Computing Distances on Graph Associahedra is Fixed-parameter Tractable

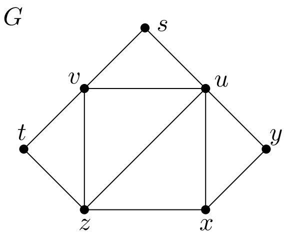
Luís Felipe I. Cunha	Univ. Federal Fluminense, Niterói, Brazil
Ignasi Sau	**LIRMM, Univ. Montpellier, CNRS, France**
Uéverton S. Souza	UFF+IMPA, Niterói+Rio de Janeiro, Brazil
Mario Valencia-Pabon	Univ. Lorraine, LORIA, Nancy, France

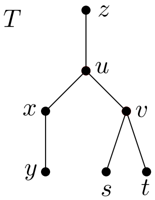
52nd International Colloquium on Automata, Languages and Programming (ICALP)

Aarhus University, Denmark, July 8-11, 2025

Elimination trees

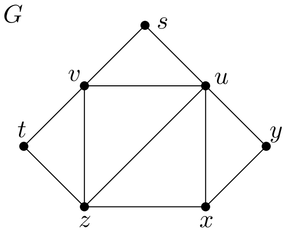
Tree T obtained from a graph G by picking recursively a vertex in each connected component of the current graph:

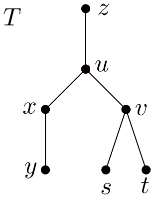




Elimination trees (or forests)

Tree T obtained from a graph G by picking recursively a vertex in each connected component of the current graph:



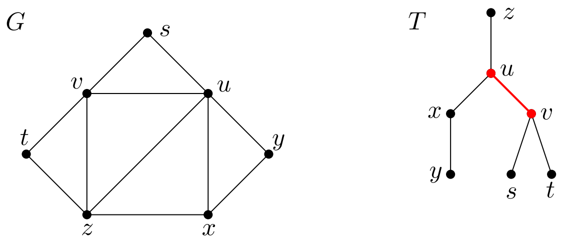


Elimination trees have been studied extensively in many contexts:

graph theory, combinatorial optimization, polyhedral combinatorics, data structures, VLSI design, ...

Elimination trees (or forests)

Tree T obtained from a graph G by picking recursively a vertex in each connected component of the current graph:

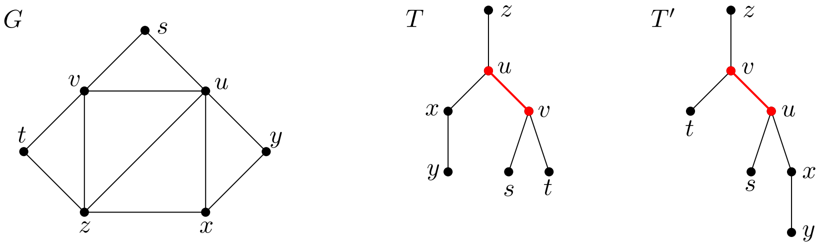


Elimination trees have been studied extensively in many contexts:

graph theory, combinatorial optimization, polyhedral combinatorics, data structures, VLSI design, ...

Elimination trees (or forests)

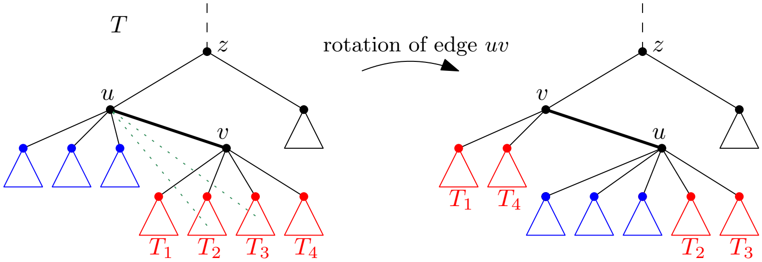
Tree T obtained from a graph G by picking recursively a vertex in each connected component of the current graph:



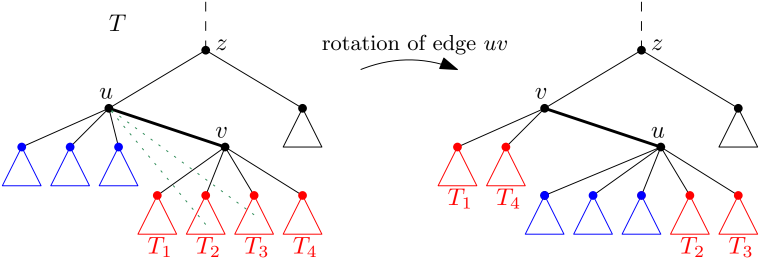
Elimination trees have been studied extensively in many contexts:

graph theory, combinatorial optimization, polyhedral combinatorics, data structures, VLSI design, ...

Rotation distance between elimination trees



Rotation distance between elimination trees



The rotation distance between two elimination trees (forests) T, T' of a graph G, denoted by $\mathrm{dist}(T, T')$, is the minimum number of rotations it takes to transform T into T'.

Graph associahedra

For any graph G, the flip graph of elimination forests of G under edge rotations is the skeleton of a polytope: graph associahedron $\mathcal{A}(G)$.

Object introduced by [Carr, Devadoss, Postnikov. 2006-2009]

Graph associahedra

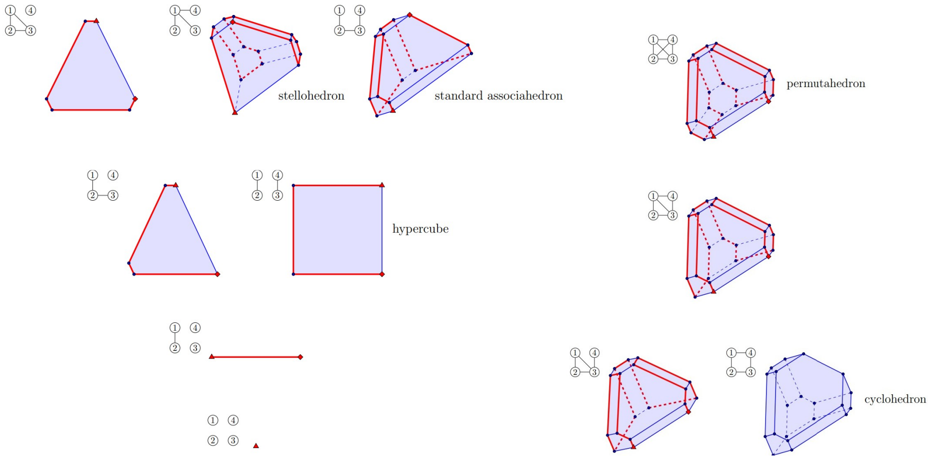
For any graph G, the flip graph of elimination forests of G under edge rotations is the skeleton of a polytope: graph associahedron $\mathcal{A}(G)$.

