Lean tree-cut decompositions
obstructions & algorithms

Jean-Florent Raymond
(TU Berlin)

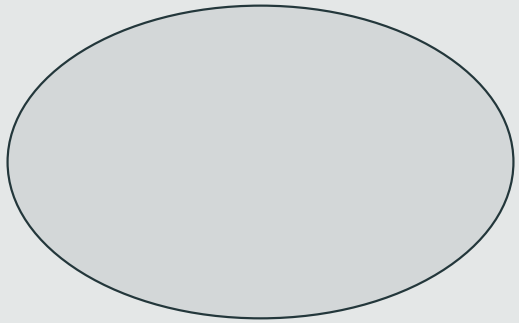
joint work with:

Archontia C. Giannopoulou
O-joung Kwon
Dimitrios M. Thilikos

1927: Menger's theorem

Theorem (Menger, 1927)

$\forall G$

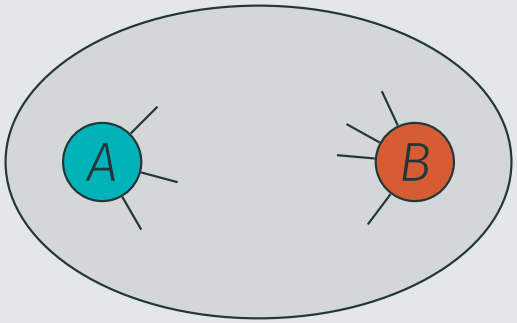


1927: Menger's theorem

Theorem (Menger, 1927)

$\forall G$,

$\forall A, B \subseteq V(G)$



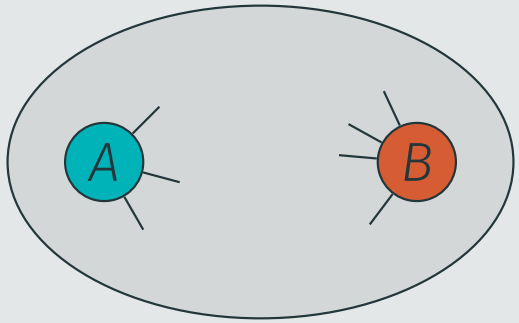
1927: Menger's theorem

Theorem (Menger, 1927)

$\forall G$,

$\forall A, B \subseteq V(G)$,

$\forall k \in \mathbb{N}$,



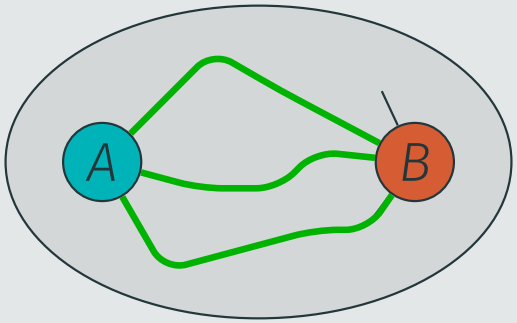
1927: Menger's theorem

Theorem (Menger, 1927)

$\forall G$,

$\forall A, B \subseteq V(G)$,

$\forall k \in \mathbb{N}$,



G has:

- *k vertex-disjoint A-B paths*

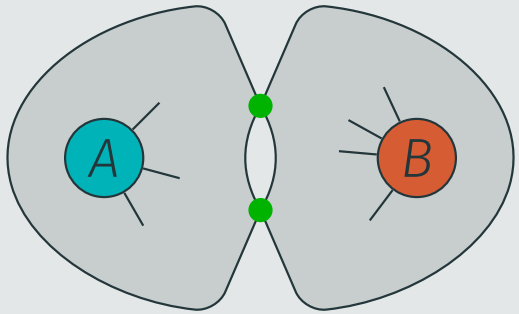
1927: Menger's theorem

Theorem (Menger, 1927)

$\forall G$,

$\forall A, B \subseteq V(G)$,

$\forall k \in \mathbb{N}$,



G has:

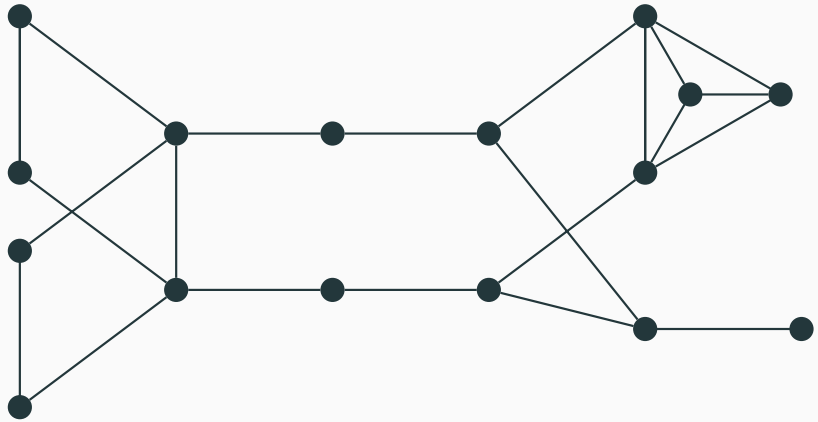
- *k vertex-disjoint A-B paths*
- *or an A-B separator of size $< k$*

1990: Thomas' theorem

A Menger-like property of treewidth: The finite case. Thomas. JCTB'90.

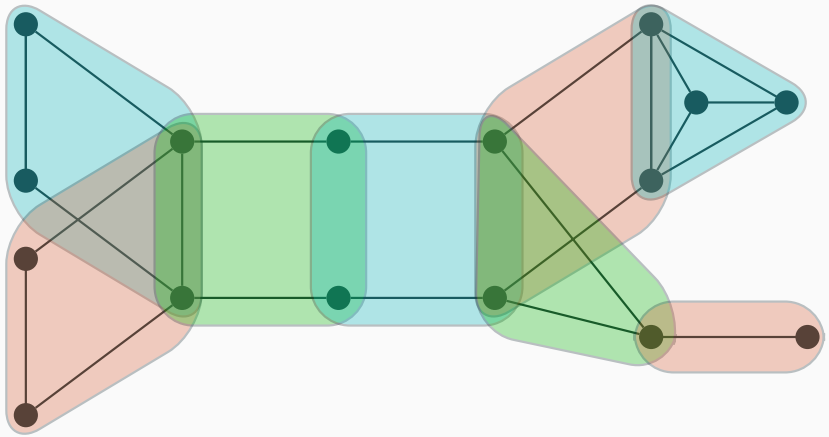
1990: Thomas' theorem

A Menger-like property of treewidth: The finite case. Thomas. JCTB'90.



1990: Thomas' theorem

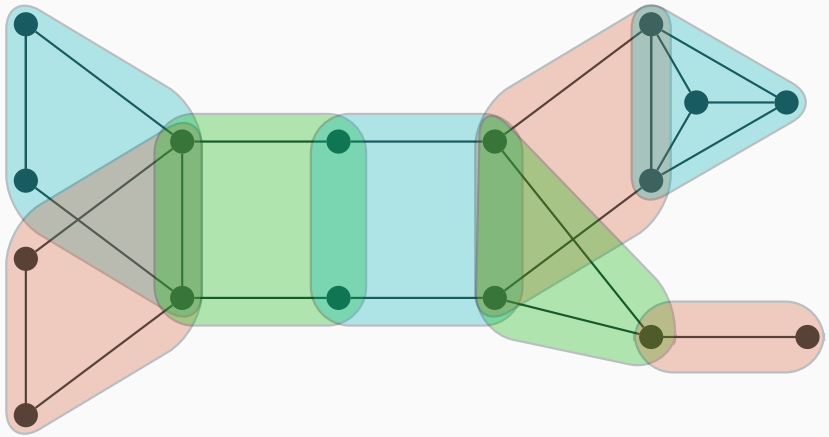
A Menger-like property of treewidth: The finite case. Thomas. JCTB'90.



tree decomposition: tree-like decomposition into *bags*

1990: Thomas' theorem

A Menger-like property of treewidth: The finite case. Thomas. JCTB'90.

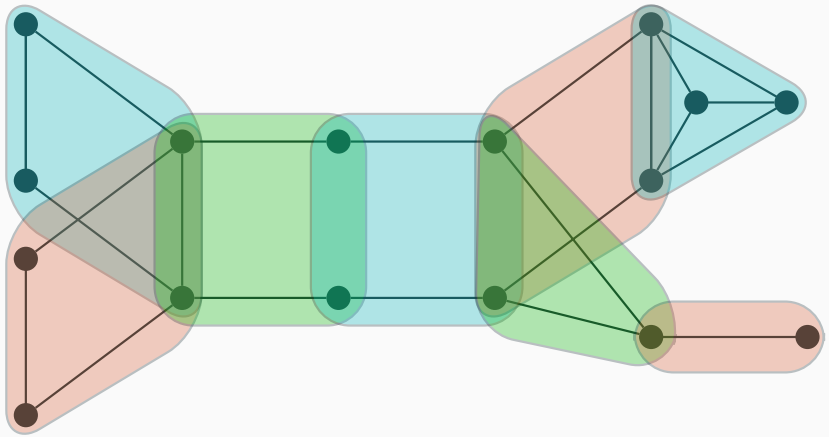


tree decomposition: tree-like decomposition into *bags*

optimal tree-decomposition: one minimizing bag size

1990: Thomas' theorem

A Menger-like property of treewidth: The finite case. Thomas. JCTB'90.



tree decomposition: tree-like decomposition into *bags*

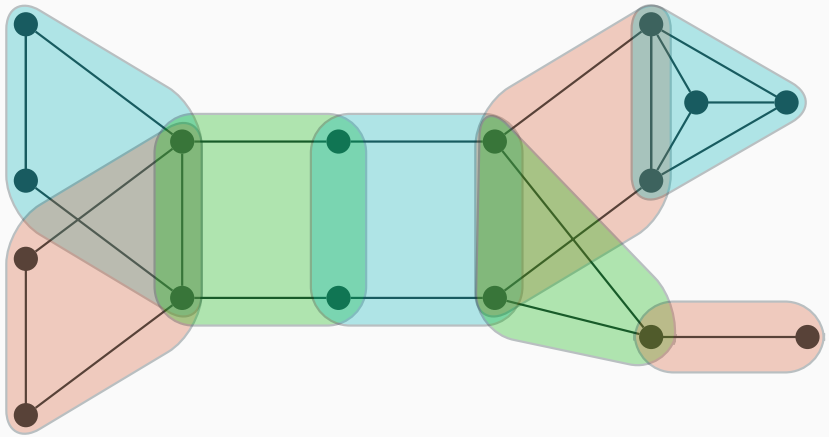
optimal tree-decomposition: one minimizing bag size

Theorem (Thomas, 1990)

Every graph G admits an optimal tree decomposition $(T, \mathcal{X})$ s.t.

