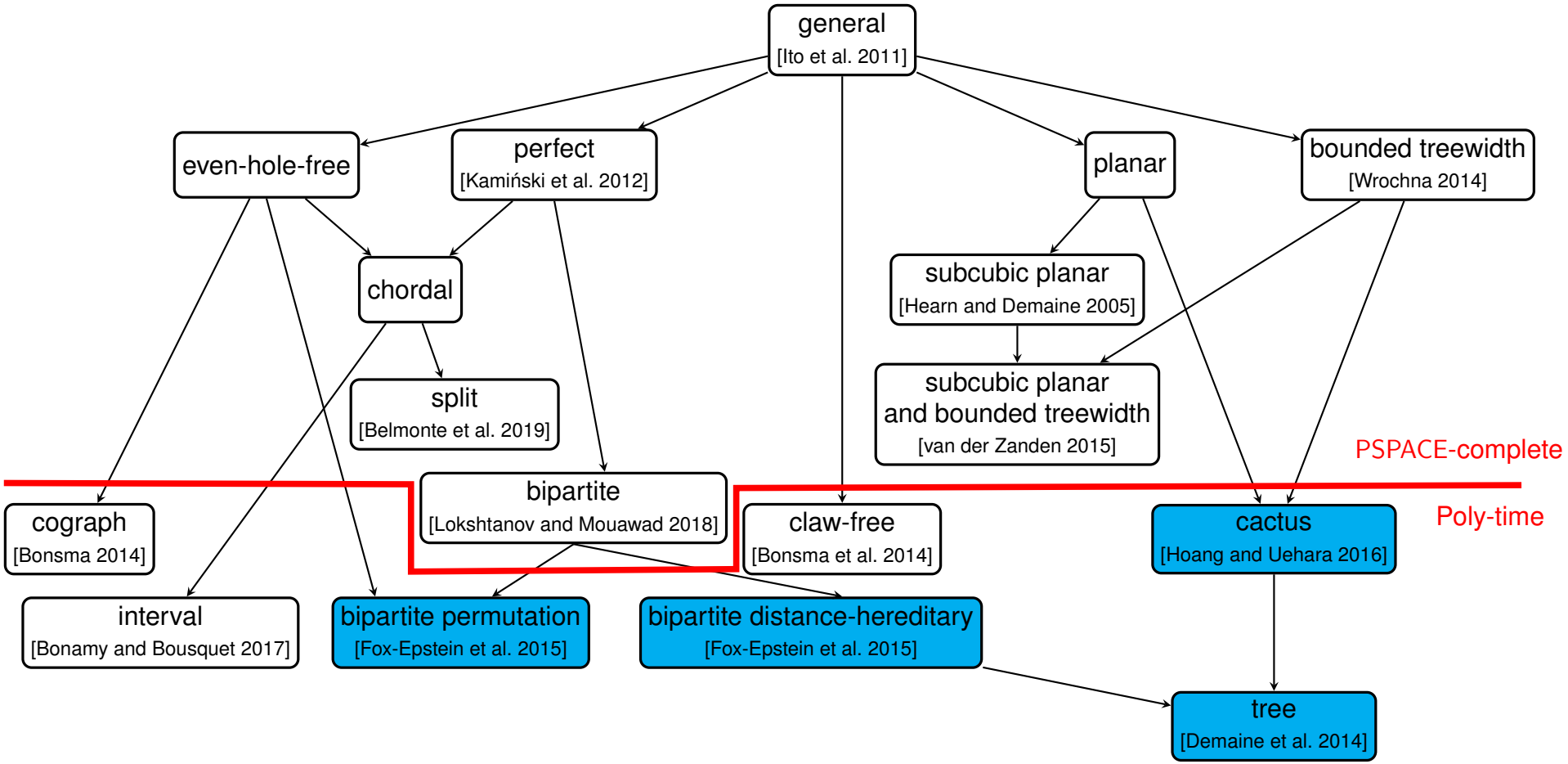
REACHABILITY

- **Input:** (G, I, J).
- **Question:** Is there a *sequence* of valid TS-moves that reconfigures I into J in G?

SHORTEST TRANSFORMATION

- **Input:** A YES-instance (G, I, J) of REACHABILITY.
- **Question:** Can we find a *shortest sequence* of valid TS-moves that reconfigures I into J in G?

Reconfiguring Independent Sets under Token Sliding



Complexity of REACHABILITY for some graphs. Our results are marked with the cyan color.

See [Nishimura 2018] for more information on problems under TJ and TAR(u).

Reconfiguring Independent Sets under Token Sliding

Our Approach

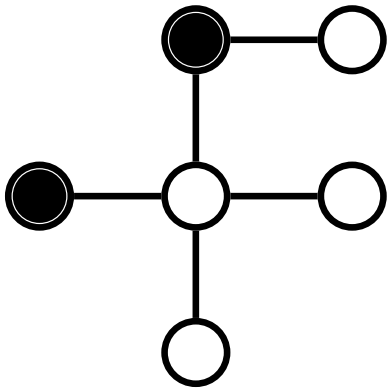
- Characterize *forbidden structures* that obstruct the reconfiguration from one independent set into another.
- Design *reconfiguration sequence* in polynomial time when those forbidden structures do not exist.