Object introduced by [Carr, Devadoss, Postnikov. 2006-2009]

Famous particular cases of $\mathcal{A}(G)$ depending on the underlying graph G:

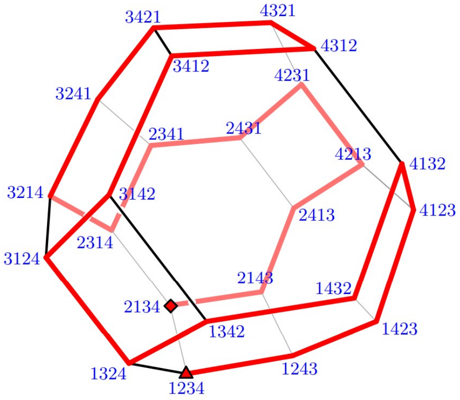
G	$\mathcal{A}(G)$
path	(standard) associahedron
complete graph	permutahedron
cycle	cyclohedron
star	stellohedron
matching	hypercube

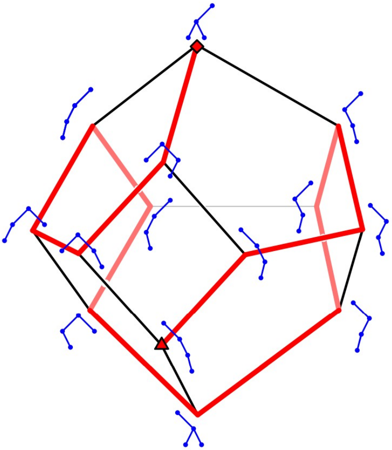
Illustration of some famous examples



Shamelessly stolen from this very nice article: [Cardinal, Merino, Mütze. 2022]

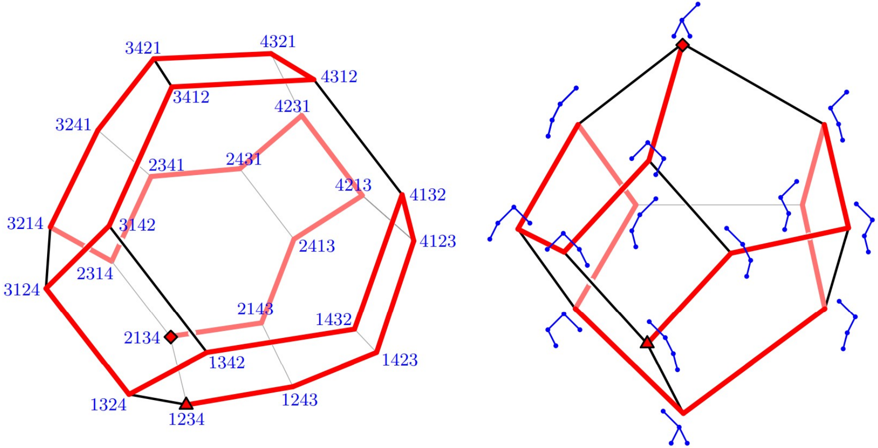
Zooming in: permutahedron and (standard) associahedron





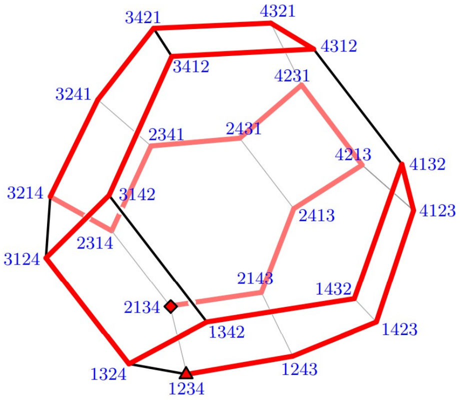
Shamelessly stolen from this very nice article: [Cardinal, Merino, Mütze. 2022]

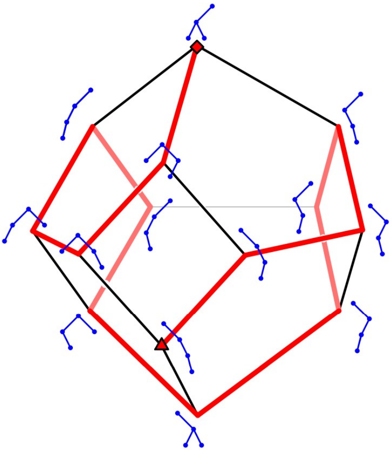
Zooming in: permutahedron and (standard) associahedron



The (standard) associahedron has a rich history and literature, connecting computer science, combinatorics, algebra, and topology.

Zooming in: permutahedron and (standard) associahedron





Binary trees are in bijection with many other Catalan objects:
triangulations of a convex polygon, well-formed parenthesis, Dyck paths, ...

Intensively studied: diameter of graph associahedra

Determining the diameter exactly, or upper/lower bounds, or estimates:

- If G is a path: [Sleator, Tarjan, Thurston. 1998] [Pournin. 2014]
- If G is a star: [Manneville, Pilaud. 2010]
- If G is a cycle: [Pournin. 2017]
- If G is a tree: [Manneville, Pilaud. 2010] [Cardinal, Langerman, Pérez-Lantero. 2018]
- If G is a complete bipartite or trivially perfect graph: [Cardinal, Pournin, Valencia-Pabon. 2022]
- If G is a caterpillar: [Berendsohn. 2022]
- If G has bounded treedepth or treewidth: [Cardinal, Pournin, Valencia-Pabon. 2022]

Our focus: computing distances on graph associahedra

Suppose for simplicity that the considered graph G is connected.

Rotation Distance

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.

Question: Is the rotation distance between T and T' at most k?

Our focus: computing distances on graph associahedra

Suppose for simplicity that the considered graph G is connected.

Rotation Distance

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.

Question: Is the rotation distance between T and T' at most k?

Only few cases known to be solvable in polynomial time:

- If G is a complete graph: [Folklore]
- If G is a star: [Cardinal, Pournin, Valencia-Pabon. 2022]
- If G is a complete split graph: [Cardinal, Pournin, Valencia-Pabon. 2024]

Our focus: computing distances on graph associahedra

Suppose for simplicity that the considered graph G is connected.

Rotation Distance

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.

Question: Is the rotation distance between T and T' at most k?

Only few cases known to be solvable in polynomial time:

- If G is a complete graph: [Folklore]
- If G is a star: [Cardinal, Pournin, Valencia-Pabon. 2022]
- If G is a complete split graph: [Cardinal, Pournin, Valencia-Pabon. 2024]

Notorious open problem (polynomial or NP-hard?): if G is a path.

Our focus: computing distances on graph associahedra

Suppose for simplicity that the considered graph G is connected.

Rotation Distance

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.

Question: Is the rotation distance between T and T' at most k?

Only few cases known to be solvable in polynomial time:

- If G is a complete graph: [Folklore]
- If G is a star: [Cardinal, Pournin, Valencia-Pabon. 2022]
- If G is a complete split graph: [Cardinal, Pournin, Valencia-Pabon. 2024]

Notorious open problem (polynomial or NP-hard?): if G is a path.