1990: Thomas' theorem

A Menger-like property of treewidth: The finite case. Thomas. JCTB'90.



tree decomposition: tree-like decomposition into *bags*

optimal tree-decomposition: one minimizing bag size

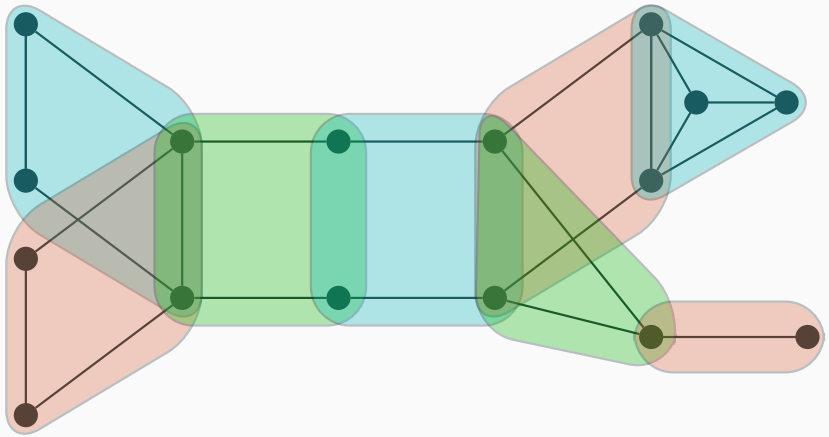
Theorem (Thomas, 1990)

Every graph G admits an optimal tree decomposition $(T, \mathcal{X})$ s.t.

$\forall a, b \in V(T)$

1990: Thomas' theorem

A Menger-like property of treewidth: The finite case. Thomas. JCTB'90.



tree decomposition: tree-like decomposition into *bags*

optimal tree-decomposition: one minimizing bag size

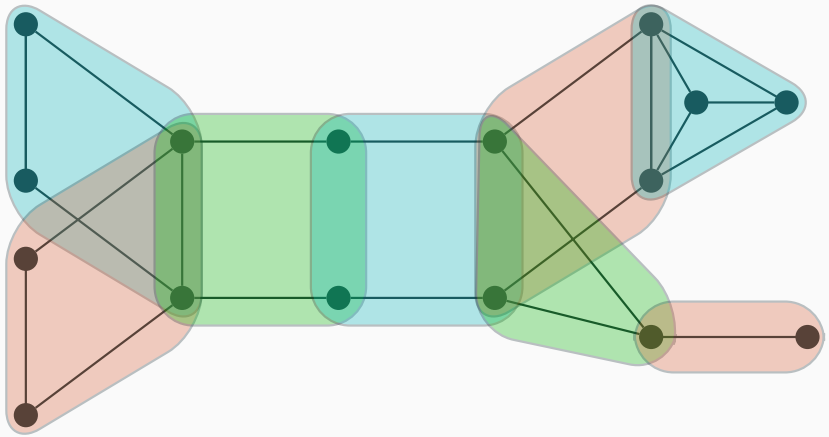
Theorem (Thomas, 1990)

Every graph G admits an optimal tree decomposition $(T, \mathcal{X})$ s.t.

$\forall a, b \in V(T), \quad \forall A \subseteq \text{bag}(a), B \subseteq \text{bag}(b)$ *with* $|A| = |B|$,

1990: Thomas' theorem

A Menger-like property of treewidth: The finite case. Thomas. JCTB'90.



tree decomposition: tree-like decomposition into *bags*

optimal tree-decomposition: one minimizing bag size

Theorem (Thomas, 1990)

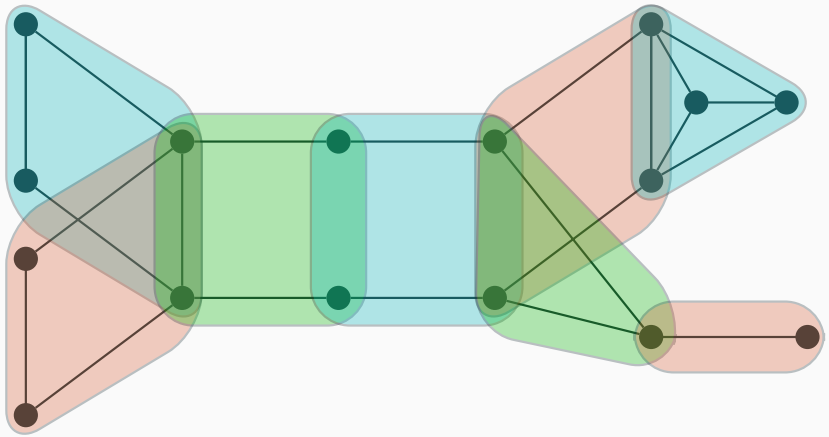
Every graph G admits an optimal tree decomposition $(T, \mathcal{X})$ s.t.

$\forall a, b \in V(T), \quad \forall A \subseteq \text{bag}(a), B \subseteq \text{bag}(b)$ *with* $|A| = |B|$,

- *there are $|A|$ disjoint paths linking A to B,*

1990: Thomas' theorem

A Menger-like property of treewidth: The finite case. Thomas. JCTB'90.



tree decomposition: tree-like decomposition into *bags*

optimal tree-decomposition: one minimizing bag size

Theorem (Thomas, 1990)

Every graph G admits an optimal tree decomposition $(T, \mathcal{X})$ s.t.

$\forall a, b \in V(T), \quad \forall A \subseteq \text{bag}(a), B \subseteq \text{bag}(b)$ *with* $|A| = |B|$,

- *there are $|A|$ disjoint paths linking A to B,*
- *or there is a node c between a and b with $|\text{bag}(c)| < |A|$.*

Applications of leanness

- bounding the size of obstructions
- algorithms

Applications of leanness

- bounding the size of obstructions
- algorithms
- well-quasi-ordering
- extremal graph theory

Applications of leanness

- bounding the size of obstructions
- algorithms
- well-quasi-ordering
- extremal graph theory

Other parameters with lean decompositions:

θ-treewidth, pathwidth (directed or not), DAG-width, rank-width (linear or not), profile-width, block-width, matroid treewidth, matroid branchwidth, etc.

Applications of leanness

- bounding the size of obstructions
- algorithms
- well-quasi-ordering
- extremal graph theory

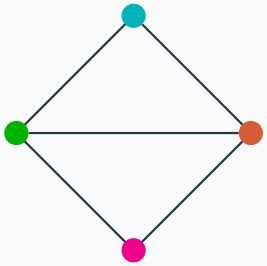
Other parameters with lean decompositions:

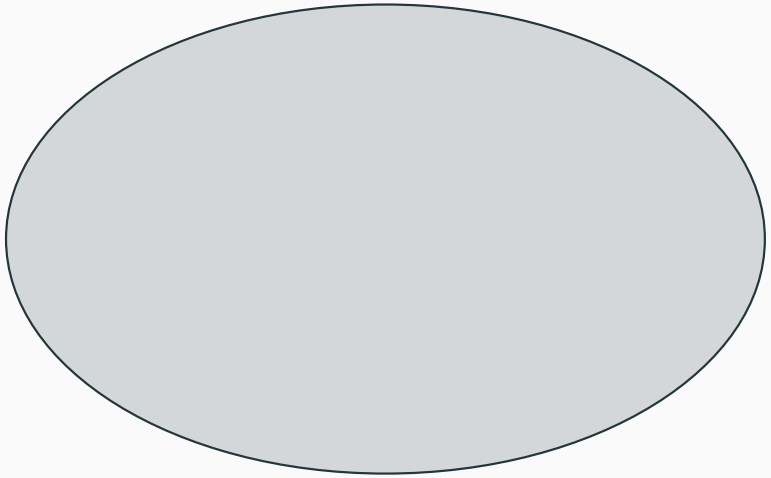
θ-treewidth, pathwidth (directed or not), DAG-width, rank-width (linear or not), profile-width, block-width, matroid treewidth, matroid branchwidth, etc.

In this talk: tree-cut width

Immersions

H





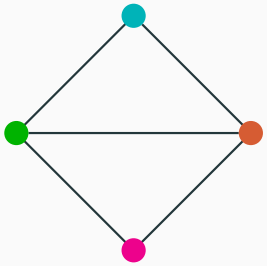
G

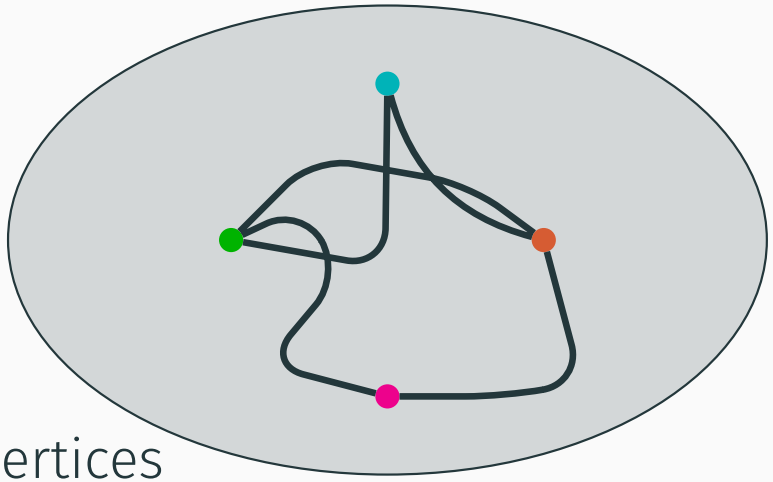
Immersions

distinct vertices $\mapsto$ distinct vertices

Immersions

H





G

distinct vertices $\mapsto$ distinct vertices

distinct edges $\mapsto$ edge-disjoint paths

Immersions

H is an *immersion* of *G*

distinct vertices $\mapsto$ distinct vertices

distinct edges $\mapsto$ edge-disjoint paths

Immersions

H is an *immersion* of *G*

distinct vertices $\mapsto$ distinct vertices

distinct edges $\mapsto$ edge-disjoint paths

Minors vs. immersions:

Immersions

H is an *immersion* of *G*

distinct vertices $\mapsto$ distinct vertices

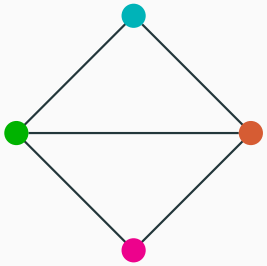
distinct edges $\mapsto$ edge-disjoint paths

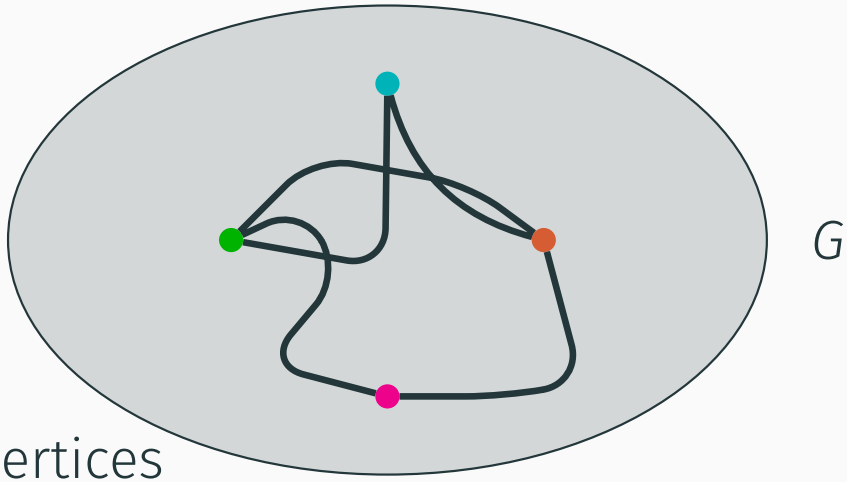
Minors vs. immersions:

	minors	immersions
well-quasi-ordering	✓ [RS 2004]	✓ [RS 2010]