Reconfiguring Independent Sets under Token Sliding

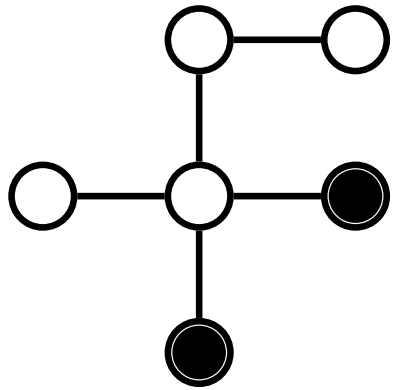
Our Approach

- Characterize *forbidden structures* that obstruct the reconfiguration from one independent set into another.
- Design *reconfiguration sequence* in polynomial time when those forbidden structures do not exist.

For example, in *trees*, the forbidden structure is *the tokens that cannot be moved at all* (called the *rigid tokens*).



I



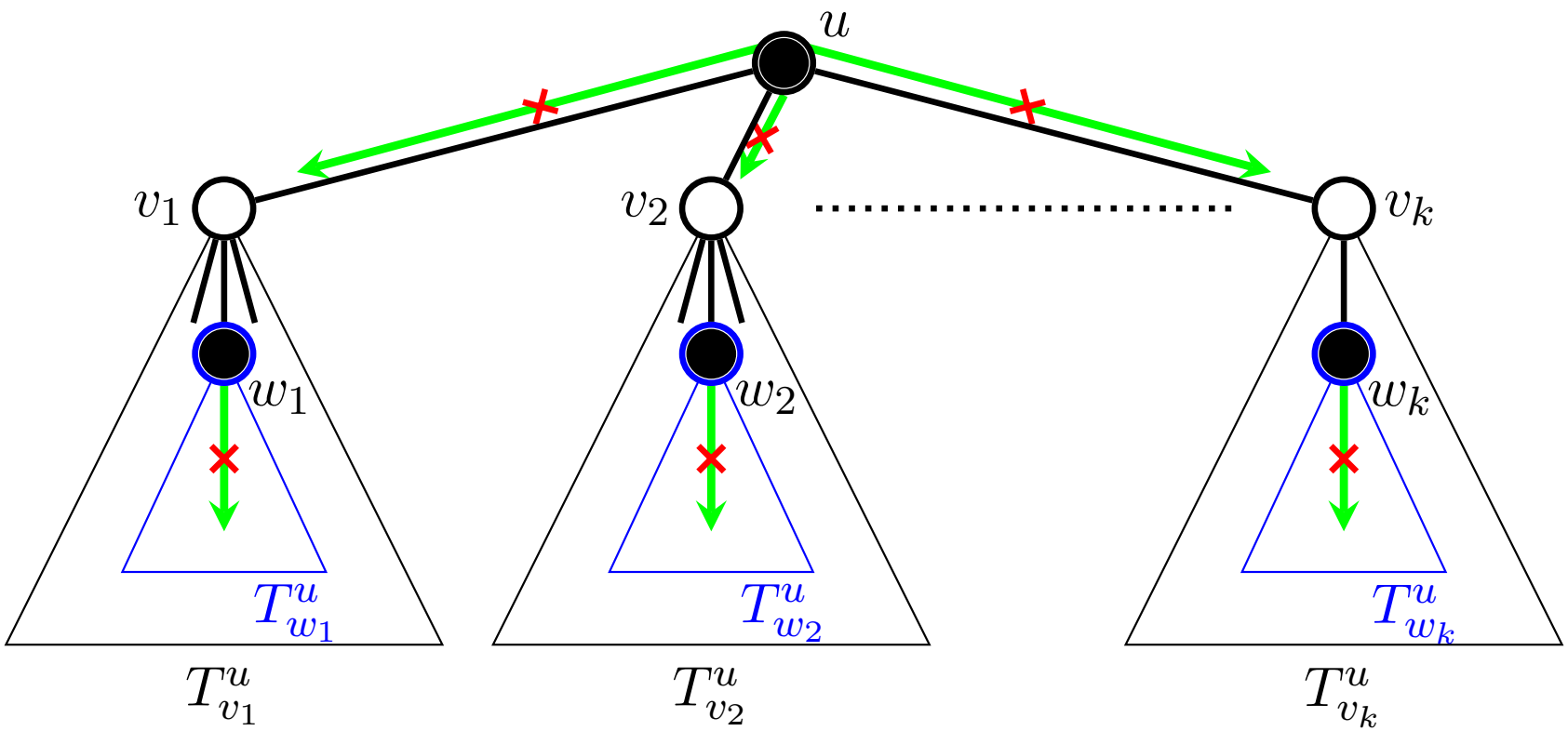
J

I can not be reconfigured into J, and vice versa. They have *different rigid tokens*: I contains *no rigid tokens*, while J has *two*.