This is $\boxed{\text{not}}$ the problem we solve!

Back to the general case

Rotation Distance

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.

Question: Is the rotation distance between T and T' at most k?

Back to the general case

ROTATION DISTANCE

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.

Question: Is the rotation distance between T and T' at most k?

Is the problem NP-hard for a general graph G?

[Cardinal, Kleist, Klemz, Lubiw, Mütze, Neuhaus, Pournin. **Dagstuhl 2022**]

Back to the general case

ROTATION DISTANCE

Instance:	A graph G, two elimination trees T and T' of G, and a positive integer k.
Question:	Is the rotation distance between T and T' at most k?

Is the problem NP-hard for a general graph G?

[Cardinal, Kleist, Klemz, Lubiw, Mütze, Neuhaus, Pournin. **Dagstuhl 2022**]

Yes, it is!

[Ito, Kakimura, Kamiyama, Kobayashi, Maezawa, Nozaki, Okamoto. **ICALP 2023**]

Back to the general case

Rotation Distance

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.

Question: Is the rotation distance between T and T' at most k?

Is the problem NP-hard for a general graph G?

[Cardinal, Kleist, Klemz, Lubiw, Mütze, Neuhaus, Pournin. **Dagstuhl 2022**]

Yes, it is!

[Ito, Kakimura, Kamiyama, Kobayashi, Maezawa, Nozaki, Okamoto. **ICALP 2023**]

[Cardinal, Steiner. 2023]

Back to the general case

Rotation Distance

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.

Question: Is the rotation distance between T and T' at most k?

Is the problem NP-hard for a general graph G?

[Cardinal, Kleist, Klemz, Lubiw, Mütze, Neuhaus, Pournin. **Dagstuhl 2022**]

Yes, it is!

[Ito, Kakimura, Kamiyama, Kobayashi, Maezawa, Nozaki, Okamoto. **ICALP 2023**]

[Cardinal, Steiner. 2023]

This motivates the study of the $\boxed{\text{parameterized complexity}}$ of the problem.

Statement of the parameterized problem

ROTATION DISTANCE

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.
Parameter: k.
Question: Is the rotation distance between T and T' at most k?

Statement of the parameterized problem and our result

ROTATION DISTANCE

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.
Parameter: k.
Question: Is the rotation distance between T and T' at most k?

Theorem

The ROTATION DISTANCE *problem can be solved in time* $f(k) \cdot |V(G)|$,

Statement of the parameterized problem and our result

ROTATION DISTANCE

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.
Parameter: k.
Question: Is the rotation distance between T and T' at most k?

Theorem

The ROTATION DISTANCE *problem can be solved in time* $f(k) \cdot |V(G)|$,

$$\textit{with} \quad f(k) = k^{k \cdot 2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}},$$

where the tower of exponentials has height at most $(3k+1)4k = \mathcal{O}(k^2)$.

Statement of the parameterized problem and our result

ROTATION DISTANCE

Instance: A graph G, two elimination trees T and T' of G, and a positive integer k.
Parameter: k.
Question: Is the rotation distance between T and T' at most k?

Theorem

The ROTATION DISTANCE *problem can be solved in time* $f(k) \cdot |V(G)|$,

$$\textit{with} \quad f(k) = k^{k \cdot 2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}},$$

where the tower of exponentials has height at most $(3k+1)4k = \mathcal{O}(k^2)$.

Prior to our work, only the case where G is a path was known to be FPT.

[Cleary, St. John. 2009] [Lucas. 2010] [Kanj, Sedgwick, Xia. 2017] [Li, Xia. 2023]

Main ideas of the FPT algorithm

Goal: find an ℓ-rotation sequence σ from T to T', for some $\ell \leq k$.

Main ideas of the FPT algorithm

Goal: find an ℓ-rotation sequence σ from T to T', for some $\ell \leq k$.

High level: identify a subset of marked vertices $M \subseteq V(T)$, of size $\leq f(k)$, so that we can assume that the desired ℓ-rotation sequence σ uses only vertices in M.

Main ideas of the FPT algorithm

Goal: find an ℓ-rotation sequence σ from T to T', for some $\ell \leq k$.

High level: identify a subset of marked vertices $M \subseteq V(T)$, of size $\leq f(k)$, so that we can assume that the desired ℓ-rotation sequence σ uses only vertices in M.

Once this is proved, an FPT algorithm follows directly by applying brute force and guessing all possible rotations using only vertices in M.

Main ideas of the FPT algorithm

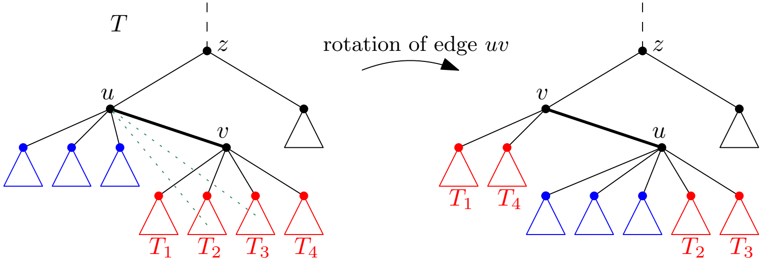
Goal: find an ℓ-rotation sequence σ from T to T', for some $\ell \leq k$.

High level: identify a subset of marked vertices $M \subseteq V(T)$, of size $\leq f(k)$, so that we can assume that the desired ℓ-rotation sequence σ uses only vertices in M.

Once this is proved, an FPT algorithm follows directly by applying brute force and guessing all possible rotations using only vertices in M.

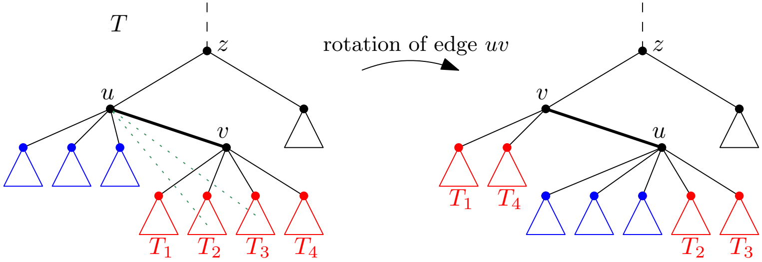
Let us see how we find such a "small" set $M \subseteq V(T)$ of marked vertices...

There are few children-bad vertices



Observation: a rotation may change the set of children of at most three vertices (but the parent of arbitrarily many vertices).