Immersions

H is an *immersion* of *G*





distinct vertices $\mapsto$ distinct vertices

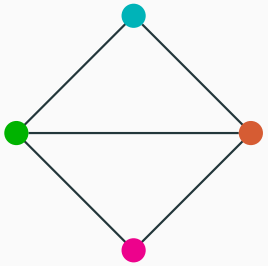
distinct edges $\mapsto$ edge-disjoint paths

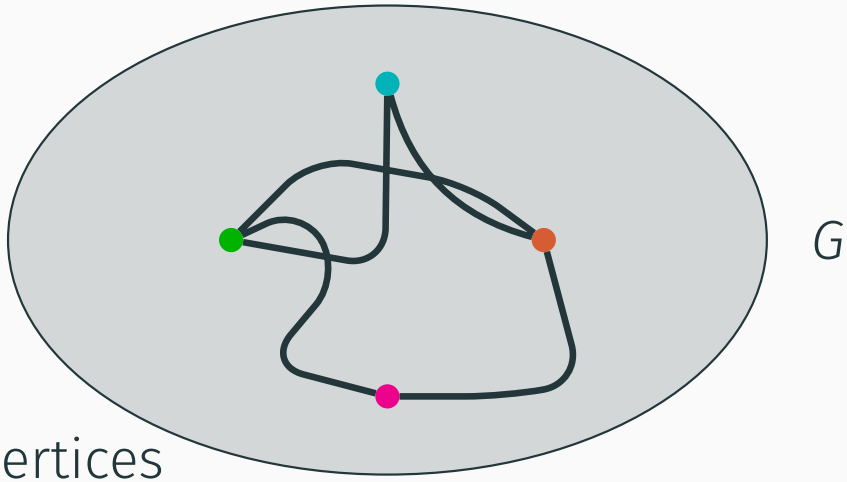
Minors vs. immersions:

	minors	immersions
well-quasi-ordering	✓ [RS 2004]	✓ [RS 2010]
excluded wall theorem	✓ [RS 1986]	✓ [Wollan 2015]

Immersions

H is an *immersion* of *G*





distinct vertices $\mapsto$ distinct vertices

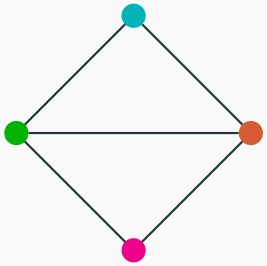
distinct edges $\mapsto$ edge-disjoint paths

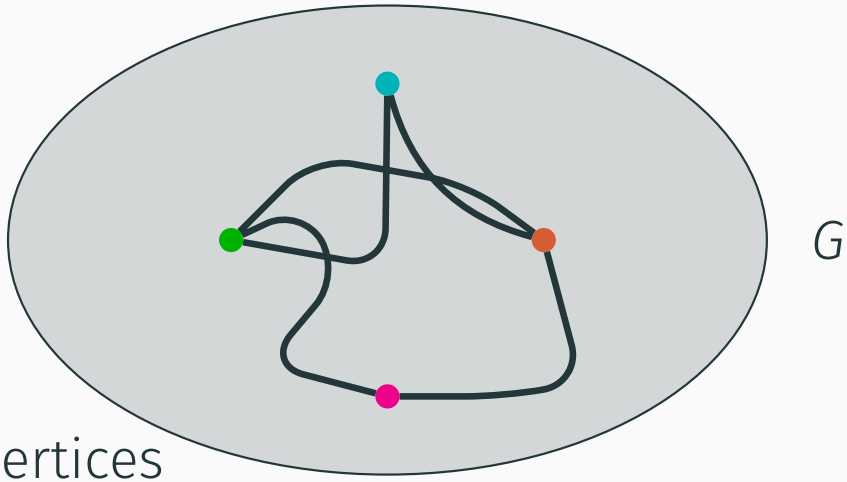
Minors vs. immersions:

	minors	immersions
well-quasi-ordering	✓ [RS 2004]	✓ [RS 2010]
excluded wall theorem	✓ [RS 1986]	✓ [Wollan 2015]
“nice” parameter	treewidth	tree-cut width

Immersions

H is an *immersion* of *G*





distinct vertices $\mapsto$ distinct vertices

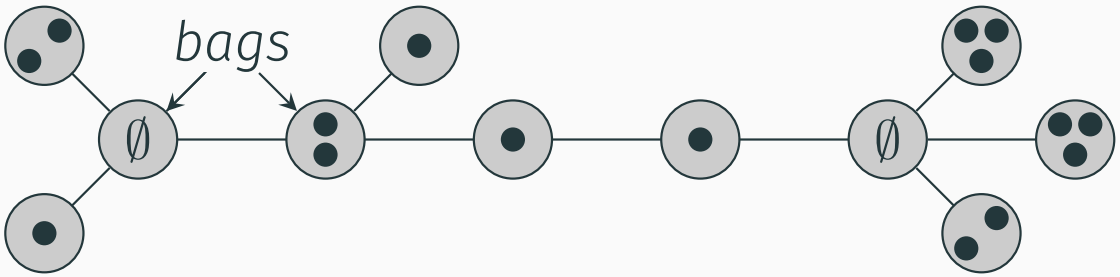
distinct edges $\mapsto$ edge-disjoint paths

Minors vs. immersions:

	minors	immersions
well-quasi-ordering	✓ [RS 2004]	✓ [RS 2010]
excluded wall theorem	✓ [RS 1986]	✓ [Wollan 2015]
"nice" parameter	treewidth	tree-cut width

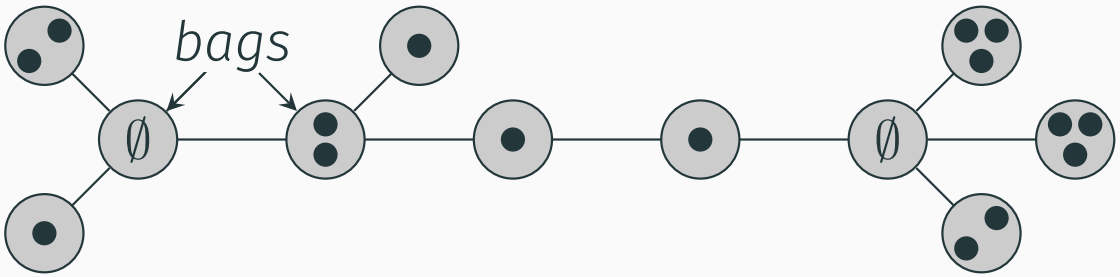
Immersions are much less studied than minors!

Tree-cut decompositions



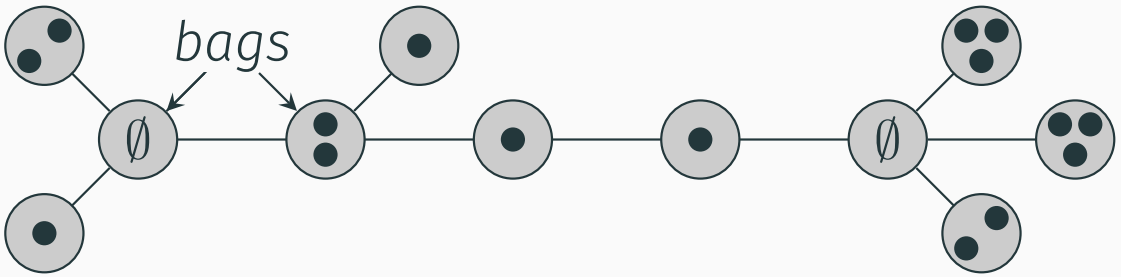
tree-cut decomposition:
partition of $V(G)$ into *bags*

Tree-cut decompositions



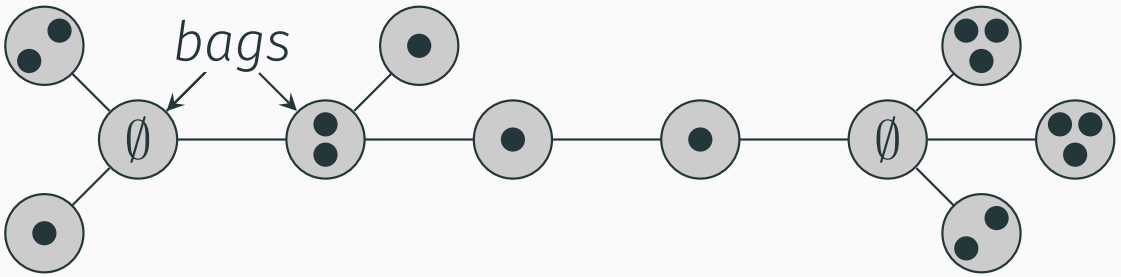
tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags

Tree-cut decompositions



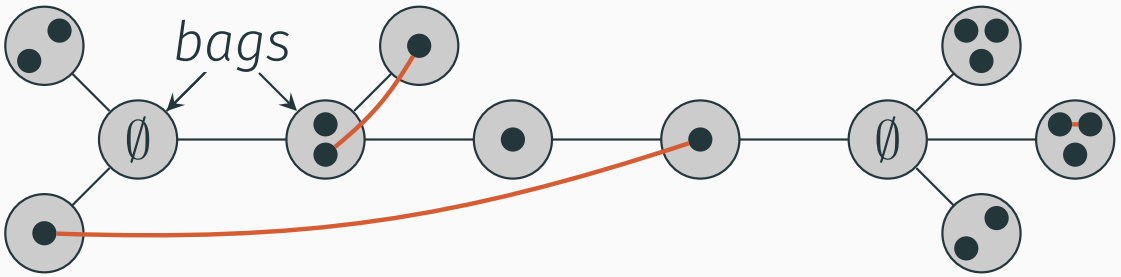
tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

Tree-cut decompositions



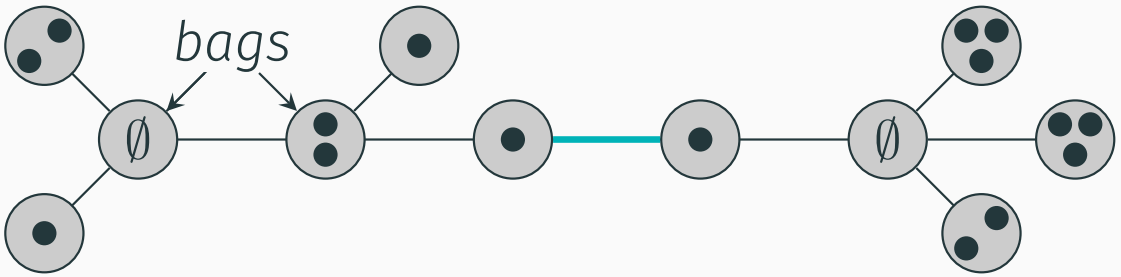
tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