Reconfiguring Independent Sets under Token Sliding

Lemma 2: [Demaine et al. 2014]

In a tree T, one can decide in $O(|V(T)|)$ time if a token is rigid.



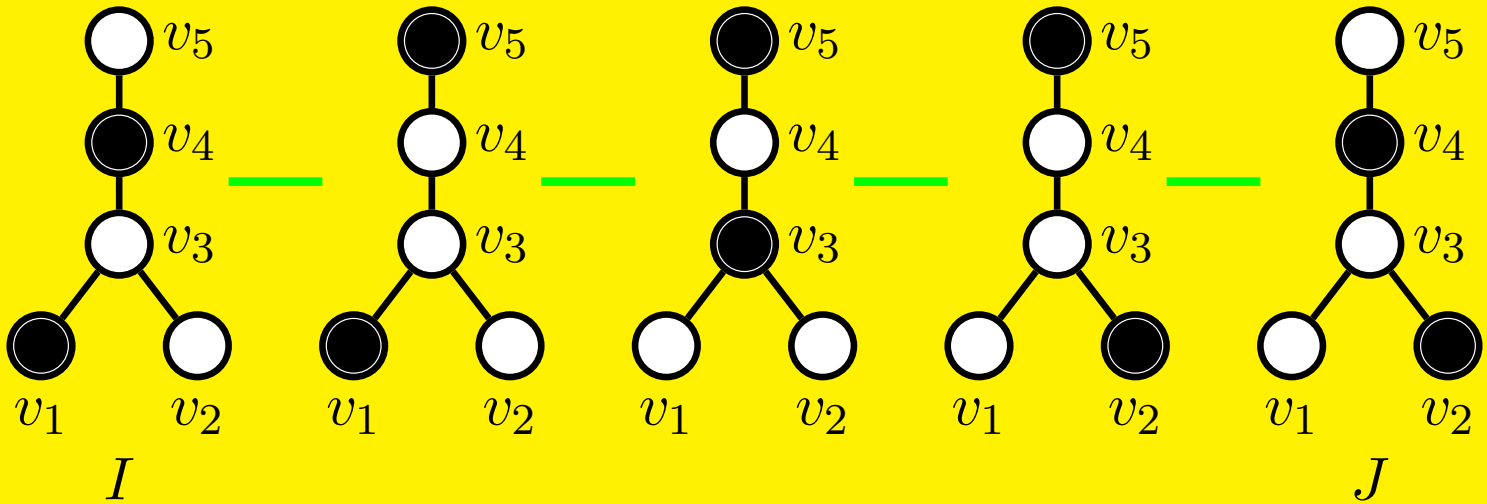
Hint: The token on u is *rigid in T* $\Leftrightarrow$ For each neighbor v_i of u ($i \in \{1, 2, \ldots, k\}$), there exists a neighbor w_i of v_i whose corresponding token is *rigid in the tree $T^u_{w_i}$*.

Reconfiguring Independent Sets under Token Sliding

Lemma 3: [Demaine et al. 2014]

Suppose that there are no rigid tokens in two independent sets I, J of a tree T. Then, a sequence of TS*-moves between them can be constructed in $O(|V(T)|^2)$ time.*

Note: Detours



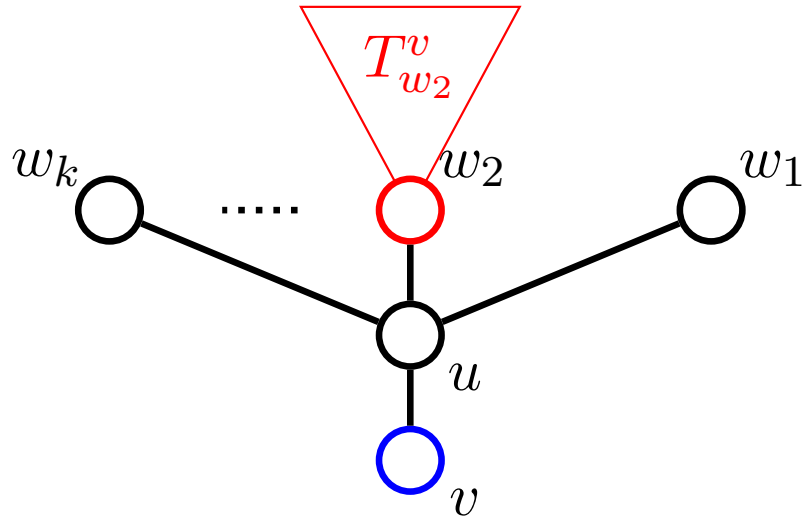
The token on v_4 was already placed at its final destination, but needs to "move away" to allow the token on v_1 to move to v_2.

