Reconfiguration of Spanning Trees with Many or Few Leaves

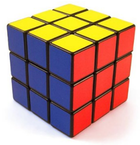
Nicolas Bousquet Takehiro Ito Yusuke Kobayashi
Haruka Mizuta Paul Ouvrard Akira Suzuki Kunihiro Wasa

ESA'20

Reconfiguration problems

Informal framework :

- Two solutions of an instance of a problem.
- An elementary transformation between two solutions.

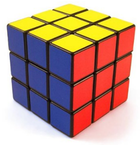


Question :

Given my current position, can I reach my target position ?

Reconfiguration problems

Informal framework :



- Two solutions of an instance of a problem.
- An elementary transformation between two solutions.

Question :

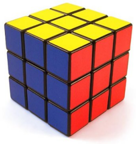
Given my current position, can I reach my target position ?

Many motivations :

- One-player games (puzzles).

Reconfiguration problems

Informal framework :



- Two solutions of an instance of a problem.
- An elementary transformation between two solutions.

Question :

Given my current position, can I reach my target position ?

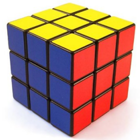
Many motivations :

- One-player games (puzzles).
- Markov chains (ergodicity and mixing time).

Reconfiguration problems

Informal framework :

- Two solutions of an instance of a problem.
- An elementary transformation between two solutions.



Question :

Given my current position, can I reach my target position ?

Many motivations :

- One-player games (puzzles).
- Markov chains (ergodicity and mixing time).
- Applications to many problems : statistical physics, biology, motion of robots, discrete geometry...

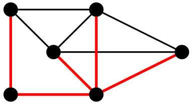
Main questions

- **Reachability problem.** Given two configurations, is it possible to transform the one into the other ?
- **Connectivity problem.** Given any pair of configurations, is it possible to transform the one into the other ?
- **Minimization.** Given two configurations, what is the length of a shortest sequence ?

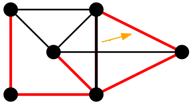
Main questions

- **Reachability problem.** Given two configurations, is it possible to transform the one into the other ?
- **Connectivity problem.** Given any pair of configurations, is it possible to transform the one into the other ?
- **Minimization.** Given two configurations, what is the length of a shortest sequence ?
- **Algorithmics.** Can we efficiently solve these problems ? (In polynomial time, FPT-time...).

Spanning tree reconfiguration



Spanning Tree : Subset of edges that induces a spanning connected graph without cycle.

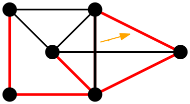


Spanning tree reconfiguration

Spanning Tree : Subset of edges that induces a spanning connected graph without cycle.

Elementary transformation.
T_1 can be transformed into T_2 via an edge flip if there exists $e_1 \in T_1$ and $e_2 \in T_2$ such that $T_2 = (T_1 \cup e_2) \setminus e_1$.

Spanning tree reconfiguration

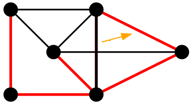


Spanning Tree : Subset of edges that induces a spanning connected graph without cycle.

Elementary transformation.
T_1 can be transformed into T_2 via an edge flip if there exists $e_1 \in T_1$ and $e_2 \in T_2$ such that $T_2 = (T_1 \cup e_2) \setminus e_1$.

SPANNING TREE RECONFIGURATION (STR)
Input : A graph G, two spanning trees T_1, T_2.
Question : It is possible to transform T_1 into T_2 via a sequence of edge flips ?

Spanning tree reconfiguration



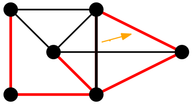
Spanning Tree : Subset of edges that induces a spanning connected graph without cycle.

Elementary transformation.
T_1 can be transformed into T_2 via an edge flip if there exists $e_1 \in T_1$ and $e_2 \in T_2$ such that $T_2 = (T_1 \cup e_2) \setminus e_1$.