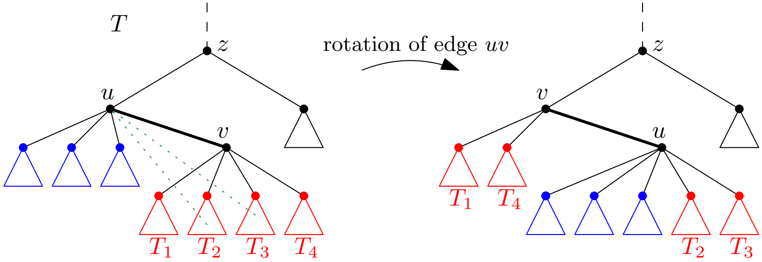
There are few children-bad vertices



Observation: a rotation may change the set of children of at most three vertices (but the parent of arbitrarily many vertices).

A vertex $v \in V(T)$ is (T, T')-children-bad if its set of children in T is different from its set of children in T'.

There are few children-bad vertices



Observation: a rotation may change the set of children of at most three vertices (but the parent of arbitrarily many vertices).

A vertex $v \in V(T)$ is (T, T')-children-bad if its set of children in T is different from its set of children in T'.

We may assume that there are at most $3k$ (T, T')-children-bad vertices.

Restricting the rotations to small balls around bad vertices

Observation: a rotation may change vertex distances (in T) by ≤ 1.

Restricting the rotations to small balls around bad vertices

Observation: a rotation may change vertex distances (in T) by ≤ 1.

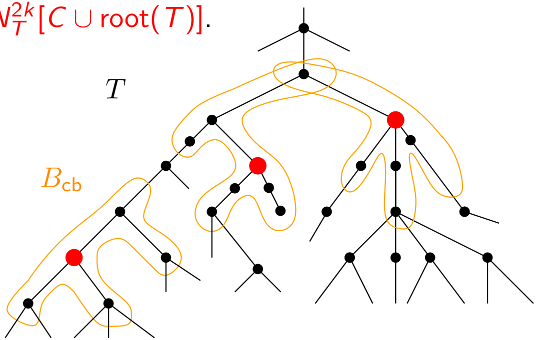
Let $C \subseteq V(T)$ be the set of (T, T')-children-bad vertices.

Restricting the rotations to small balls around bad vertices

Observation: a rotation may change vertex distances (in T) by ≤ 1.

Let $C \subseteq V(T)$ be the set of (T, T')-children-bad vertices.

We define $B_{\mathrm{cb}} = N_T^{2k}[C \cup \mathrm{root}(T)]$.

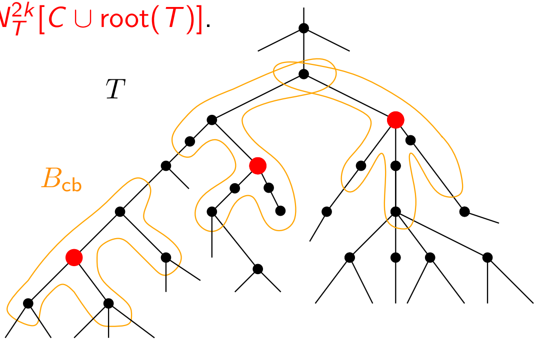


Restricting the rotations to small balls around bad vertices

Observation: a rotation may change vertex distances (in T) by ≤ 1.

Let $C \subseteq V(T)$ be the set of (T, T')-children-bad vertices.

We define $B_{\text{cb}} = N_T^{2k}[C \cup \text{root}(T)]$.



Lemma

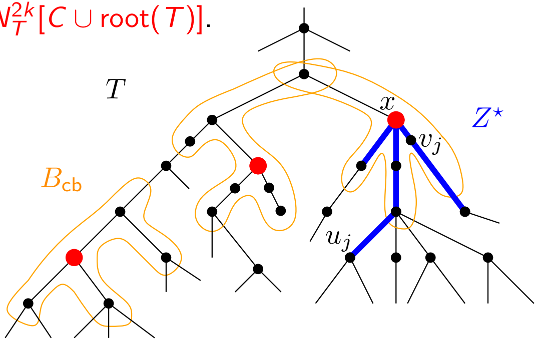
If $\text{dist}(T, T') \leq k$, then there exists an ℓ-rotation sequence from T to T', with $\ell \leq k$, using only vertices in B_{cb}.

Restricting the rotations to small balls around bad vertices

Observation: a rotation may change vertex distances (in T) by ≤ 1.

Let $C \subseteq V(T)$ be the set of (T, T')-children-bad vertices.

We define $B_{\text{cb}} = N_T^{2k}[C \cup \text{root}(T)]$.



Lemma

If $\text{dist}(T, T') \leq k$, then there exists an ℓ-rotation sequence from T to T', with $\ell \leq k$, using only vertices in B_{cb}.

If $\Delta(T)$ is bounded (in particular, if $\Delta(G)$ is bounded), we are done!

Towards the marking algorithm: trace of a vertex

Only obstacle to get our FPT algorithm: high-degree vertices in $T[B_{\text{cb}}]$.

Towards the marking algorithm: trace of a vertex

Only obstacle to get our FPT algorithm: high-degree vertices in $T[B_{\mathrm{cb}}]$.

Fix a connected component Z of $T[B_{\mathrm{cb}}]$ (considered as a rooted tree).

Towards the marking algorithm: trace of a vertex

Only obstacle to get our FPT algorithm: high-degree vertices in $T[B_{\mathrm{cb}}]$.

Fix a connected component Z of $T[B_{\mathrm{cb}}]$ (considered as a rooted tree).

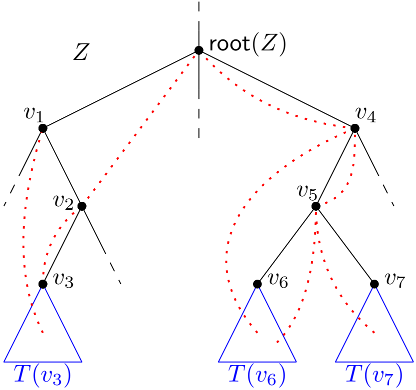
$\mathrm{trace}(T, Z, v)$: "abstract" neighborhood of $T(v)$ in its ancestors in Z.

Towards the marking algorithm: trace of a vertex

Only obstacle to get our FPT algorithm: high-degree vertices in $T[B_{\text{cb}}]$.

Fix a connected component Z of $T[B_{\text{cb}}]$ (considered as a rooted tree).

trace(T, Z, v): "abstract" neighborhood of $T(v)$ in its ancestors in Z.



trace$(T, Z, v_1) = (1)$
trace$(T, Z, v_2) = (1, 1)$
trace$(T, Z, v_3) = (1, 1, 0)$
trace$(T, Z, v_4) = (1)$
trace$(T, Z, v_5) = (1, 0)$
trace$(T, Z, v_6) = (1, 1, 0)$
trace$(T, Z, v_7) = (1, 0, 0)$

Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

Same type: same "variety of traces among children in Z"

Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

Same type: same "variety of traces among children in Z" $\rightarrow$ recursive!

Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

Same type: same "variety of traces among children in Z" $\rightarrow$ recursive!

Problem: #children may be unbounded, and we want #types $\leq f(k)$.