Reconfiguring Independent Sets under Token Sliding

Lemma 3: [Demaine et al. 2014]

Suppose that there are no rigid tokens in two independent sets I, J of a tree T. Then, a sequence of TS*-moves between them can be constructed in $O(|V(T)|^2)$ time.*

A leaf v in a tree is called *safe* if removing v and its unique neighbor u results at most *one* component which has more than one vertex.



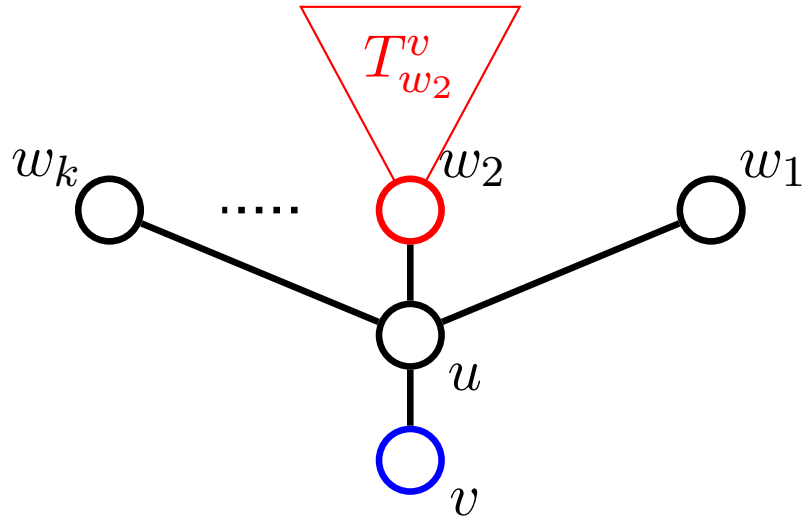
v is safe: only $T' = T^v_{w_2}$ has more than one vertex.

Reconfiguring Independent Sets under Token Sliding

Lemma 3: [Demaine et al. 2014]

Suppose that there are no rigid tokens in two independent sets I, J of a tree T. Then, a sequence of TS*-moves between them can be constructed in $O(|V(T)|^2)$ time.*

Hint: We construct a sequence of TS-moves from each of I and J to some *intermediate set* $I^\star$. To reconfigure I into J, we first reconfigure I into $I^\star$, then $I^\star$ into J.



v is safe: only $T' = T^v_{w_2}$ has more than one vertex.

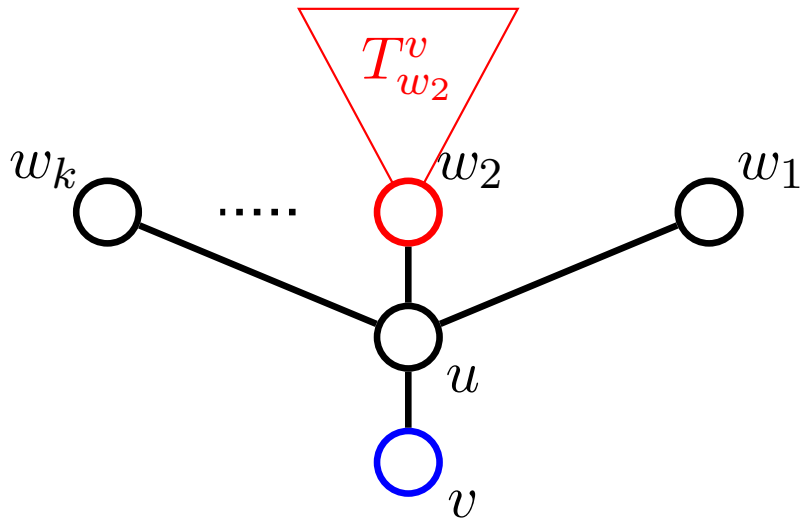
Reconfiguring Independent Sets under Token Sliding

Lemma 3: [Demaine et al. 2014]

Suppose that there are no rigid tokens in two independent sets I, J of a tree T. Then, a sequence of TS*-moves between them can be constructed in $O(|V(T)|^2)$ time.*

Hint: We construct a sequence of TS-moves from each of I and J to some *intermediate set* $I^\star$. To reconfigure I into J, we first reconfigure I into $I^\star$, then $I^\star$ into J.

- *Pick* a safe leaf v in T.
- For each of I and J, *move* a token to v. Then, *remove* v and its neighbor u.
- *Repeat* with the only resulting component T' having more than one vertex (if it exists).



v is safe: only $T' = T^v_{w_2}$ has more than one vertex.

Reconfiguring Independent Sets under Token Sliding

SHORTEST TRANSFORMATION

Can we find a *shortest* sequence of TS-moves (if it exists)?

Reconfiguring Independent Sets under Token Sliding

SHORTEST TRANSFORMATION

Can we find a *shortest* sequence of TS-moves (if it exists)?