Tree-cut decompositions



tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

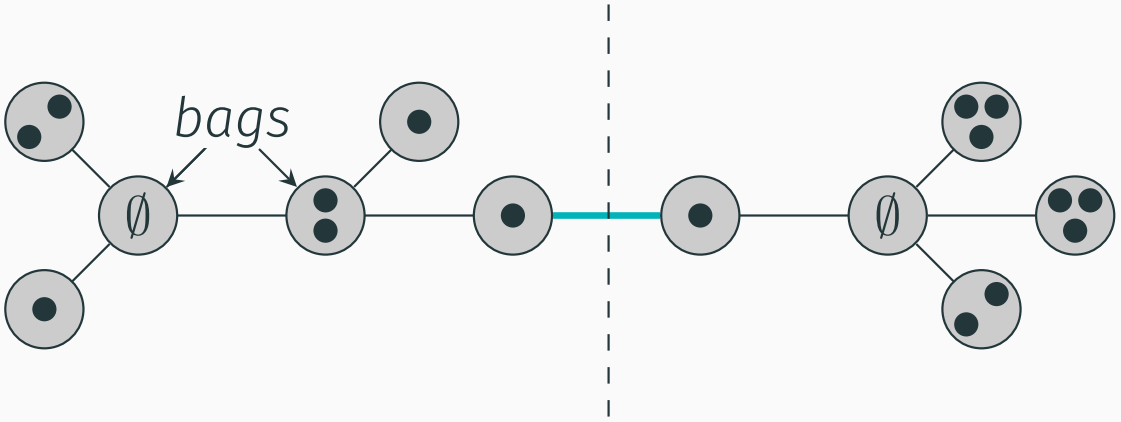
Tree-cut decompositions



tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

- **adhesion of $e \in V(T)$:** edges of G between the two parts of $T - e$

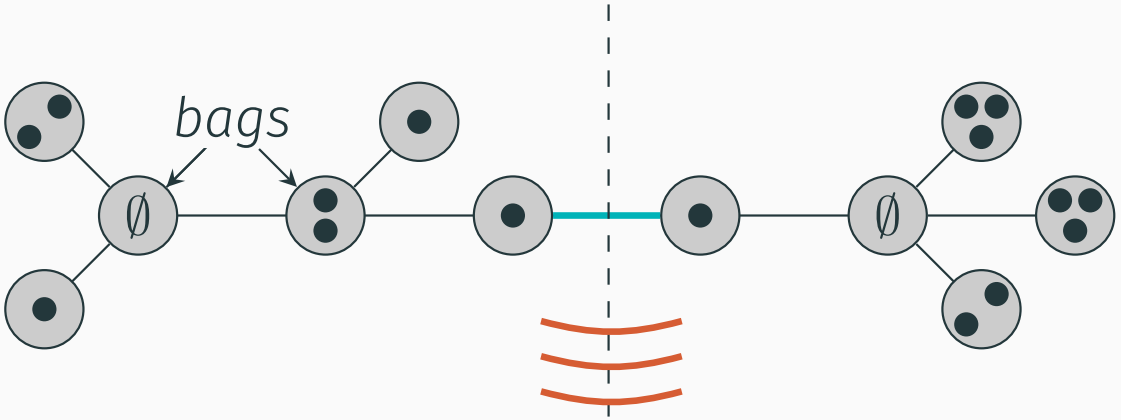
Tree-cut decompositions



tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

- **adhesion of $e \in V(T)$:** edges of G between the two parts of $T - e$

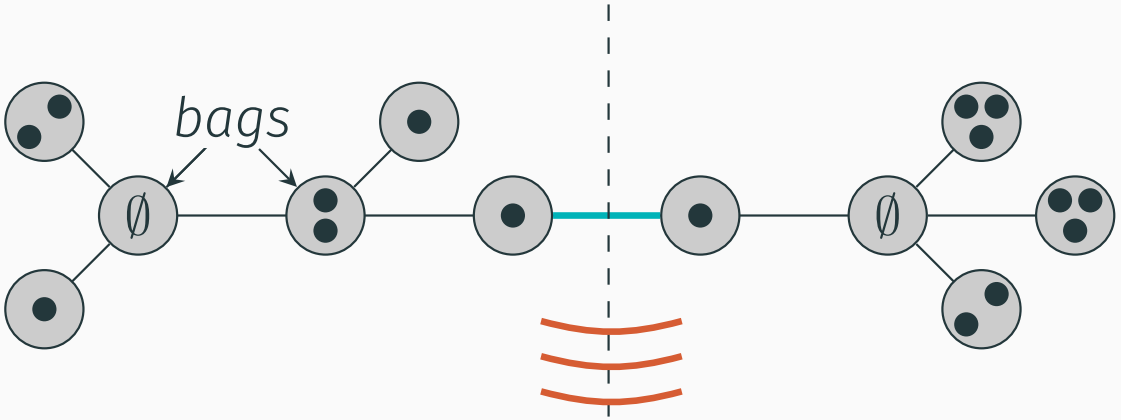
Tree-cut decompositions



tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

- **adhesion of** $e \in V(T)$**:** edges of G between the two parts of $T - e$

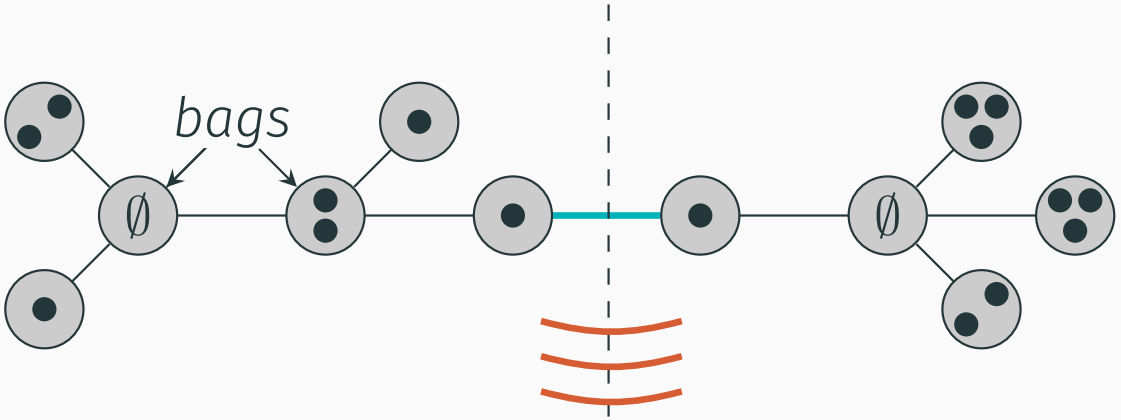
Tree-cut decompositions



tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

- **adhesion of** $e \in V(T)$**:** edges of G between the two parts of $T - e$
- **width of the decomposition** for 3-edge-connected graphs**:**

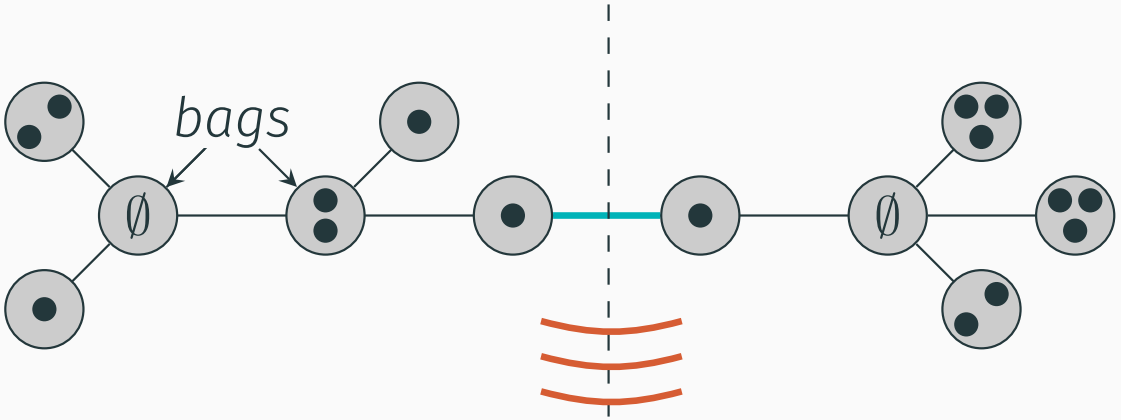
Tree-cut decompositions



tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

- **adhesion of** $e \in V(T)$**:** edges of G between the two parts of $T - e$
- **width of the decomposition** for 3-edge-connected graphs**:**
  $\max\left\{\max_{r \in E(T)} |\mathrm{adh}(e)|, \quad \max_{v \in V(T)}(|\mathrm{bag}(v)| + \Delta(v))\right\}$

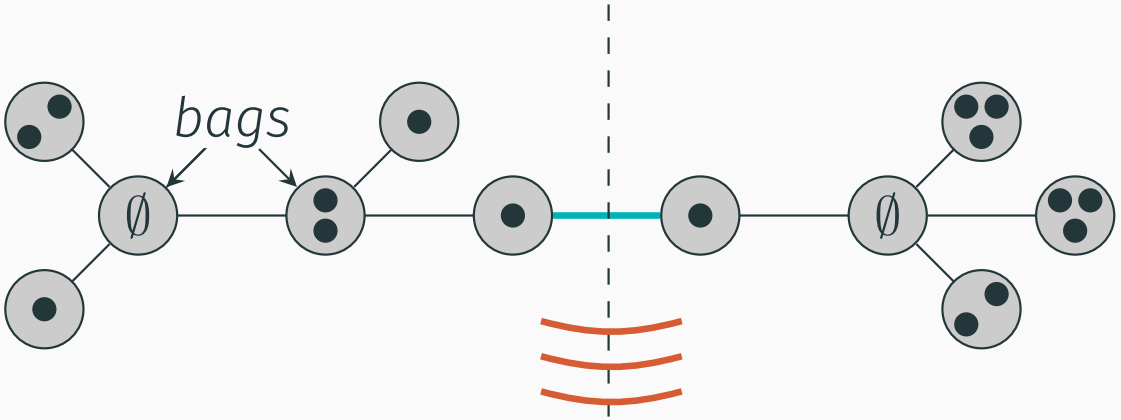
Tree-cut decompositions



tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

- **adhesion of** $e \in V(T)$**:** edges of G between the two parts of $T - e$
- **width of the decomposition** for 3-edge-connected graphs**:**
  $\max\left\{\max_{r \in E(T)} |\mathrm{adh}(e)|, \quad \max_{v \in V(T)}(|\mathrm{bag}(v)| + \Delta(v))\right\}$
  grows with adhesion size, bag size, and degree in T