Defining the type of a vertex: $\tau(T, Z, v)$

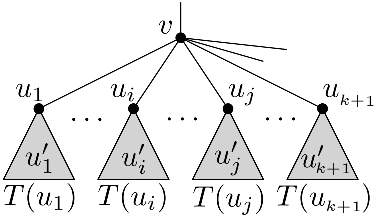
Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

Same type: same "variety of traces among children in Z" $\rightarrow$ recursive!

Problem: #children may be unbounded, and we want #types $\leq f(k)$.

Lemma

Let σ be an ℓ-rotation sequence from T to T', for some $\ell \leq k$. For every vertex $v \in V(T)$, there are at most k vertices $u_1, \ldots, u_k \in$ children(T, v) such that σ uses a vertex in each of the rooted subtrees $T(u_1), \ldots, T(u_k)$.



Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

Same type: same "variety of traces among children in Z" $\rightarrow$ recursive!

Problem: #children may be unbounded, and we want #types $\leq f(k)$.

Lemma

Let σ be an ℓ-rotation sequence from T to T', for some $\ell \leq k$. For every vertex $v \in V(T)$, there are at most k vertices $u_1, \ldots, u_k \in \mathrm{children}(T, v)$ such that σ uses a vertex in each of the rooted subtrees $T(u_1), \ldots, T(u_k)$.

Crucial: enough to keep track of "at least $k + 1$", not the actual number.

Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

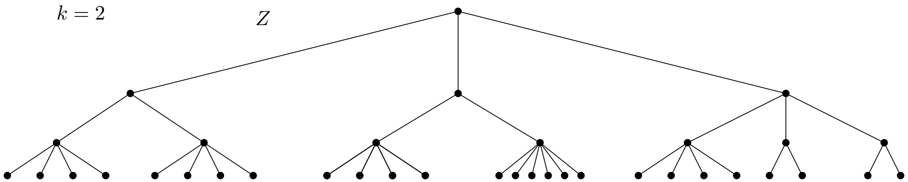
Same type: same "variety of traces among children in Z" $\rightarrow$ recursive!

Problem: #children may be unbounded, and we want #types $\leq f(k)$.

Lemma

Let σ be an ℓ-rotation sequence from T to T', for some $\ell \leq k$. For every vertex $v \in V(T)$, there are at most k vertices $u_1, \ldots, u_k \in \text{children}(T, v)$ such that σ uses a vertex in each of the rooted subtrees $T(u_1), \ldots, T(u_k)$.

Crucial: enough to keep track of "at least $k+1$", not the actual number.



Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

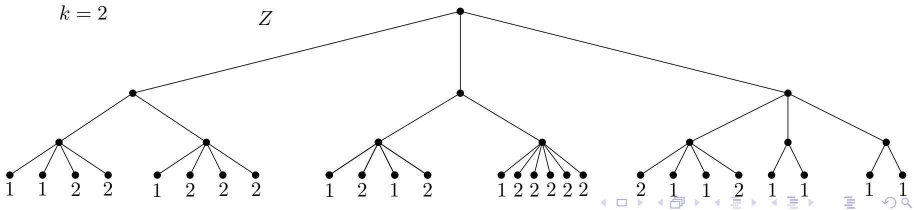
Same type: same "variety of traces among children in Z" $\rightarrow$ recursive!

Problem: #children may be unbounded, and we want #types $\leq f(k)$.

Lemma

Let σ be an ℓ-rotation sequence from T to T', for some $\ell \leq k$. For every vertex $v \in V(T)$, there are at most k vertices $u_1, \ldots, u_k \in \text{children}(T, v)$ such that σ uses a vertex in each of the rooted subtrees $T(u_1), \ldots, T(u_k)$.

Crucial: enough to keep track of "at least $k + 1$", not the actual number.



Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

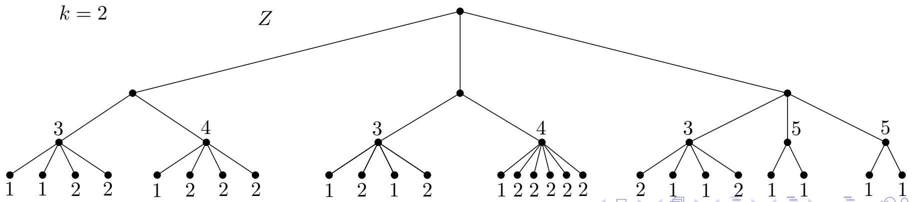
Same type: same "variety of traces among children in Z" $\rightarrow$ recursive!

Problem: #children may be unbounded, and we want #types $\leq f(k)$.

Lemma

Let σ be an ℓ-rotation sequence from T to T', for some $\ell \leq k$. For every vertex $v \in V(T)$, there are at most k vertices $u_1, \ldots, u_k \in$ children(T, v) such that σ uses a vertex in each of the rooted subtrees $T(u_1), \ldots, T(u_k)$.

Crucial: enough to keep track of "at least $k+1$", not the actual number.



Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

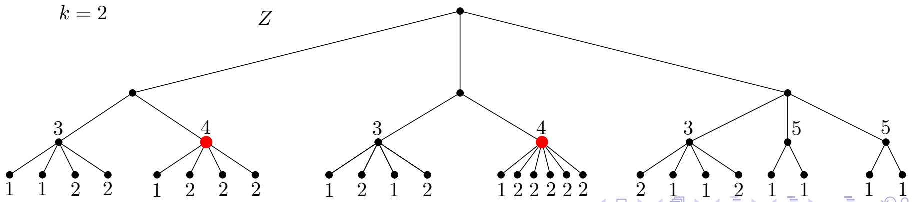
Same type: same "variety of traces among children in Z" $\rightarrow$ recursive!

Problem: #children may be unbounded, and we want #types $\leq f(k)$.

Lemma

Let σ be an ℓ-rotation sequence from T to T', for some $\ell \leq k$. For every vertex $v \in V(T)$, there are at most k vertices $u_1, \ldots, u_k \in \text{children}(T, v)$ such that σ uses a vertex in each of the rooted subtrees $T(u_1), \ldots, T(u_k)$.

Crucial: enough to keep track of "at least $k+1$", not the actual number.



Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

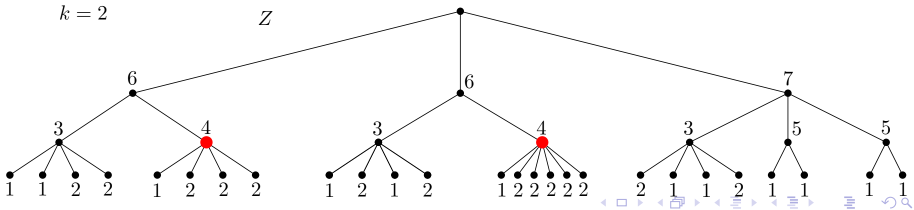
Same type: same "variety of traces among children in Z" $\rightarrow$ recursive!