In general,

Theorem 4: [Kamiński et al. 2012]

Deciding if there is a sequence of at most ℓ TS-moves between two given independent sets in a perfect graph is NP*-hard.*

Reconfiguring Independent Sets under Token Sliding

SHORTEST TRANSFORMATION

Can we find a *shortest* sequence of TS-moves (if it exists)?

In general,

Theorem 4: [Kamiński et al. 2012]

Deciding if there is a sequence of at most ℓ TS-moves between two given independent sets in a perfect graph is NP-hard.

On the other hand, it can be solved in polynomial time for some sub-classes of trees: *caterpillar* [Yamada and Uehara 2016] and *spider tree* [Hoang et al. 2018].
Recently, it has been shown that

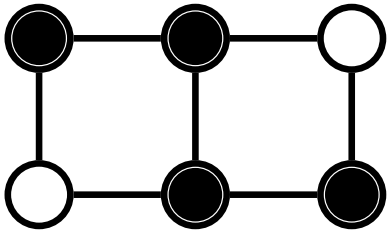
Theorem 5: [Sugimori 2019]

One can find in $O(n^5)$ time a shortest sequence of TS-moves (if it exists) between two independent sets in a n-vertex tree.

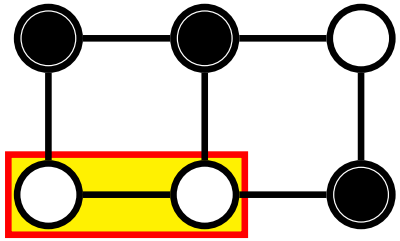
Outline

Remind: k-Path Vertex Cover

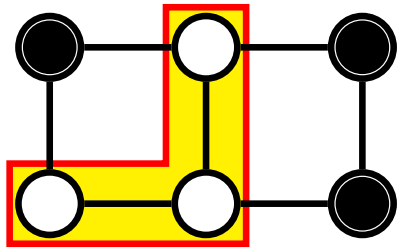
A *k-path vertex cover* (k-PVC, first introduced in [Brešar et al. 2011]) of a graph $G = (V, E)$ is a set K of vertices such that for any path P in G on k vertices, the set $K \cap V(P)$ is not empty. A *2-path vertex cover* is indeed a *vertex cover*.