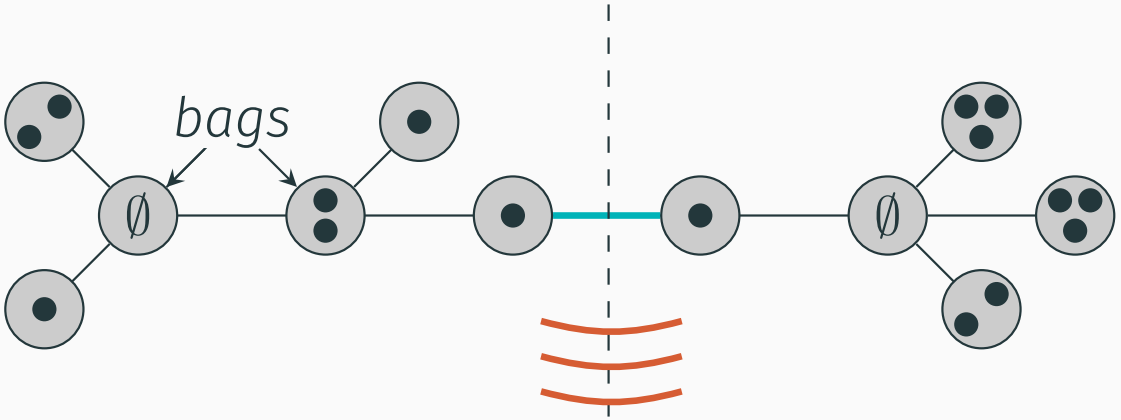
Tree-cut decompositions



tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

- **adhesion of** $e \in V(T)$**:** edges of G between the two parts of $T - e$
- **width of the decomposition** for 3-edge-connected graphs**:**
  $\max\left\{\max_{r \in E(T)} |\mathrm{adh}(e)|, \quad \max_{v \in V(T)}(|\mathrm{bag}(v)| + \Delta(v))\right\}$
  grows with adhesion size, bag size, and degree in T
- **tcw**(G)**:** min width of a decomposition of G

Tree-cut decompositions

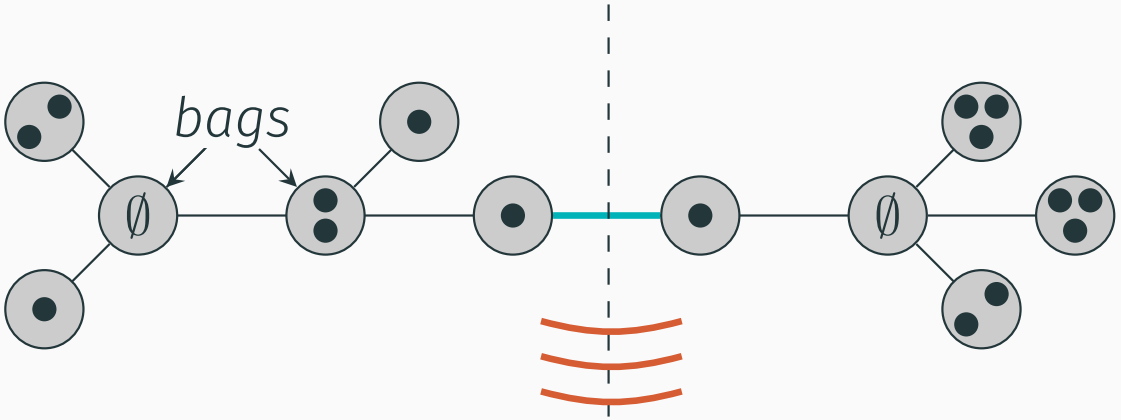


tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

- **adhesion of** $e \in V(T)$**:** edges of G between the two parts of $T - e$
- **width of the decomposition** for 3-edge-connected graphs**:**
  $\max\left\{\max_{r \in E(T)} |\mathrm{adh}(e)|, \quad \max_{v \in V(T)}(|\mathrm{bag}(v)| + \Delta(v))\right\}$
  grows with adhesion size, bag size, and degree in T
- **tcw**(G)**:** min width of a decomposition of G

Main message: **tcw** $\sim$ edge version of **tw**

Tree-cut decompositions

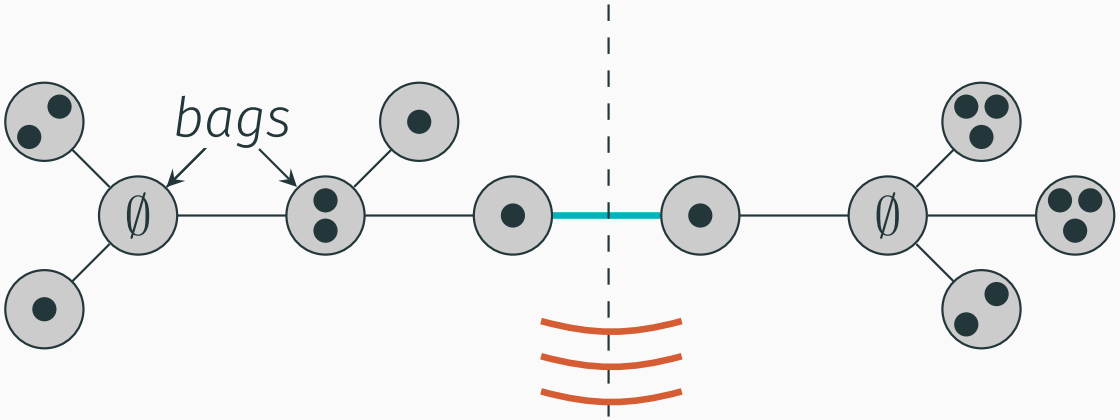


tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

- **adhesion of** $e \in V(T)$**:** edges of G between the two parts of $T - e$
- **width of the decomposition** for 3-edge-connected graphs**:**
  $\max\left\{\max_{r \in E(T)} |\mathrm{adh}(e)|, \quad \max_{v \in V(T)}(|\mathrm{bag}(v)| + \Delta(v))\right\}$
  grows with adhesion size, bag size, and degree in T
- **tcw**(G)**:** min width of a decomposition of G

Main message: **tcw** $\sim$ edge version of **tw**, shows small edge separators

Tree-cut decompositions



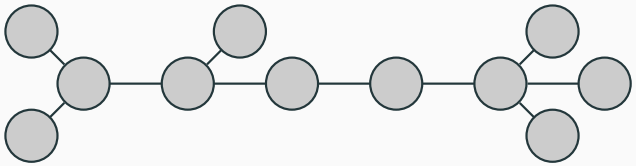
tree-cut decomposition:
partition of $V(G)$ into *bags*
+possibly some empty bags
organized along a tree T

- **adhesion of** $e \in V(T)$**:** edges of G between the two parts of $T - e$
- **width of the decomposition** for 3-edge-connected graphs**:**
  $\max\left\{\max_{r \in E(T)} |\mathrm{adh}(e)|, \quad \max_{v \in V(T)}(|\mathrm{bag}(v)| + \Delta(v))\right\}$
  grows with adhesion size, bag size, and degree in T
- **tcw**(G)**:** min width of a decomposition of G

Main message: **tcw** $\sim$ edge version of **tw**, shows small edge separators, well behaved wrt immersions

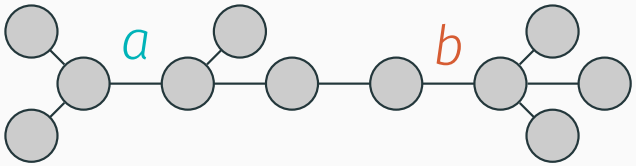
A Menger-like property of tree-cut width

A tree-cut decomposition $(T, \mathcal{X})$ is **lean** if:



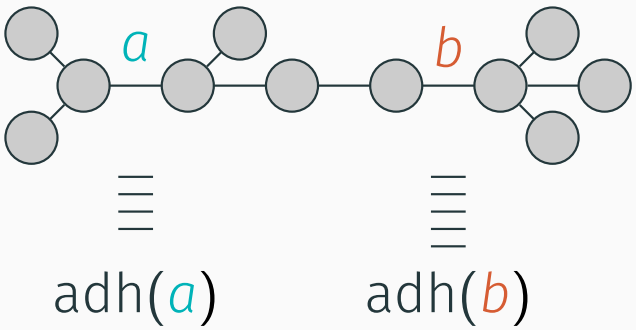
A Menger-like property of tree-cut width

A tree-cut decomposition $(T, \mathcal{X})$ is **lean** if:
$\forall a, b \in E(G)$



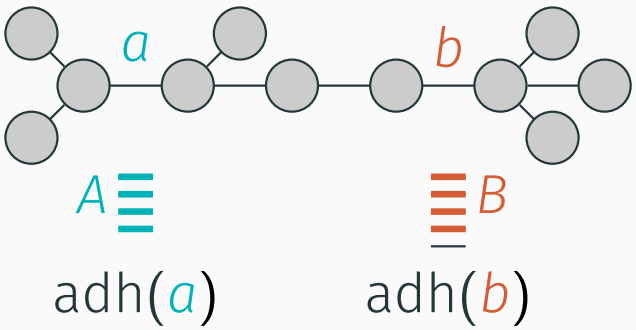
A Menger-like property of tree-cut width

A tree-cut decomposition $(T, \mathcal{X})$ is **lean** if:
$\forall a, b \in E(G)$



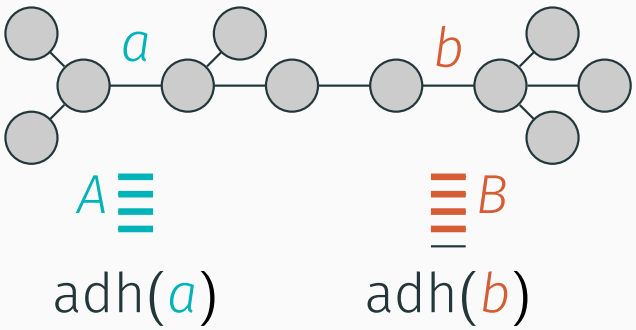
A Menger-like property of tree-cut width

A tree-cut decomposition $(T, \mathcal{X})$ is **lean** if:
$\forall a, b \in E(G)$,
$\forall A \subseteq \text{adh}(a), \forall B \subseteq \text{adh}(b)$ with $|A| = |B|$,



A Menger-like property of tree-cut width

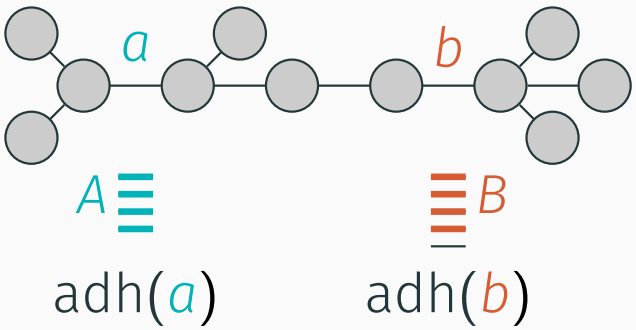
A tree-cut decomposition $(T, \mathcal{X})$ is **lean** if:
$\forall a, b \in E(G)$,
$\forall A \subseteq \text{adh}(a), \forall B \subseteq \text{adh}(b)$ with $|A| = |B|$,



- $|A|$ edge-disjoint A-B paths in G
- or an edge of T of adhesion $< |A|$ between a and b

A Menger-like property of tree-cut width

A tree-cut decomposition $(T, \mathcal{X})$ is **lean** if:
$\forall a, b \in E(G)$,
$\forall A \subseteq \text{adh}(a), \forall B \subseteq \text{adh}(b)$ with $|A| = |B|$,



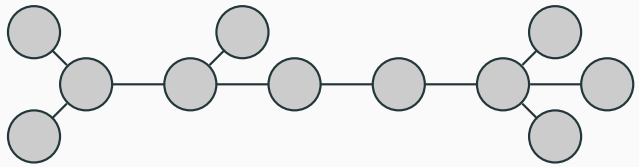
- $|A|$ edge-disjoint A-B paths in G
- or an edge of T of adhesion $< |A|$ between a and b

Theorem (Giannopoulou, Kwon, R., Thilikos, 2019)

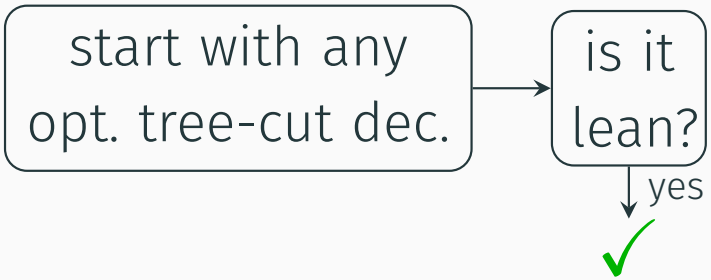
Every G has a tree-cut decomposition of width $\mathbf{tcw}(G)$ *that is lean.*

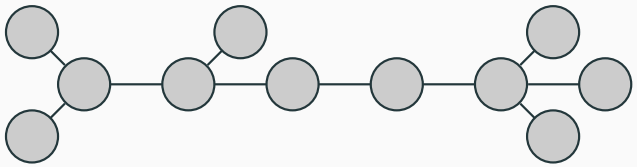
Every graph has a lean optimal tree-cut decomposition

start with any
opt. tree-cut dec.