Problem: #children may be unbounded, and we want #types $\leq f(k)$.

Lemma

Let σ be an ℓ-rotation sequence from T to T', for some $\ell \leq k$. For every vertex $v \in V(T)$, there are at most k vertices $u_1, \ldots, u_k \in \text{children}(T, v)$ such that σ uses a vertex in each of the rooted subtrees $T(u_1), \ldots, T(u_k)$.

Crucial: enough to keep track of "at least $k+1$", not the actual number.



Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

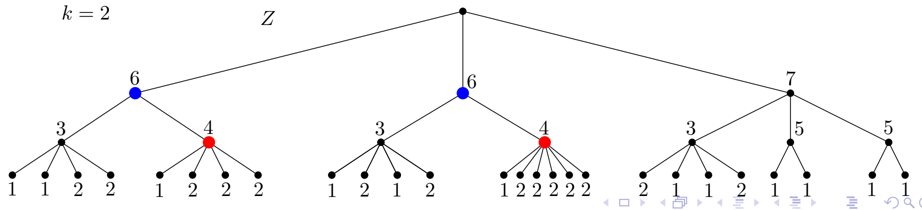
Same type: same "variety of traces among children in Z" $\rightarrow$ recursive!

Problem: #children may be unbounded, and we want #types $\leq f(k)$.

Lemma

Let σ be an ℓ-rotation sequence from T to T', for some $\ell \leq k$. For every vertex $v \in V(T)$, there are at most k vertices $u_1, \ldots, u_k \in \mathrm{children}(T, v)$ such that σ uses a vertex in each of the rooted subtrees $T(u_1), \ldots, T(u_k)$.

Crucial: enough to keep track of "at least $k + 1$", not the actual number.



Defining the type of a vertex: $\tau(T, Z, v)$

Goal: if $\tau(T, Z, v) = \tau(T, Z, v')$, then $T(v)$ and $T(v')$ interchangeable.

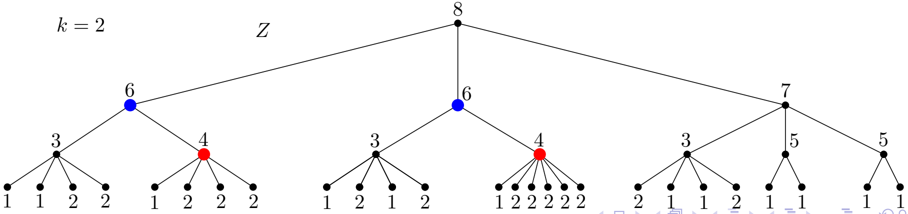
Same type: same "variety of traces among children in Z" $\rightarrow$ recursive!

Problem: #children may be unbounded, and we want #types $\leq f(k)$.

Lemma

Let σ be an ℓ-rotation sequence from T to T', for some $\ell \leq k$. For every vertex $v \in V(T)$, there are at most k vertices $u_1, \ldots, u_k \in \mathrm{children}(T, v)$ such that σ uses a vertex in each of the rooted subtrees $T(u_1), \ldots, T(u_k)$.

Crucial: enough to keep track of "at least $k+1$", not the actual number.



Number of types bounded by a function of k

Lemma

$\{\tau(T, Z, v) \mid v \in V(Z)\}$ *has size bounded by a function* $g(k)$,

Number of types bounded by a function of k $\rightarrow$ marking

Lemma

$\{\tau(T, Z, v) \mid v \in V(Z)\}$ has size bounded by a function $g(k)$, with

$$g(k) = k^{2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}}, \text{ where the tower has height } \operatorname{diam}(Z) = \mathcal{O}(k^2).$$

Number of types bounded by a function of $k \to$ marking

Lemma

$\{\tau(T, Z, v) \mid v \in V(Z)\}$ has size bounded by a function $g(k)$, with

$$g(k) = k^{2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}}, \text{ where the tower has height } \operatorname{diam}(Z) = \mathcal{O}(k^2).$$

$\boxed{\text{Marking algorithm}}$: for every vertex $v \in V(Z)$, pre-mark up to $k+1$ children of each type, and then prune from the root.

Number of types bounded by a function of k → marking

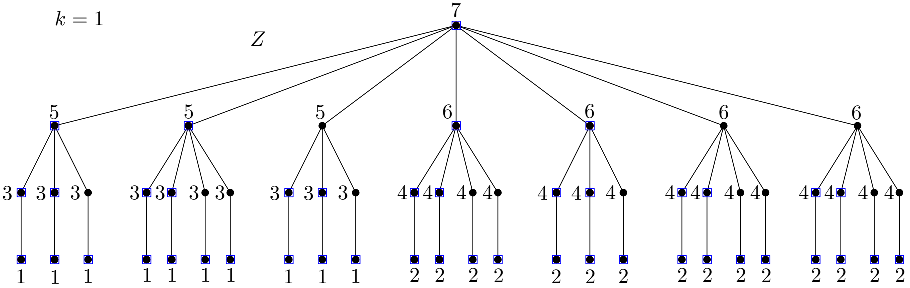
Lemma

$\{\tau(T, Z, v) \mid v \in V(Z)\}$ has size bounded by a function $g(k)$, with

$$g(k) = k^{2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}},$$ *where the tower has height $\mathrm{diam}(Z) = \mathcal{O}(k^2)$.*

$\boxed{\text{Marking algorithm}}$: for every vertex $v \in V(Z)$, pre-mark up to $k+1$ children of each type, and then prune from the root.

skip



Number of types bounded by a function of k $\rightarrow$ marking

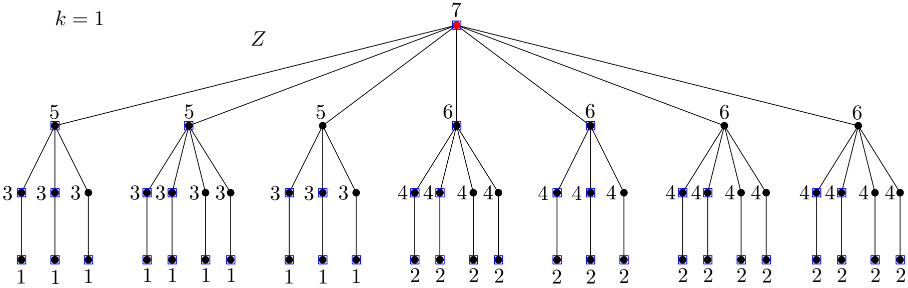
Lemma

$\{\tau(T, Z, v) \mid v \in V(Z)\}$ *has size bounded by a function* $g(k)$, *with*

$$g(k) = k^{2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}},$$ *where the tower has height* $\mathrm{diam}(Z) = \mathcal{O}(k^2)$.