A 3-PVC
Also a 2-PVC

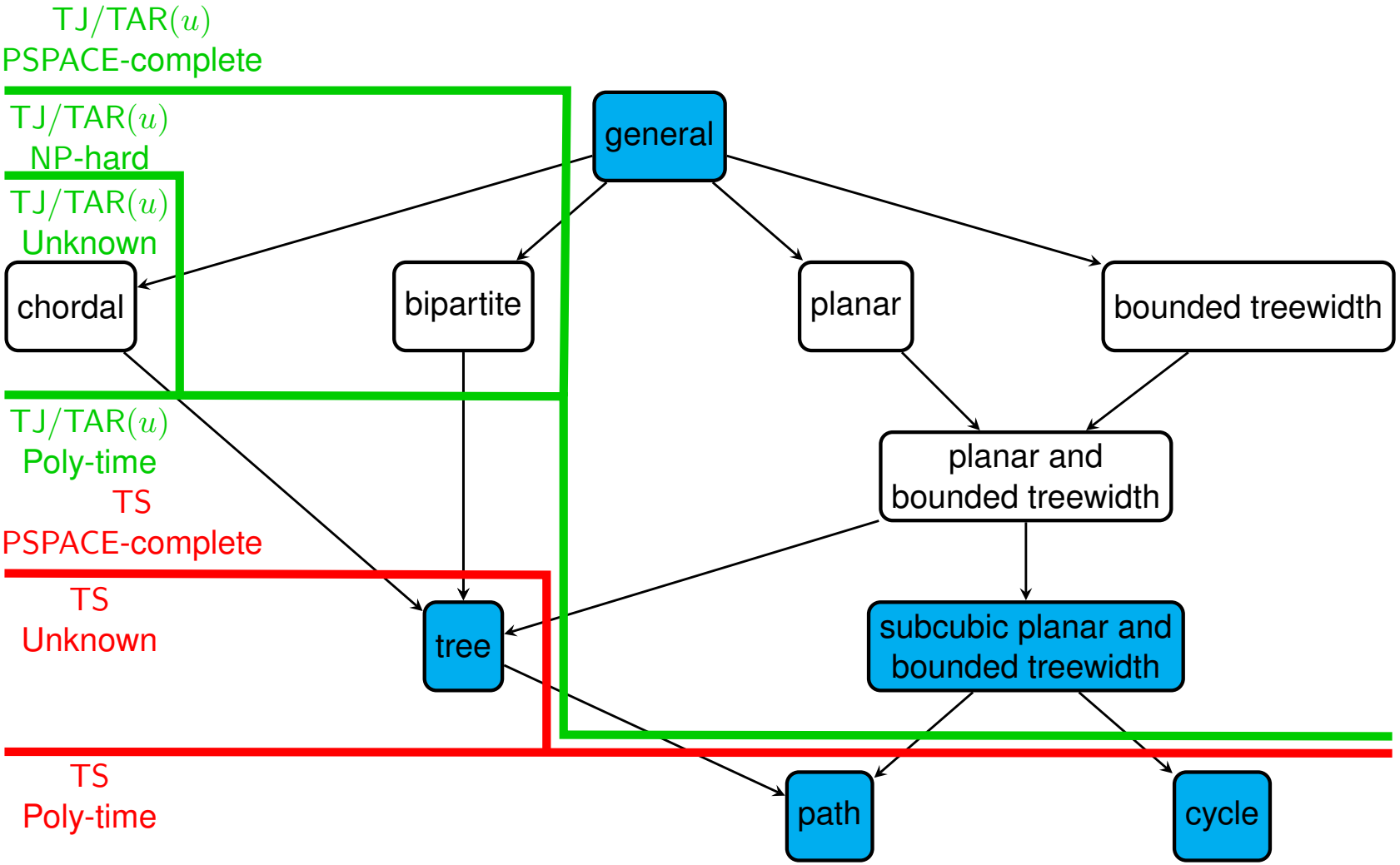


A 3-PVC
Not a 2-PVC



Not a 3-PVC
Not a 2-PVC

Reconfiguring k-Path Vertex Covers in Graphs



Complexity of REACHABILITY for some graphs [Hoang et al. 2020]. The graphs we considered are marked with the cyan color. Our reduction from MINIMUM VERTEX COVER RECONFIGURATION for *general graphs* implies the hardness on many other graphs.

Reconfiguring k-Path Vertex Covers in Graphs

Theorem 6: [Hoang et al. 2020]

REACHABILITY *for* k-PATH VERTEX COVER RECONFIGURATION *is* PSPACE*-hard under each of* TS, TJ, *and* TAR(u).

Hint: Reduction from MINIMUM VERTEX COVER RECONFIGURATION – a known PSPACE-complete problem.

- Each instance (G, I, J) of MINIMUM VERTEX COVER RECONFIGURATION corresponds to an instance (G', I, J) of MINIMUM k-PATH VERTEX COVER RECONFIGURATION.
- G' is constructed from G by attaching to each vertex of G a new path on $\lfloor (k-1)/2 \rfloor$ vertices.

Reconfiguring k-Path Vertex Covers in Graphs

Theorem 6: [Hoang et al. 2020]

REACHABILITY *for* k-PATH VERTEX COVER RECONFIGURATION *is* PSPACE*-hard under each of* TS, TJ, *and* TAR(u).

Hint: Reduction from MINIMUM VERTEX COVER RECONFIGURATION – a known PSPACE-complete problem.

- Each instance (G, I, J) of MINIMUM VERTEX COVER RECONFIGURATION corresponds to an instance (G', I, J) of MINIMUM k-PATH VERTEX COVER RECONFIGURATION.
- G' is constructed from G by attaching to each vertex of G a new path on $\lfloor (k-1)/2 \rfloor$ vertices.

Theorem 7: [Hoang et al. 2020]

For any two k-PVCs I, J of size s in a tree, one can always construct a sequence of TJ*-moves between them in linear time.*

Hint: Construct an *intermediate* k-PVC that is reachable from both I and J.

Outline

Graphs of Treewidth ≤ 2?

It is well-known that

Theorem 8: [Wrochna 2014]; [Wrochna 2018]

There is a positive integer c such that INDEPENDENT SET RECONFIGURATION *under any of* TS, TJ, *or* TAR(u) *is* PSPACE*-hard even in graphs of treewidth at most c.*

For $c = 1$, the problems can be solved in polynomial time in *trees* (see [Kamiński et al. 2012]; [Demaine et al. 2014]).

Open Question

For $c = 2$, is INDEPENDENT SET RECONFIGURATION under any of TS, TJ, or TAR(u) solvable in polynomial time?

The question remains *open* even for *outerplanar graphs*.

Reconfiguration Graph of Independent Sets?

Open Question

Which *graph* can be a *reconfiguration graph* for INDEPENDENT SET RECONFIGURATION?

- Alikhani and Fatehi [Alikhani and Fatehi 2017] initiated the study of this question under TAR(u) rule.
- How about TJ and TS rules?

Naturally, a first step may be to study the question on some *simple, restricted graphs*.

- Which *graph* can be a *reconfiguration graph* for INDEPENDENT SET RECONFIGURATION under TS on *trees*/*cycles*/*cliques*?

Reconfiguration Graph of Independent Sets?

A partial answer may be:

Theorem 9

The reconfiguration graph for INDEPENDENT SET RECONFIGURATION *under* TS

(a) *on cliques is clique;*

(b) *on trees is bipartite;*

(c) *on cycles is not bipartite.*

Hint:

- To see (b), suppose that J can be obtained from I by sliding a single token, say from u to v. *How do the tokens placed at vertices in* $I \setminus \{u, v\}$ *move?*
- To see (c), take C_5 and consider *only independent sets of size* 2. How does the corresponding reconfiguration graph look like?