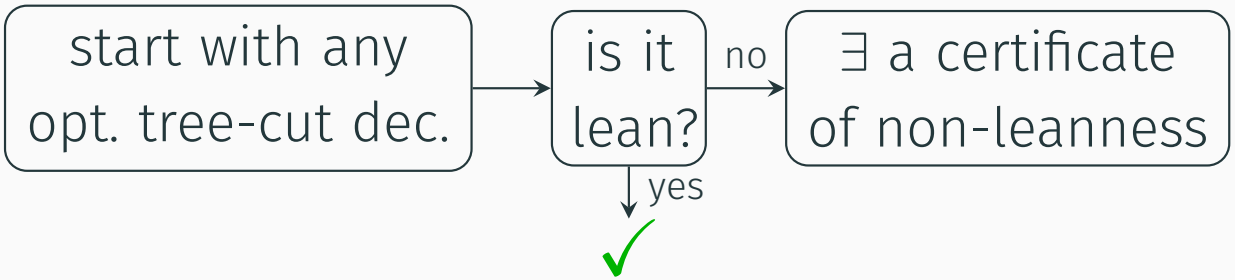


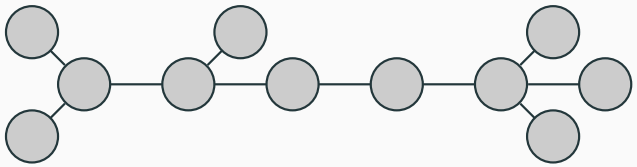
Every graph has a lean optimal tree-cut decomposition



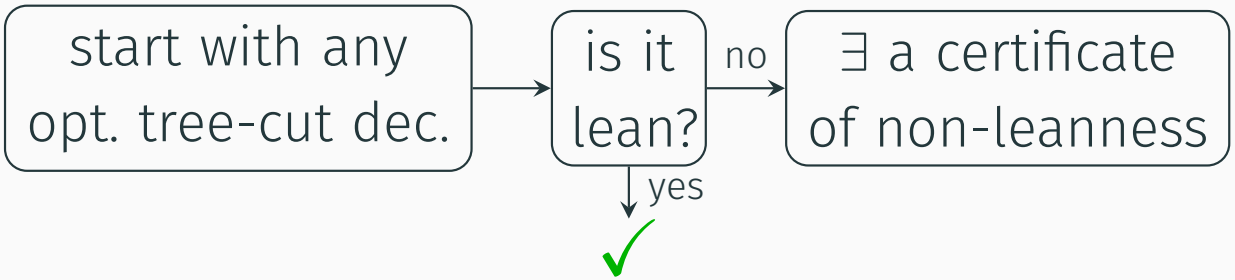


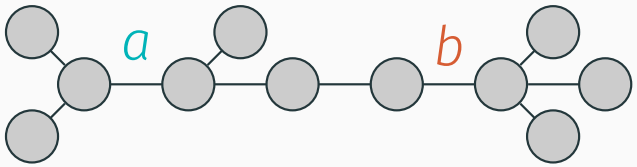
Every graph has a lean optimal tree-cut decomposition



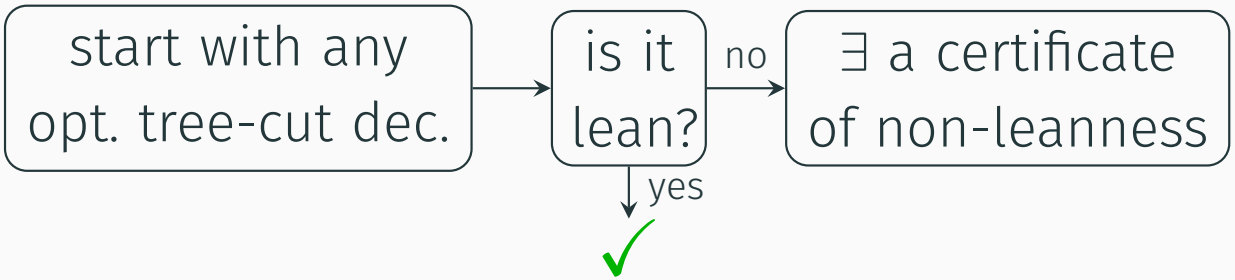


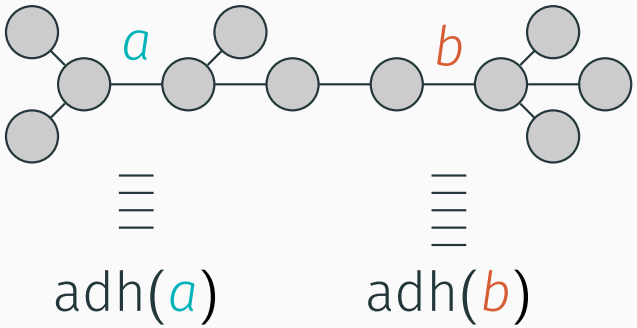
Every graph has a lean optimal tree-cut decomposition



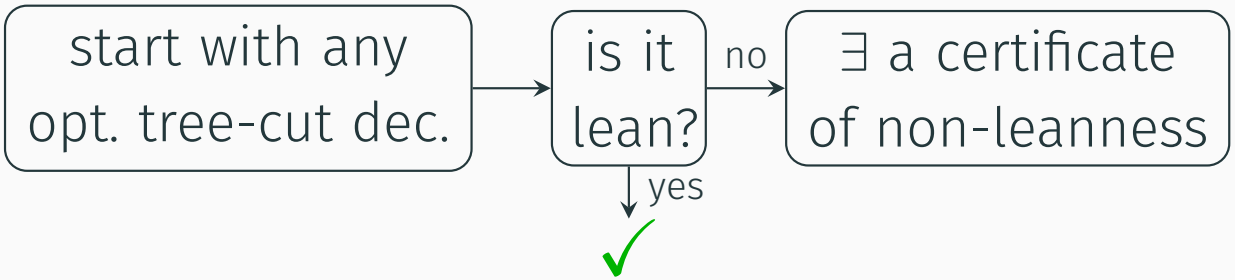


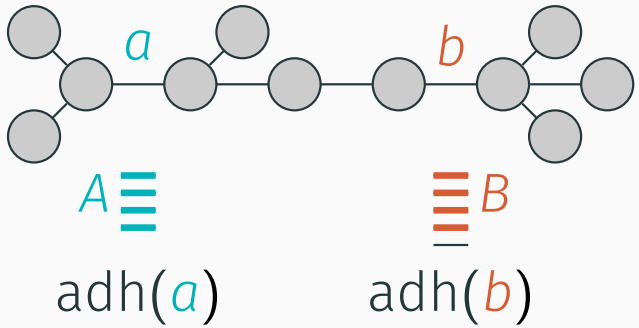
Every graph has a lean optimal tree-cut decomposition



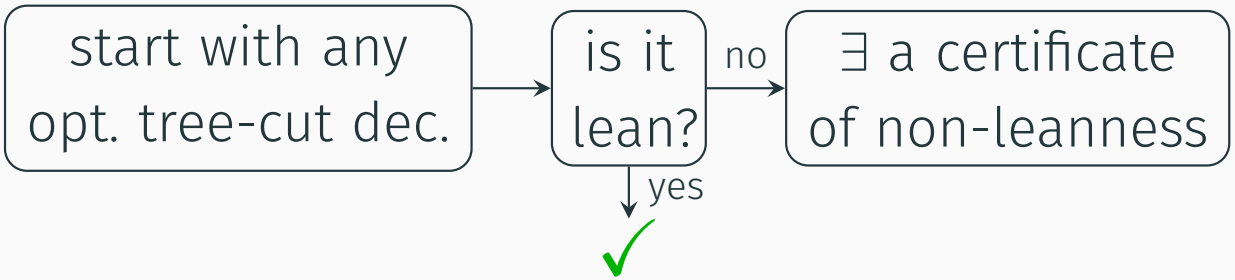


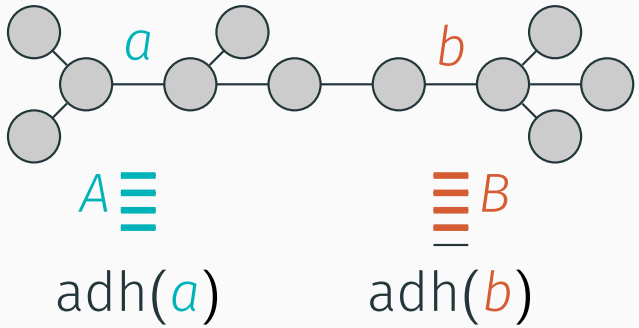
Every graph has a lean optimal tree-cut decomposition





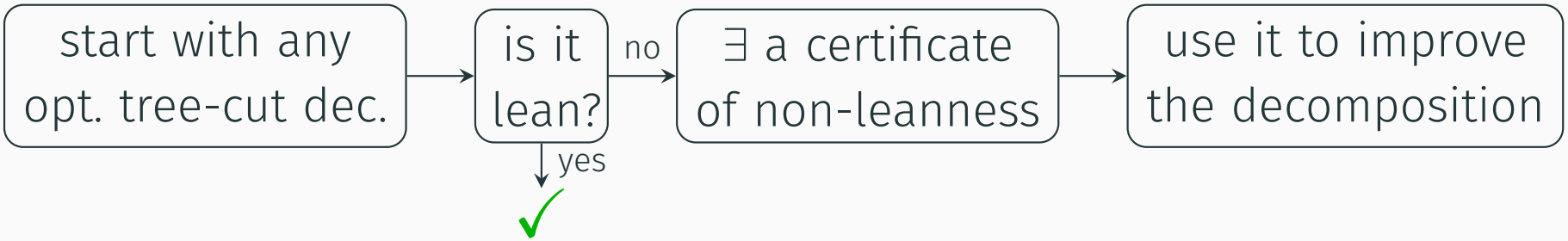
Every graph has a lean optimal tree-cut decomposition