SPANNING TREE RECONFIGURATION (STR)
Input : A graph G, two spanning trees T_1, T_2.
Question : It is possible to transform T_1 into T_2 via a sequence of edge flips ?

Answer : [Ito et al.] YES !

Spanning tree reconfiguration



Spanning Tree : Subset of edges that induces a spanning connected graph without cycle.

Elementary transformation.
T_1 can be transformed into T_2 via an edge flip if there exists $e_1 \in T_1$ and $e_2 \in T_2$ such that $T_2 = (T_1 \cup e_2) \setminus e_1$.

Spanning Tree Reconfiguration (STR)
Input : A graph G, two spanning trees T_1, T_2.
Question : It is possible to transform T_1 into T_2 via a sequence of edge flips ?

Answer : [Ito et al.] YES !

What if we add some constraints on the the spanning trees ?
$\rightarrow$ In this paper : Number of leaves.

STR with few leaves

Spanning Tree Reconfiguration (STR) with $\leq k$ Leaves
Input : A graph G, two spanning trees T_1, T_2 with $\leq k$ leaves.
Output : YES iff $T_1 \rightsquigarrow T_2$ via spanning trees with $\leq k$ leaves.

Our Result 1 :
STR with ≤ 3 Leaves is PSPACE-complete.

STR with few leaves

SPANNING TREE RECONFIGURATION (STR) WITH $\leq k$ LEAVES
Input : A graph G, two spanning trees T_1, T_2 with $\leq k$ leaves.
Output : YES iff $T_1 \rightsquigarrow T_2$ via spanning trees with $\leq k$ leaves.

Our Result 1 :
STR WITH ≤ 3 LEAVES is PSPACE-complete.

Remarks :

- STR WITH ≤ 2 LEAVES is HAMILTONIAN PATH RECONFIGURATION whose complexity is unknown.

STR with few leaves

SPANNING TREE RECONFIGURATION (STR) WITH $\leq k$ LEAVES
Input : A graph G, two spanning trees T_1, T_2 with $\leq k$ leaves.
Output : YES iff $T_1 \rightsquigarrow T_2$ via spanning trees with $\leq k$ leaves.

Our Result 1 :
STR WITH ≤ 3 LEAVES is PSPACE-complete.

Remarks :

- STR WITH ≤ 2 LEAVES is HAMILTONIAN PATH RECONFIGURATION whose complexity is unknown.
- 3 can be replaced by any integer ≥ 3 in the statement.

STR with many leaves

Spanning Tree Reconfiguration with Many Leaves
Input : A graph G, an integer k and two spanning trees T_1, T_2 with $\geq k$ leaves.
Output : YES iff $T_1 \rightsquigarrow T_2$ via spanning trees with $\geq k$ leaves.

STR with many leaves

SPANNING TREE RECONFIGURATION WITH MANY LEAVES
Input : A graph G, an integer k and two spanning trees T_1, T_2 with $\geq k$ leaves.
Output : YES iff $T_1 \rightsquigarrow T_2$ via spanning trees with $\geq k$ leaves.

Our results 2 :
STR WITH MANY LEAVES :

- is PSPACE-complete even restricted to bipartite graphs or split graphs ;
- is PSPACE-complete restricted to planar graphs ;

STR with many leaves

SPANNING TREE RECONFIGURATION WITH MANY LEAVES
Input : A graph G, an integer k and two spanning trees T_1, T_2 with $\geq k$ leaves.
Output : YES iff $T_1 \rightsquigarrow T_2$ via spanning trees with $\geq k$ leaves.

Our results 2 :
STR WITH MANY LEAVES :

- is PSPACE-complete even restricted to bipartite graphs or split graphs ;
- is PSPACE-complete restricted to planar graphs ;
- can be decided in polynomial time on cographs and interval graphs.