Reconfiguration Graph of Independent Sets?

A partial answer may be:

Theorem 9

The reconfiguration graph for INDEPENDENT SET RECONFIGURATION *under* TS

(a) *on cliques is clique;*

(b) *on trees is bipartite;*

(c) *on cycles is not bipartite.*

Conjecture 10

The reconfiguration graph for INDEPENDENT SET RECONFIGURATION *under* TS

(a) *on trees is planar;*

(b) *on cycles is planar.*

Vertex Cover v.s. k-Path Vertex Cover?

Open Question

Intuitively, reconfiguring k-path vertex covers should be *"harder"* than reconfiguring vertex covers. Can we find any *graph class* to *confirm* this intuition?

Problems that we have not yet been able to solve in [Hoang et al. 2020].

- Reconfiguring k-path vertex covers on *chordal graphs* under TJ/TAR(u)?
  - Note that reconfiguring vertex covers on *chordal graphs* under TJ/TAR(u) is solvable in polynomial time [Kamiński et al. 2012].
- Reconfiguring k-path vertex covers on *trees* under TS?
  - Note that reconfiguring vertex covers on *trees* under TS is solvable in polynomial time [Demaine et al. 2014].

Naturally, one may start with $k = 3$.

Learn More About Reconfiguration

- Surveys:
  - 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. arXiv: 1312.2816.
  - Naomi Nishimura (2018). “Introduction to Reconfiguration”. In: *Algorithms* 11.4. (article 52). DOI: 10.3390/a11040052.
  - C.M. Mynhardt and S. Nasserasr (2019). “Reconfiguration of colourings and dominating sets in graphs”. In: *50 years of Combinatorics, Graph Theory, and Computing*. Ed. by Fan Chung et al. 1st. CRC Press, pp. 171–191. DOI: 10.1201/9780429280092-10. arXiv: 2003.05956.
- Web: `http://www.ecei.tohoku.ac.jp/alg/core/` and `http://reconf.wikidot.com/` (I occasionally maintain this site).
- Open Problems from CoRe2019: `https://pagesperso.g-scop.grenoble-inp.fr/~bousquen/CoRe_2019/CoRe_2019_Open_Problems.pdf`.

Bibliography

Hoang, Duc A., Akira Suzuki, and Tsuyoshi Yagita (2020). "Reconfiguring k-path vertex covers". In: *Proceedings of WALCOM 2020*. Ed. by M. Sohel Rahman, Kunihiko Sadakane, and Wing-Kin Sung. Vol. 12049. LNCS. Springer, pp. 133–145. DOI: 10.1007/978-3-030-39881-1_12. arXiv: 1911.03026.

Belmonte, Rémy, Eun Jung Kim, Michael Lampis, Valia Mitsou, Yota Otachi, and Florian Sikora (2019). "Token Sliding on Split Graphs". In: *Proceedings of STACS 2019*. Ed. by Rolf Niedermeier and Christophe Paul. Vol. 126. LIPIcs. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 13:1–13:7. DOI: 10.4230/LIPIcs.STACS.2019.13. arXiv: 1807.05322.

Bibliography (cont.)

Censor-Hillel, Keren and Mikaël Rabie (2019). “Distributed Reconfiguration of Maximal Independent Sets”. In: *Proceedings of ICALP 2019*. Ed. by Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi. Vol. 132. LIPIcs, 135:1–135:14. DOI: 10.4230/LIPIcs.ICALP.2019.135. arXiv: 1810.02106.

Ito, Takehiro, Haruka Mizuta, Naomi Nishimura, and Akira Suzuki (2019). “Incremental Optimization of Independent Sets under Reachability Constraints”. In: *Proceedings of COCOON 2019*. Ed. by Ding-Zhu Du, Zhenhua Duan, and Cong Tian. Vol. 11653. LNCS, pp. 313–324. DOI: 10.1007/978-3-030-26176-4_26. arXiv: 1804.09422.

Bibliography (cont.)


Mynhardt, C.M. and S. Nasserasr (2019). "Reconfiguration of colourings and dominating sets in graphs". In: *50 years of Combinatorics, Graph Theory, and Computing*. Ed. by Fan Chung, Ron Graham, Frederick Hoffman, Ronald C. Mullin, Leslie Hogben, and Douglas B. West. 1st. CRC Press, pp. 171–191. DOI: 10.1201/9780429280092-10. arXiv: 2003.05956.


Sugimori, Ken (2019). "Shortest Reconfiguration of Sliding Tokens on a Tree". MA thesis. University of Tokyo.


de Berg, Mark, Bart M. P. Jansen, and Debankur Mukherjee (2018). "Independent-Set Reconfiguration Thresholds of Hereditary Graph Classes". In: *Discrete Applied Mathematics* 250, pp. 165–182. DOI: 10.1016/j.dam.2018.05.029.

Bibliography (cont.)