$\boxed{\text{Marking algorithm}}$: for every vertex $v \in V(Z)$, pre-mark up to $k+1$ children of each type, and then prune from the root.

skip



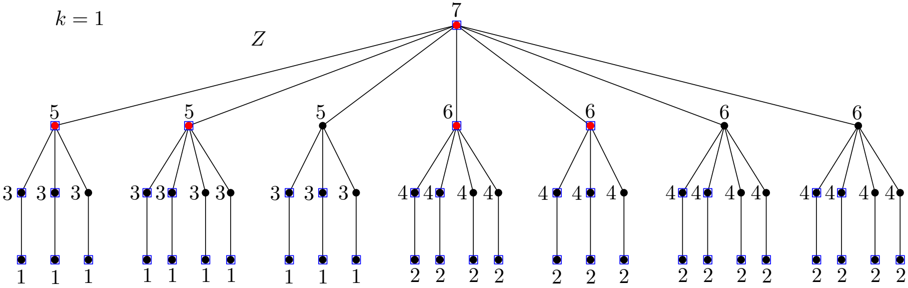
Lemma

$\{\tau(T,Z,v) \mid v \in V(Z)\}$ *has size bounded by a function* $g(k)$, *with*

$$g(k) = k^{2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}}, \text{ where the tower has height } \mathrm{diam}(Z) = \mathcal{O}(k^2).$$

$\boxed{\text{Marking algorithm}}$: for every vertex $v \in V(Z)$, pre-mark up to $k+1$ children of each type, and then prune from the root.

skip



Number of types bounded by a function of k $\rightarrow$ marking

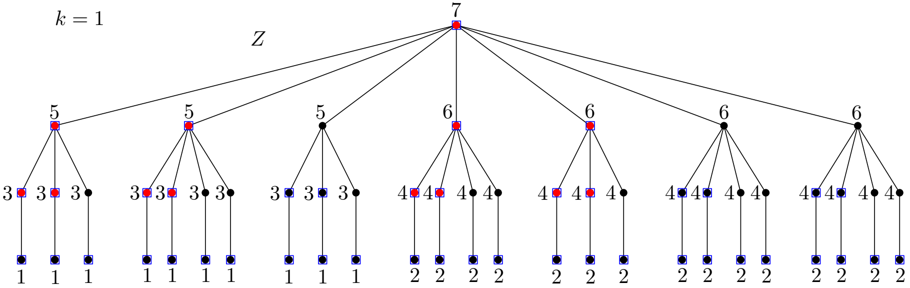
Lemma

$\{\tau(T, Z, v) \mid v \in V(Z)\}$ *has size bounded by a function* $g(k)$, *with*

$$g(k) = k^{2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}},$$ *where the tower has height* $\mathrm{diam}(Z) = \mathcal{O}(k^2)$.

Marking algorithm: for every vertex $v \in V(Z)$, pre-mark up to $k+1$ children of each type, and then prune from the root.

skip



Number of types bounded by a function of k → marking

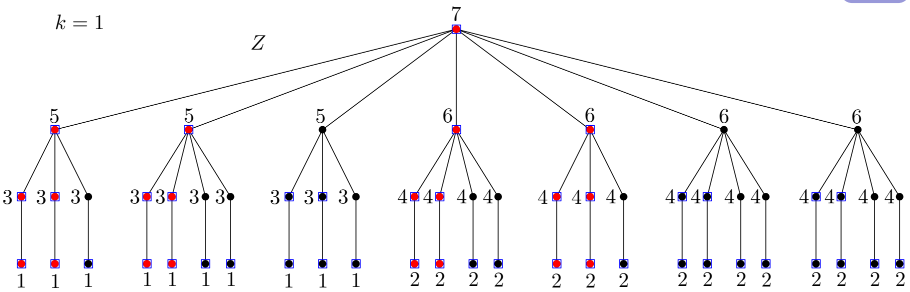
Lemma

$\{\tau(T, Z, v) \mid v \in V(Z)\}$ *has size bounded by a function* $g(k)$, *with*

$$g(k) = k^{2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}}, \text{ where the tower has height } \mathrm{diam}(Z) = \mathcal{O}(k^2).$$

$\boxed{\text{Marking algorithm}}$: for every vertex $v \in V(Z)$, pre-mark up to $k+1$ children of each type, and then prune from the root.

skip



The set of marked vertices satisfies what we want

Lemma

The set $M \subseteq V(T)$ *of marked vertices has size bounded by a function* $h(k)$*, with the same asymptotic growth as the function* $g(k)$ *given by the number of types. Moreover,* M *can be computed in time* $h(k) \cdot |V(G)|$*.*

The set of marked vertices satisfies what we want

Lemma

The set $M \subseteq V(T)$ of marked vertices has size bounded by a function $h(k)$, with the same asymptotic growth as the function $g(k)$ given by the number of types. Moreover, M can be computed in time $h(k) \cdot |V(G)|$.

Main technical lemma:

Lemma

If $\mathrm{dist}(T, T') \leq k$, then there exists an ℓ-rotation sequence from T to T', with $\ell \leq k$, using only vertices in M.

skip

The set of marked vertices satisfies what we want

Lemma

The set $M \subseteq V(T)$ of marked vertices has size bounded by a function $h(k)$, with the same asymptotic growth as the function $g(k)$ given by the number of types. Moreover, M can be computed in time $h(k) \cdot |V(G)|$.

Main technical lemma:

Lemma

If $\text{dist}(T, T') \leq k$, then there exists an ℓ-rotation sequence from T to T', with $\ell \leq k$, using only vertices in M.

skip

Scheme of the proof:

- Let σ be an ℓ-rotation sequence from T to T' minimizing the number of non-marked vertices used by σ.

The set of marked vertices satisfies what we want

Lemma

The set $M \subseteq V(T)$ of marked vertices has size bounded by a function $h(k)$, with the same asymptotic growth as the function $g(k)$ given by the number of types. Moreover, M can be computed in time $h(k) \cdot |V(G)|$.

Main technical lemma:

Lemma

If $\text{dist}(T, T') \leq k$, then there exists an ℓ-rotation sequence from T to T', with $\ell \leq k$, using only vertices in M.

skip

Scheme of the proof:

- Let σ be an ℓ-rotation sequence from T to T' minimizing the number of non-marked vertices used by σ.
- Goal: define another sequence σ' using $<$ non-marked vertices.

The set of marked vertices satisfies what we want

Lemma

The set $M \subseteq V(T)$ of marked vertices has size bounded by a function $h(k)$, with the same asymptotic growth as the function $g(k)$ given by the number of types. Moreover, M can be computed in time $h(k) \cdot |V(G)|$.

Main technical lemma:

Lemma

If $\text{dist}(T, T') \leq k$, then there exists an ℓ-rotation sequence from T to T', with $\ell \leq k$, using only vertices in M.

skip

Scheme of the proof:

- Let σ be an ℓ-rotation sequence from T to T' minimizing the number of non-marked vertices used by σ.
- Goal: define another sequence σ' using $<$ non-marked vertices.
- Let $v \in V(T)$ be a downmost non-marked vertex used by σ.