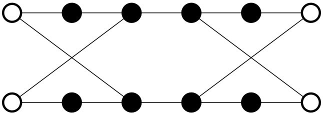
Reduction for STR with ≤ 3 leaves

Adaptation of a reduction from VERTEX COVER to HAMILTONIAN PATH.

Reduction for STR with ≤ 3 leaves

Adaptation of a reduction from Vertex Cover to Hamiltonian Path.

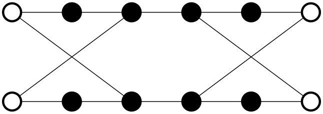
Edge gadget :



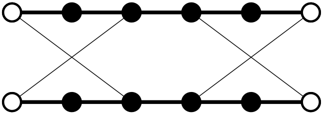
Reduction for STR with ≤ 3 leaves

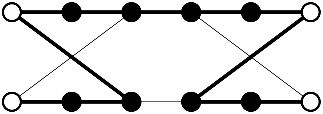
Adaptation of a reduction from Vertex Cover to Hamiltonian Path.

Edge gadget :



Lemma (folklore) : Any hamiltonian path must intersect the edge gadget in one of the two following ways (up to symmetry) :

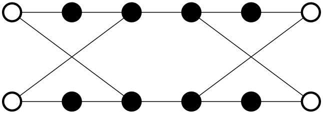




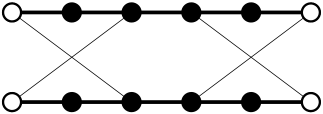
Reduction for STR with ≤ 3 leaves

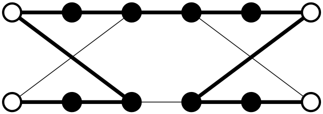
Adaptation of a reduction from Vertex Cover to Hamiltonian Path.

Edge gadget :



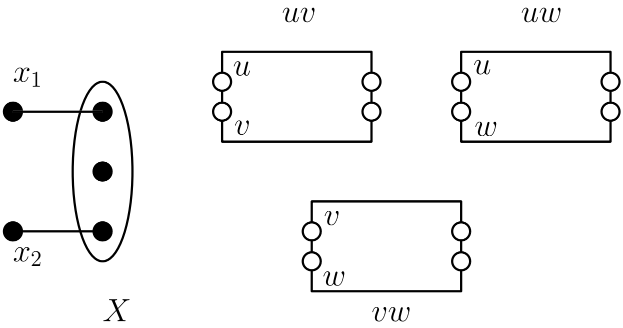
Lemma (folklore) : Any hamiltonian path must intersect the edge gadget in one of the two following ways (up to symmetry) :





Lemma : In any spanning tree with ≤ 3 leaves, ≥ 1 white vertex has degree one in the subgraph induced by the edge-gadget.

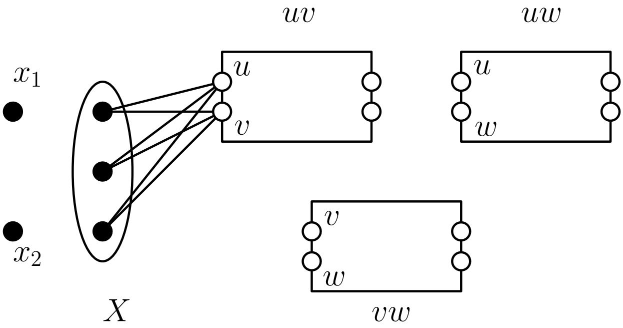
Reduction for STR with ≤ 3 leaves (cont.)



Reduction from *k*-Vertex Cover :

- Two vertices of degree 1, an independant set X of size $k+1$, an edge gadget for every edge.