- Hoang, Duc A., Amanj Khorramian, and Ryuhei Uehara (2018). “Shortest Reconfiguration Sequence for Sliding Tokens on Spiders”. In: arXiv: 1806.08291.
- Lokshtanov, Daniel and Amer E. Mouawad (2018). “The Complexity of Independent Set Reconfiguration on Bipartite Graphs”. In: *Proceedings of SODA 2018*. Siam, pp. 185–195. DOI: 10.1137/1.9781611975031.13. arXiv: 1707.02638.
- Nishimura, Naomi (2018). “Introduction to Reconfiguration”. In: *Algorithms* 11.4. (article 52). DOI: 10.3390/a11040052.
- Wrochna, Marcin (2018). “Reconfiguration in Bounded Bandwidth and Treedepth”. In: *Journal of Computer and System Sciences* 93, pp. 1–10. DOI: 10.1016/j.jcss.2017.11.003.

Bibliography (cont.)

- Alikhani, Saeid and Davood Fatehi (2017). “The k-Independent Graph of a Graph”. In: *Advances and Applications in Discrete Mathematics* 18.1, pp. 45–56. DOI: 10.17654/dm018010045. arXiv: 1510.05360.
- Bonamy, Marthe and Nicolas Bousquet (2017). “Token Sliding on Chordal Graphs”. In: *Proceedings of WG 2017*. Vol. 10520. LNCS. Springer, pp. 127–139. DOI: 10.1007/978-3-319-68705-6_10. arXiv: 1605.00442.
- Hoang, Duc A. and Ryuhei Uehara (2016). “Sliding Tokens on a Cactus”. In: *Proceedings of ISAAC 2016*. Vol. 64. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 37:1–37:26. DOI: 10.4230/LIPIcs.ISAAC.2016.37.

Bibliography (cont.)


Yamada, Takeshi and Ryuhei Uehara (2016). “Shortest Reconfiguration of Sliding Tokens on a Caterpillar”. In: *Proceedings of WALCOM 2016*. Vol. 9627. LNCS. Springer, pp. 236–248. DOI: 10.1007/978-3-319-30139-6_19. arXiv: 1511.00243.


Fox-Epstein, Eli, Duc A. Hoang, Yota Otachi, and Ryuhei Uehara (2015). “Sliding Token on Bipartite Permutation Graphs”. In: *Proceedings of ISAAC 2015*. Vol. 9472. LNCS. Springer, pp. 237–247. DOI: 10.1007/978-3-662-48971-0_21.

Bibliography (cont.)


van der Zanden, Tom C. (2015). “Parameterized Complexity of Graph Constraint Logic”. In: *Proceedings of IPEC 2015*. Vol. 43. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, pp. 282–293. DOI: 10.4230/LIPIcs.IPEC.2015.282. arXiv: 1509.02683.


Bonsma, Paul S. (2014). “Independent Set Reconfiguration in Cographs”. In: *arXiv preprint*. arXiv: 1402.1587.


Bonsma, Paul S., Marcin Kamiński, and Marcin Wrochna (2014). “Reconfiguring Independent Sets in Claw-Free Graphs”. In: *Proceedings of SWAT 2014*. Vol. 8503. LNCS. Springer, pp. 86–97. DOI: 10.1007/978-3-319-08404-6_8. arXiv: 1403.0359.

Bibliography (cont.)

Demaine, Erik D., Martin L. Demaine, Eli Fox-Epstein, Duc A. Hoang, Takehiro Ito, Hirotaka Ono, Yota Otachi, Ryuhei Uehara, and Takeshi Yamada (2014). "Polynomial-Time Algorithm for Sliding Tokens on Trees". In: *Proceedings of ISAAC 2014*. Vol. 8889. LNCS. Springer, pp. 389–400. DOI: 10.1007/978-3-319-13075-0_31. arXiv: 1406.6576.

Wrochna, Marcin (2014). "Reconfiguration in Bounded Bandwidth and Treedepth". In: *arXiv preprint*. arXiv: 1405.0847.

Bibliography (cont.)


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. arXiv: 1312.2816.


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.


Brešar, Boštjan, František Kardoš, Ján Katrenič, and Gabriel Semanišin (2011). “Minimum k-path vertex cover”. In: *Discrete Applied Mathematics* 159.12, pp. 1189–1195. DOI: 10.1016/j.dam.2011.04.008.

Bibliography (cont.)


Ito, Takehiro, Erik D. Demaine, Nicholas J. A. Harvey, Christos H. Papadimitriou, Martha Sideri, Ryuhei Uehara, and Yushi Uno (2011). "On the Complexity of Reconfiguration Problems". In: *Theoretical Computer Science* 412.12-14, pp. 1054–1065. DOI: 10.1016/j.tcs.2010.12.005.


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. arXiv: cs/0205005.


Story, William E. (1879). "Notes on the "15" Puzzle". In: *American Journal of Mathematics* 2.4, pp. 397–404. DOI: 10.2307/2369492.