The set of marked vertices satisfies what we want

Lemma

The set $M \subseteq V(T)$ of marked vertices has size bounded by a function $h(k)$, with the same asymptotic growth as the function $g(k)$ given by the number of types. Moreover, M can be computed in time $h(k) \cdot |V(G)|$.

Main technical lemma:

Lemma

If $\text{dist}(T, T') \leq k$, then there exists an ℓ-rotation sequence from T to T', with $\ell \leq k$, using only vertices in M.

skip

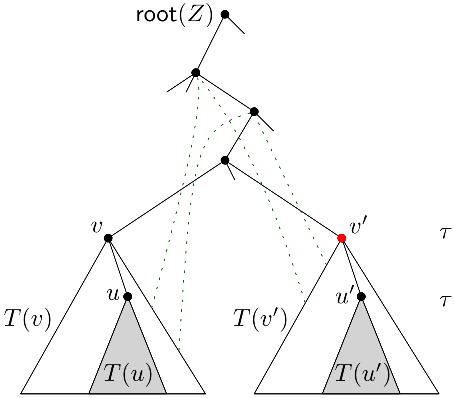
Scheme of the proof:

- Let σ be an ℓ-rotation sequence from T to T' minimizing the number of non-marked vertices used by σ.
- Goal: define another sequence σ' using $<$ non-marked vertices.
- Let $v \in V(T)$ be a downmost non-marked vertex used by σ.
- We distinguish two cases...

skip

Proof of the main technical lemma: Case 1

If v has a marked (non-used) T-sibling v' with $\tau(T, Z, v) = \tau(T, Z, v')$:

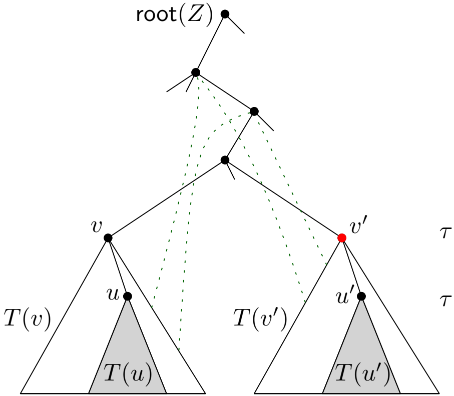


$\tau(T, Z, v) = \tau(T, Z, v')$

$\tau(T, Z, u) = \tau(T, Z, u')$

Proof of the main technical lemma: Case 1

If v has a marked (non-used) T-sibling v' with $\tau(T, Z, v) = \tau(T, Z, v')$:



$$\tau(T, Z, v) = \tau(T, Z, v')$$

$$\tau(T, Z, u) = \tau(T, Z, u')$$

We define σ' from σ by just replacing v with v' in all the rotations of σ involving v.

skip

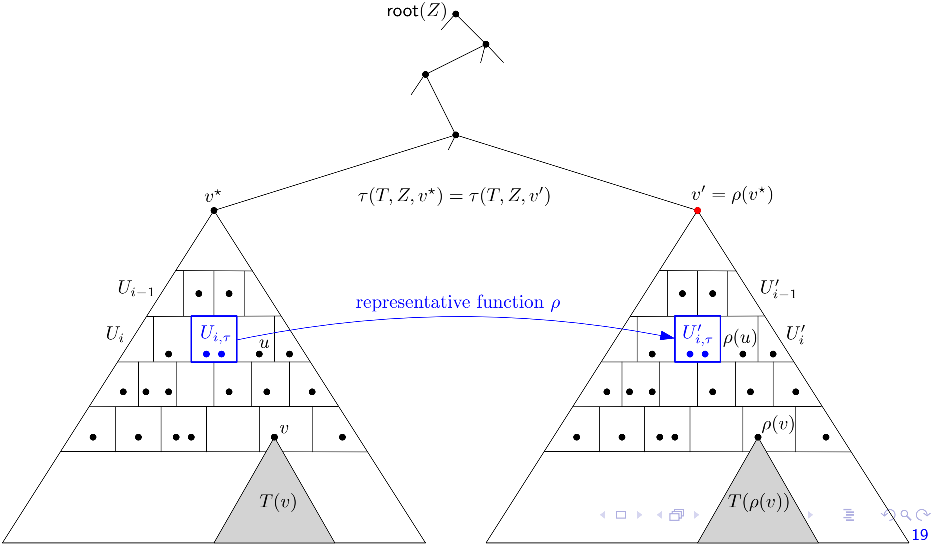
Proof of the main technical lemma: Case 2

All T-siblings v' of v with $\tau(T, Z, v) = \tau(T, Z, v')$ are non-marked.

Proof of the main technical lemma: Case 2

All T-siblings v' of v with $\tau(T, Z, v) = \tau(T, Z, v')$ are non-marked.

In this case, to define σ', we need to modify σ in a more global way:



Conclusions and further research

Theorem

The ROTATION DISTANCE *problem can be solved in time* $f(k) \cdot |V(G)|$,

$$\text{with} \quad f(k) = k^{k \cdot 2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}},$$

where the tower of exponentials has height at most $(3k+1)4k = \mathcal{O}(k^2)$.

Conclusions and further research

Theorem

The ROTATION DISTANCE *problem can be solved in time* $f(k) \cdot |V(G)|$,

$$\text{with} \quad f(k) = k^{k \cdot 2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}},$$

where the tower of exponentials has height at most $(3k+1)4k = \mathcal{O}(k^2)$.

It should be possible to improve $f(k)$ (dominated by the number of types).

Conclusions and further research

Theorem

The ROTATION DISTANCE *problem can be solved in time* $f(k) \cdot |V(G)|$,

$$\text{with} \quad f(k) = k^{k \cdot 2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}},$$

where the tower of exponentials has height at most $(3k+1)4k = \mathcal{O}(k^2)$.

It should be possible to improve $f(k)$ (dominated by the number of types).

ROTATION DISTANCE	paths	general graphs
NP-hard	open	✓
FPT	✓	✓[this article]
Polynomial kernel	✓	open

Conclusions and further research

Theorem

The ROTATION DISTANCE *problem can be solved in time* $f(k) \cdot |V(G)|$,

$$\text{with} \quad f(k) = k^{k \cdot 2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathcal{O}(k^2)}}}}}}},$$

where the tower of exponentials has height at most $(3k+1)4k = \mathcal{O}(k^2)$.

It should be possible to improve $f(k)$ (dominated by the number of types).

ROTATION DISTANCE	paths	general graphs
NP-hard	open	✓
FPT	✓	✓[this article]
Polynomial kernel	✓	open

COMBINATORIAL SHORTEST PATH ON POLYMATROIDS:

- NP-hard. [Ito, Kakimura, Kamiyama, Kobayashi, Maezawa, Nozaki, Okamoto. 2023]
- Is it also FPT?

Gràcies!