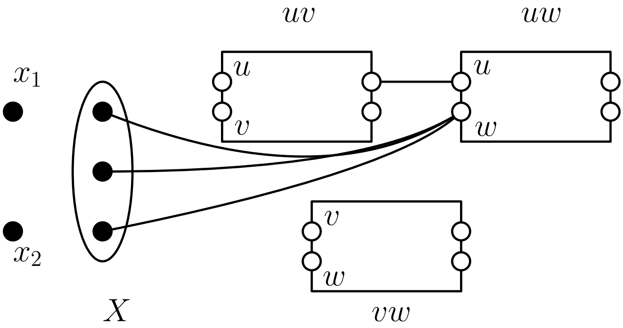
Reduction for STR with ≤ 3 leaves (cont.)



Reduction from *k*-Vertex Cover :

- Two vertices of degree 1, an independant set X of size $k+1$, an edge gadget for every edge.
- For every edge uv, connect the "special" entering vertex of u to the vertices of X or to the previous "special" exit vertex of u.

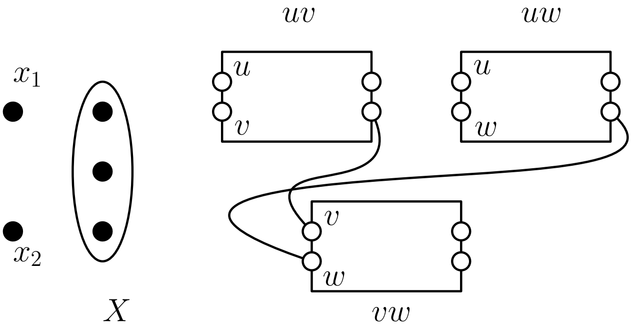
Reduction for STR with ≤ 3 leaves (cont.)



Reduction from k-Vertex Cover :

- Two vertices of degree 1, an independant set X of size $k + 1$, an edge gadget for every edge.
- For every edge uv, connect the "special" entering vertex of u to the vertices of X or to the previous "special" exit vertex of u.

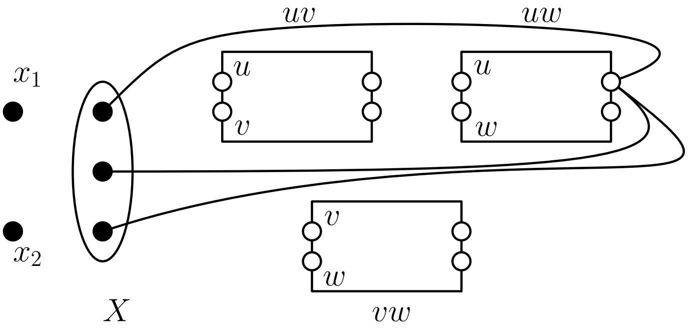
Reduction for STR with ≤ 3 leaves (cont.)



Reduction from k-Vertex Cover :

- Two vertices of degree 1, an independant set X of size $k+1$, an edge gadget for every edge.
- For every edge uv, connect the "special" entering vertex of u to the vertices of X or to the previous "special" exit vertex of u.

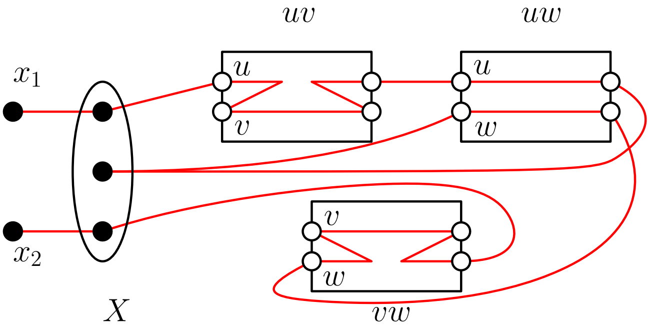
Reduction for STR with ≤ 3 leaves (cont.)



Reduction from k-Vertex Cover :

- Two vertices of degree 1, an independant set X of size $k + 1$, an edge gadget for every edge.
- For every edge uv, connect the "special" entering vertex of u to the vertices of X or to the previous "special" exit vertex of u.