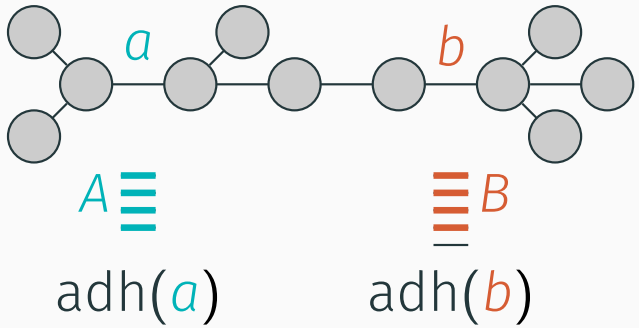




- no $|A|$ edge-disjoint A-B paths
- no edge of adhesion $< |A|$ between a and b

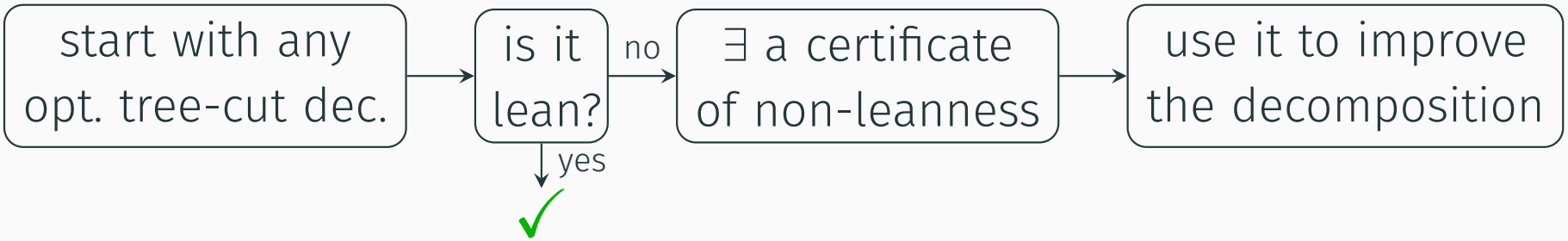
Every graph has a lean optimal tree-cut decomposition

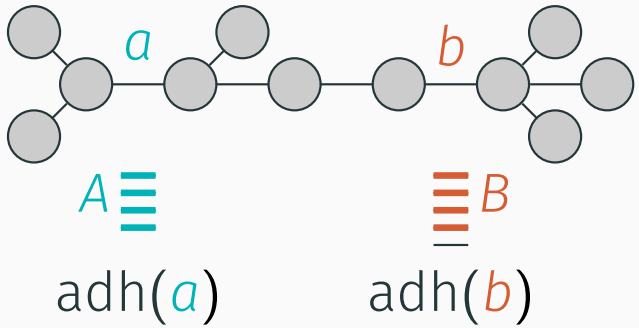




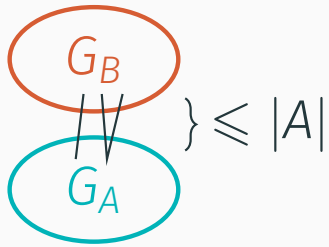
- no $|A|$ edge-disjoint A-B paths
- no edge of adhesion $< |A|$ between a and b