Reduction for STR with ≤ 3 leaves (cont.)



Reduction from k-Vertex Cover :

- Two vertices of degree 1, an independant set X of size $k + 1$, an edge gadget for every edge.
- For every edge uv, connect the "special" entering vertex of u to the vertices of X or to the previous "special" exit vertex of u.

Reconfiguration adaptation

Theorem [Wrochna]

(TAR) MINIMUM VERTEX COVER RECONFIGURATION is PSPACE-complete.

Reconfiguration adaptation

Theorem [Wrochna]

(TAR) MINIMUM VERTEX COVER RECONFIGURATION is PSPACE-complete.

$S(T)$: Set of vertices v such that, for some edge gadget, (at least) one of the white vertices of v has degree one in the restriction of T to the edge gadget.

Reconfiguration adaptation

Theorem [Wrochna]

(TAR) Minimum Vertex Cover Reconfiguration is PSPACE-complete.

$S(T)$: Set of vertices v such that, for some edge gadget, (at least) one of the white vertices of v has degree one in the restriction of T to the edge gadget.

Remark : $S(T)$ is a vertex cover for any spanning tree with ≤ 3 leaves.

Reconfiguration adaptation

Theorem [Wrochna]

(TAR) MINIMUM VERTEX COVER RECONFIGURATION is PSPACE-complete.

$S(T)$: Set of vertices v such that, for some edge gadget, (at least) one of the white vertices of v has degree one in the restriction of T to the edge gadget.

Remark : $S(T)$ is a vertex cover for any spanning tree with ≤ 3 leaves.

Technical lemmas :

- Show that $|S(T)| \leq k + 1$ for any T with ≤ 3 leaves.

Reconfiguration adaptation

Theorem [Wrochna]

(TAR) Minimum Vertex Cover Reconfiguration is PSPACE-complete.

$S(T)$: Set of vertices v such that, for some edge gadget, (at least) one of the white vertices of v has degree one in the restriction of T to the edge gadget.

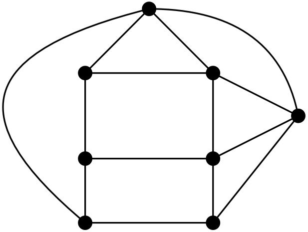
Remark : $S(T)$ is a vertex cover for any spanning tree with ≤ 3 leaves.

Technical lemmas :

- Show that $|S(T)| \leq k + 1$ for any T with ≤ 3 leaves.
- Show that if T_2 can be obtained from T_1 via an edge flip then it corresponds to a "single step" modication for vertex cover reconfiguration.

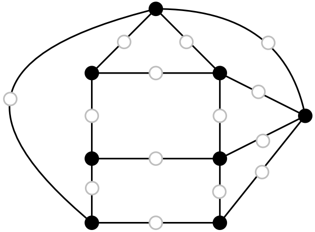
STR with many Leaves hardness for planar graphs

Reduction from MINIMUM VERTEX COVER RECONFIGURATION ON PLANAR GRAPHS



STR with many Leaves hardness for planar graphs

Reduction from MINIMUM VERTEX COVER RECONFIGURATION ON PLANAR GRAPHS

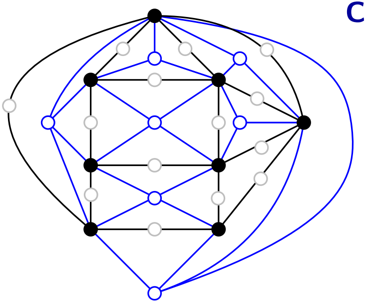


Construction :

- First subdivide every edge once.