Every graph has a lean optimal tree-cut decomposition

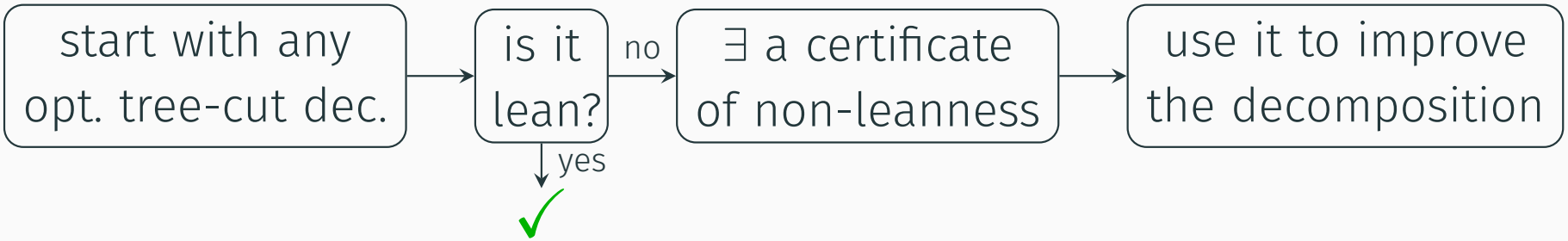


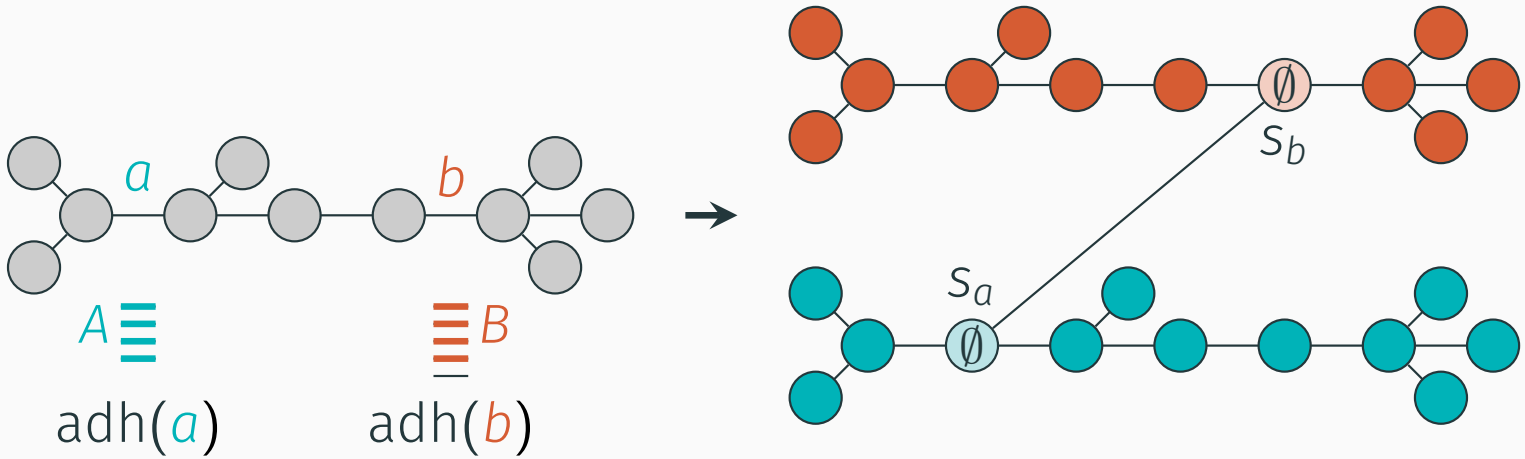


- no $|A|$ edge-disjoint A-B paths
- no edge of adhesion $< |A|$ between a and b

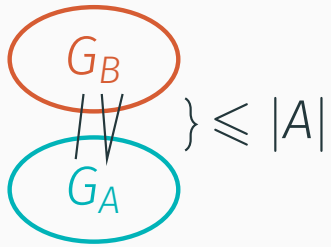


Every graph has a lean optimal tree-cut decomposition

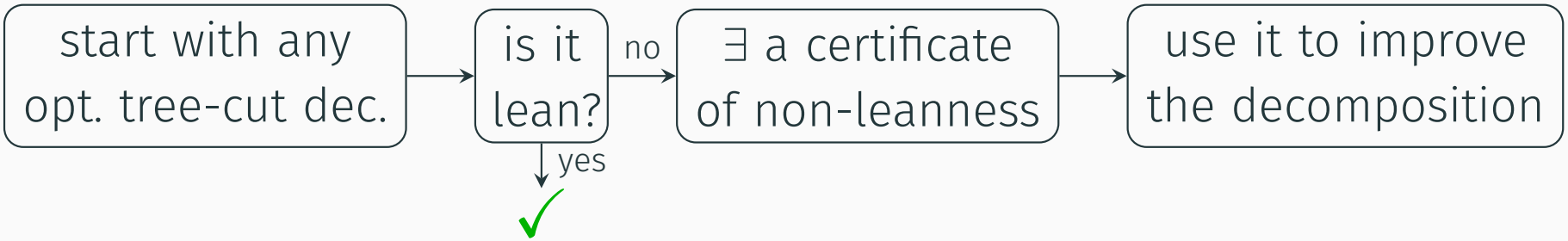


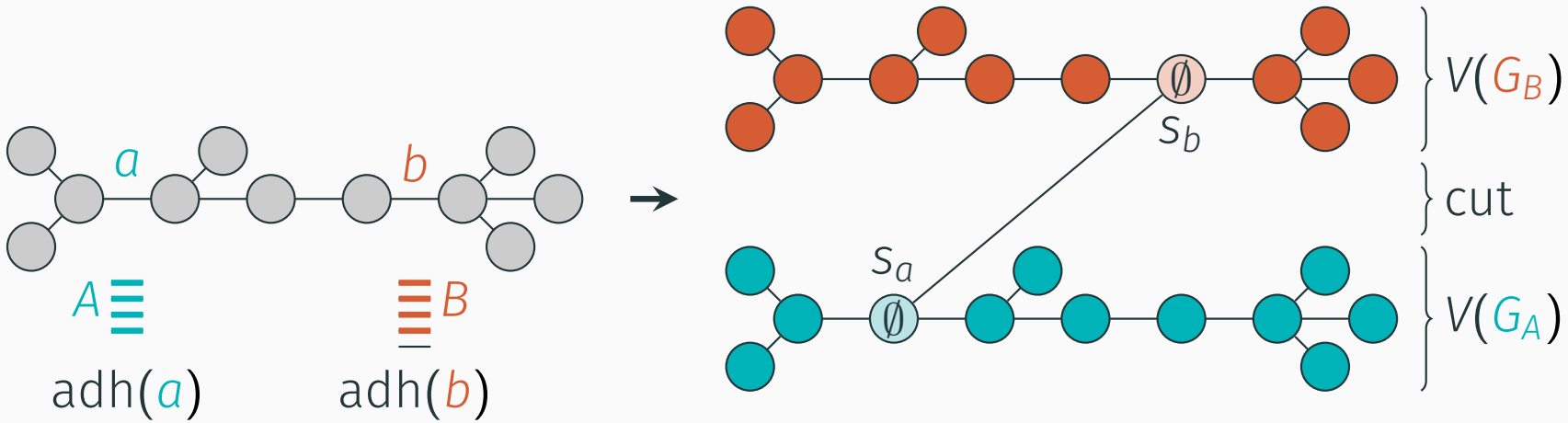


- no $|A|$ edge-disjoint A-B paths
- no edge of adhesion $< |A|$ between a and b

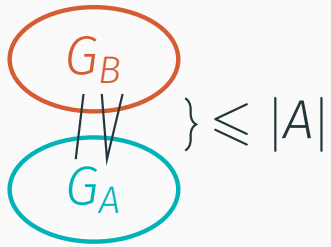


Every graph has a lean optimal tree-cut decomposition

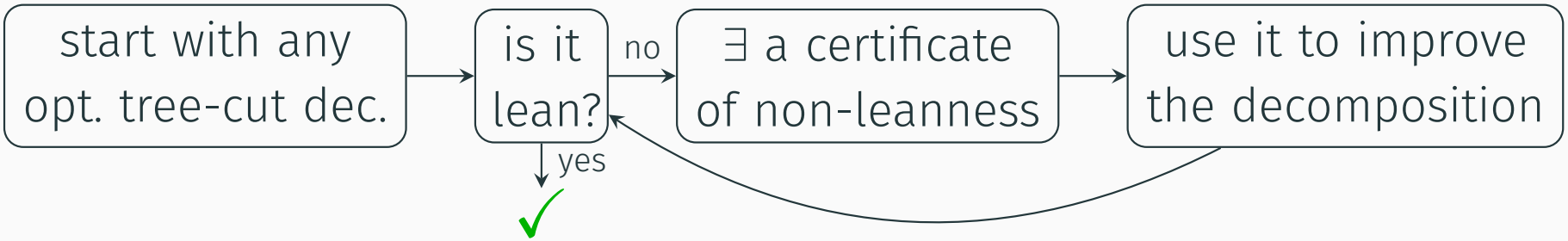


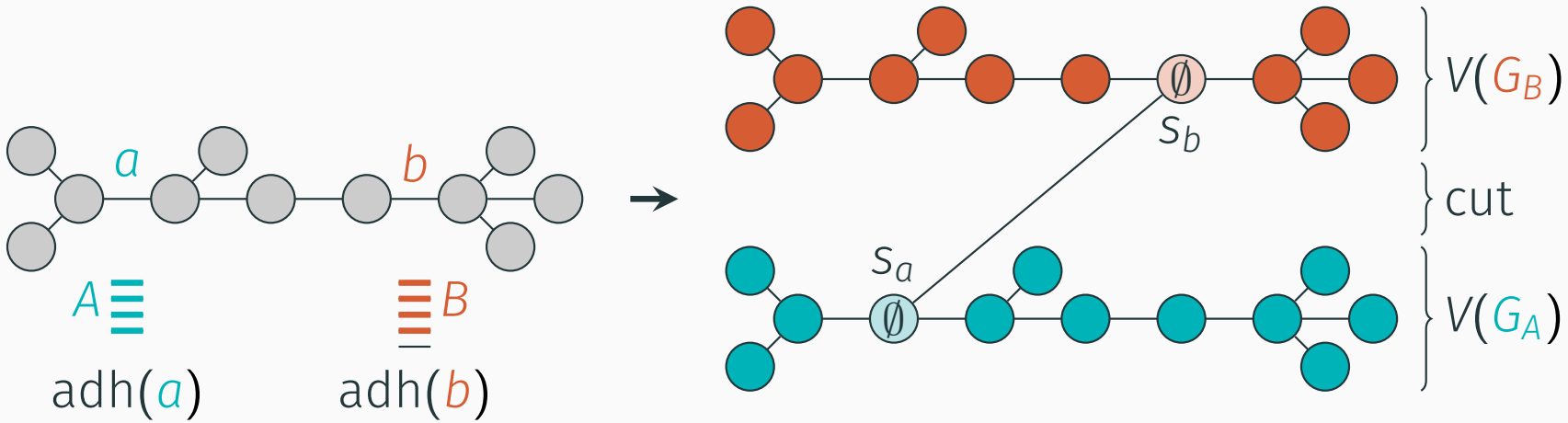


- no $|A|$ edge-disjoint A-B paths
- no edge of adhesion $< |A|$ between a and b

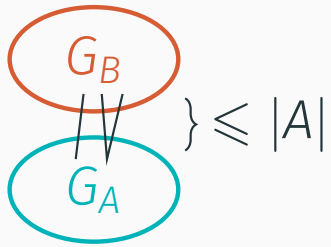


Every graph has a lean optimal tree-cut decomposition

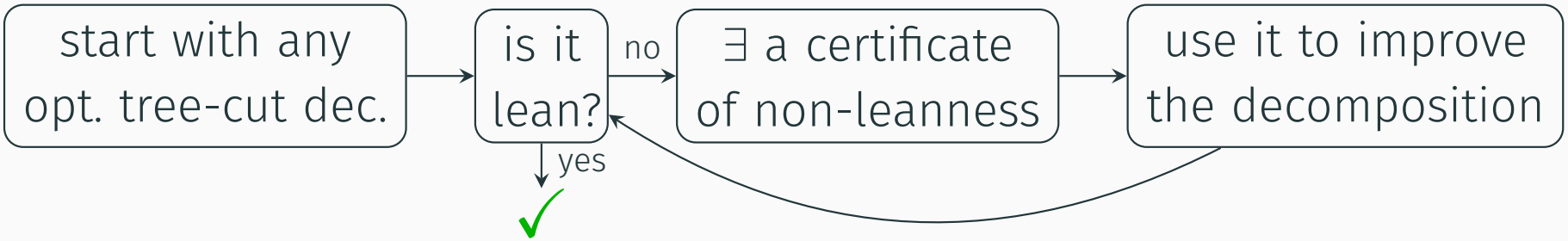




- no $|A|$ edge-disjoint A-B paths
- no edge of adhesion $< |A|$ between a and b



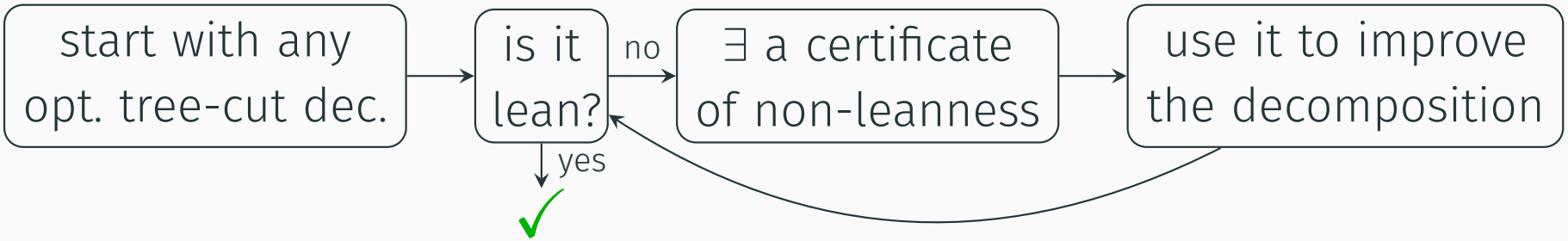
Every graph has a lean optimal tree-cut decomposition



Two statements to prove:

- the tree-cut width does not increase
- the process terminates

Every graph has a lean optimal tree-cut decomposition



Two statements to prove:

- the tree-cut width does not increase
- the process terminates

Not included in this talk!

Application 1: obstructions bounds

obstruction to $\mathcal{F}$: minimal structure present in non-members of $\mathcal{F}$

Application 1: obstructions bounds

obstruction to $\mathcal{F}$: minimal structure present in non-members of $\mathcal{F}$

minor-obstructions to planarity: K_5 or $K_{3,3}$ (minor-minimal non-planar)

Application 1: obstructions bounds

obstruction to $\mathcal{F}$: minimal structure present in non-members of $\mathcal{F}$

$\mathcal{TCW}_{\leqslant k}$: graphs with **tcw** $\leqslant k$

Application 1: obstructions bounds

obstruction to $\mathcal{F}$: minimal structure present in non-members of $\mathcal{F}$

$\mathcal{TCW}_{\leqslant k}$: graphs with **tcw** $\leqslant k$

Theorem (Giannopoulou, Kwon, R., Thilikos, 2019)

G immersion-obstruction to $\mathcal{TCW}_{\leqslant k} \Rightarrow |G| = 2^{2^{O(k^4)}}$

Application 1: obstructions bounds

obstruction to $\mathcal{F}$: minimal structure present in non-members of $\mathcal{F}$

$\mathcal{TCW}_{\leqslant k}$: graphs with **tcw** $\leqslant k$

Theorem (Giannopoulou, Kwon, R., Thilikos, 2019)

G immersion-obstruction to $\mathcal{TCW}_{\leqslant k} \Rightarrow |G| = 2^{2^{O(k^4)}}$

- G immersion-obstruction of $\mathcal{CTW}_{\leqslant k} \Rightarrow |G| = 2^{O(k^3 \log k)}$ (Giannopoulou et al. 2018)

Application 1: obstructions bounds

obstruction to $\mathcal{F}$: minimal structure present in non-members of $\mathcal{F}$

$\mathcal{TCW}_{\leqslant k}$: graphs with **tcw** $\leqslant k$

Theorem (Giannopoulou, Kwon, R., Thilikos, 2019)

G immersion-obstruction to $\mathcal{TCW}_{\leqslant k} \Rightarrow |G| = 2^{2^{O(k^4)}}$

- G immersion-obstruction of $\mathcal{CTW}_{\leqslant k} \Rightarrow |G| = 2^{O(k^3 \log k)}$ (Giannopoulou et al. 2018)
- G minor-obstruction of $\mathcal{TW}_{\leqslant k} \Rightarrow |G| = 2^{2^{O(k^5)}}$ (Lagergren 1998)
- G minor-obstruction of $\mathcal{PW}_{\leqslant k} \Rightarrow |G| = 2^{O(k^4)}$ (idem)

Application 2: computing tree-cut width

Problem: given (G, k), decide if $\mathbf{tcw}(G) \leqslant k$

Application 2: computing tree-cut width

Problem: given (G, k), decide if $\mathbf{tcw}(G) \leqslant k$

- Robertson and Seymour 2010 + Grohe et al. 2011: FPT algorithm (existential / non-uniform)

Application 2: computing tree-cut width

Problem: given (G, k), decide if $\mathbf{tcw}(G) \leqslant k$

- Robertson and Seymour 2010 + Grohe et al. 2011: FPT algorithm (existential / non-uniform)
- Kim et al. 2018: 2-approximation in $O\left(2^{k^2 \log^k} n^2\right)$ time

Application 2: computing tree-cut width

Problem: given (G, k), decide if $\mathbf{tcw}(G) \leqslant k$

- Robertson and Seymour 2010 + Grohe et al. 2011: FPT algorithm (existential / non-uniform)
- Kim et al. 2018: 2-approximation in $O\left(2^{k^2 \log^k} n^2\right)$ time

Theorem (Giannopoulou, Kwon, R., Thilikos, 2019)

One can construct an algorithm that runs in time $f(k) \cdot n$.

Further research

Further research

- simplify the proof of leanness
- get tight bounds on the size of obstructions
- reduce the contribution of k in the algorithm complexity

Further research

- simplify the proof of leanness
- get tight bounds on the size of obstructions
- reduce the contribution of k in the algorithm complexity

Thank you for your attention!