STR with many Leaves hardness for planar graphs

Reduction from MINIMUM VERTEX COVER RECONFIGURATION ON PLANAR GRAPHS

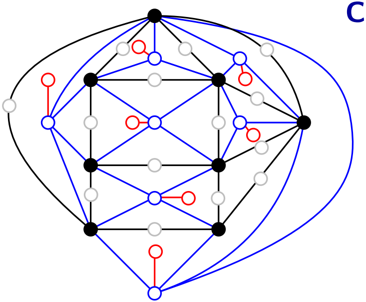


Construction :

- First subdivide every edge once.
- Create a new vertex for each face and connect it to all the vertices of its face.

STR with many Leaves hardness for planar graphs

Reduction from Minimum Vertex Cover Reconfiguration on Planar Graphs

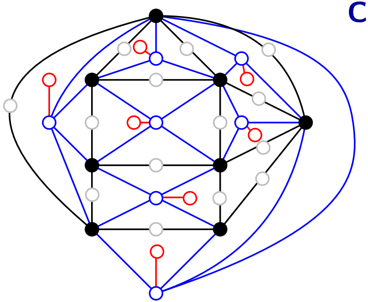


Construction :

- First subdivide every edge once.
- Create a new vertex for each face and connect it to all the vertices of its face.
- Create a new vertex for each face and connect it to its corresponding blue vertex.

STR with many Leaves hardness for planar graphs

Reduction from MINIMUM VERTEX COVER RECONFIGURATION ON PLANAR GRAPHS



Construction :

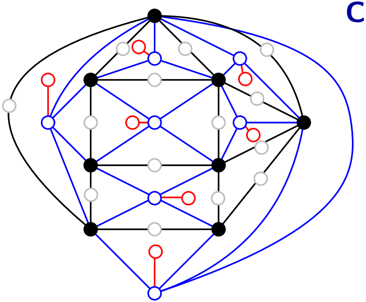
- First subdivide every edge once.
- Create a new vertex for each face and connect it to all the vertices of its face.
- Create a new vertex for each face and connect it to its corresponding blue vertex.

Remarks :

1. • vertices are internal nodes.

STR with many Leaves hardness for planar graphs

Reduction from MINIMUM VERTEX COVER RECONFIGURATION ON PLANAR GRAPHS



Construction :

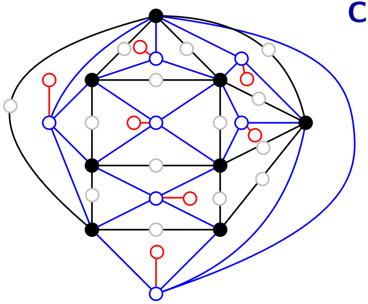
- First subdivide every edge once.
- Create a new vertex for each face and connect it to all the vertices of its face.
- Create a new vertex for each face and connect it to its corresponding blue vertex.

Remarks :

1. • vertices are internal nodes.
2. • vertices are leaves.

STR with many Leaves hardness for planar graphs

Reduction from Minimum Vertex Cover Reconfiguration on Planar Graphs



Construction :

- First subdivide every edge once.
- Create a new vertex for each face and connect it to all the vertices of its face.
- Create a new vertex for each face and connect it to its corresponding blue vertex.

Remarks :

1. • vertices are internal nodes.
2. • vertices are leaves.
3. ≥ 1 neighbor of a • vertex is internal.
   $\Rightarrow \geq$ faces + min vertex cover leaves.

Polynomial algorithm on cographs

Polynomial algorithm on cographs

Lemma 1

The answer (on cographs) is always positive if $k \leq n - 3$.

Polynomial algorithm on cographs

Lemma 1

The answer (on cographs) is always positive if $k \leq n - 3$.

Lemma 2

STR WITH $\geq n - 2$ LEAVES is in P (for any graph).

Polynomial algorithm on cographs

Lemma 1

The answer (on cographs) is always positive if $k \leq n - 3$.

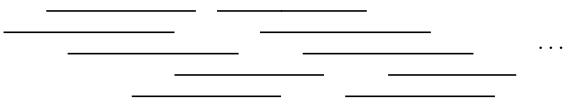
Lemma 2

STR WITH $\geq n - 2$ LEAVES is in P (for any graph).

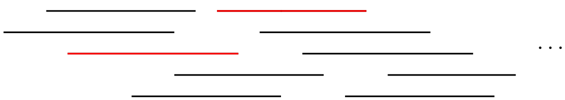
Sketch of the proof :

- All the spanning trees with the same internal nodes are reachable from each other.
- Given two sets X, Y of size 2, we can decide in polytime if there is an edge flip from a tree with internal nodes X to a tree with internal nodes Y.

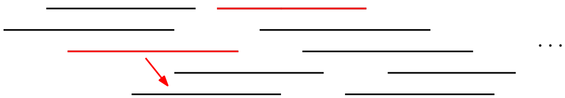
Polynomial algorithm for interval graphs



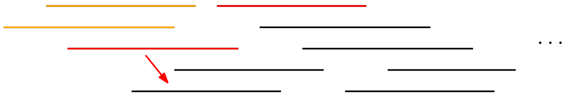
Polynomial algorithm for interval graphs



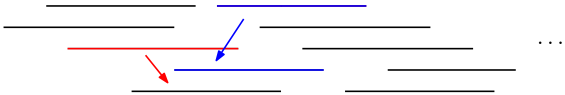
Polynomial algorithm for interval graphs



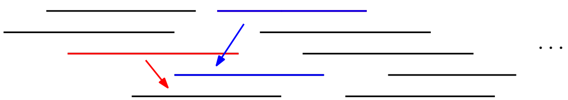
Polynomial algorithm for interval graphs



Polynomial algorithm for interval graphs



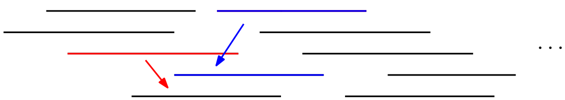
Polynomial algorithm for interval graphs



Idea of the polynomial time algorithm :

- Fix the first interval vertex.
- Push left the second interval vertex by "induction". (It is actually more complicated, it is not really an induction since we cannot simply delete the left part, but we can reduce it somehow...)
- Push right the first interval vertex.

Polynomial algorithm for interval graphs



Idea of the polynomial time algorithm :

- Fix the first interval vertex.
- Push left the second interval vertex by “induction”. (It is actually more complicated, it is not really an induction since we cannot simply delete the left part, but we can reduce it somehow...)
- Push right the first interval vertex.

(Technical) Lemma : The vertex obtained by this algorithm is the rightmost possible position of the first interval vertex in a spanning tree reachable from the initial tree.

Conclusion

Questions :

- Complexity of Spanning Tree Reconfiguration (STR) with Few Leaves on restricted graph classes ? (The reduction do not maintain any parameter...)

Conclusion

Questions :

- Complexity of Spanning Tree Reconfiguration (STR) with Few Leaves on restricted graph classes ? (The reduction do not maintain any parameter...)
- Lemma : STR with $\geq n-2$ Leaves is in P.
  Is STR with $\geq n-\ell$ leaves FPT parameterized by ℓ ?

Conclusion

Questions :

- Complexity of Spanning Tree Reconfiguration (STR) with Few Leaves on restricted graph classes ? (The reduction do not maintain any parameter...)
- Lemma : STR with $\geq n-2$ Leaves is in P.
  Is STR with $\geq n-\ell$ leaves FPT parameterized by ℓ ?
- Is STR with many leaves polynomial in outerplanar graphs ?

Conclusion

Questions :

- Complexity of Spanning Tree Reconfiguration (STR) with Few Leaves on restricted graph classes ? (The reduction do not maintain any parameter...)
- Lemma : STR with $\geq n - 2$ Leaves is in P.
  Is STR with $\geq n - \ell$ leaves FPT parameterized by ℓ ?
- Is STR with many leaves polynomial in outerplanar graphs ?

Thanks !