Graph modification problems with forbidden minors

Ignasi Sau

LIRMM, Université de Montpellier, CNRS

UFC, Fortaleza

November 4, 2022

Outline of the talk

Next section is...

Graph modification problems

Let $\mathcal{C}$ be a target graph class (planar graphs, bounded degree, ...).

Let $\mathcal{M}$ be a set of allowed graph modification operations (vertex deletion, edge deletion/addition/contraction, ...).

Graph modification problems

Let $\mathcal{C}$ be a target graph class (planar graphs, bounded degree, ...).

Let $\mathcal{M}$ be a set of allowed graph modification operations
(vertex deletion, edge deletion/addition/contraction, ...).

$\mathcal{M}$-Modification to $\mathcal{C}$	
Input:	A graph G and an integer k.
Question:	Can we transform G to a graph in $\mathcal{C}$ by applying at most k operations from $\mathcal{M}$?

This meta-problem has a huge expressive power.

Many possible interesting variants

- $\mathcal{M}$ = vertex deletion, $\mathcal{C}$ = generalization of bipartite graphs.
  [Baste, Faria, Klein, S. 2015: arXiv 1504.05515]

Many possible interesting variants

- $\mathcal{M}$ = vertex deletion, $\mathcal{C}$ = generalization of bipartite graphs.

  [Baste, Faria, Klein, S. 2015: arXiv 1504.05515]

- $\mathcal{M}$ = vertex deletion, $\mathcal{C}$ = forbidden induced subgraphs.

  [S., Souza. 2020: arXiv 2004.08324]

Many possible interesting variants

- $\mathcal{M}$ = vertex deletion, $\mathcal{C}$ = generalization of bipartite graphs.
  [Baste, Faria, Klein, S. 2015: arXiv 1504.05515]
- $\mathcal{M}$ = vertex deletion, $\mathcal{C}$ = forbidden induced subgraphs.
  [S., Souza. 2020: arXiv 2004.08324]
- $\mathcal{M}$ = edge contraction, $\mathcal{C}$ = graph transversal parameters.
  [Lima, dos Santos, S., Souza. 2020: arXiv 2005.01460]
  [Lima, dos Santos, S., Souza, Tale. 2022: arXiv 2202.03322]

This talk: forbidden minors

$\mathcal{M}$ = vertex deletion (or more), $\mathcal{C}$ = excluded minors.

This talk: forbidden minors

$\mathcal{M}$ = vertex deletion (or more), $\mathcal{C}$ = excluded minors.

- Linear kernels on sparse graph classes.

[Garnero, Paul, S., Thilikos. 2014: arXiv 1312.6585]

[Garnero, Paul, S., Thilikos. 2016: arXiv 1610.06131]

This talk: forbidden minors

$\mathcal{M}$ = vertex deletion (or more), $\mathcal{C}$ = excluded minors.

- Linear kernels on sparse graph classes.

  [Garnero, Paul, S., Thilikos. 2014: arXiv 1312.6585]

  [Garnero, Paul, S., Thilikos. 2016: arXiv 1610.06131]

- FPT algorithms parameterized by treewidth.

  [Baste, S., Thilikos. 2017: arXiv 1704.07284]

  [Baste, S., Thilikos. 2018: arXiv 2103.06536]

  [Baste, S., Thilikos. 2019: arXiv 2103.06614]

  [Baste, S., Thilikos. 2019: arXiv 1907.04442]

This talk: forbidden minors

$\mathcal{M}$ = vertex deletion (or more), $\mathcal{C}$ = excluded minors.

- Linear kernels on sparse graph classes.

  [Garnero, Paul, S., Thilikos. 2014: arXiv 1312.6585]

  [Garnero, Paul, S., Thilikos. 2016: arXiv 1610.06131]

- FPT algorithms parameterized by treewidth.

  [Baste, S., Thilikos. 2017: arXiv 1704.07284]

  [Baste, S., Thilikos. 2018: arXiv 2103.06536]

  [Baste, S., Thilikos. 2019: arXiv 2103.06614]

  [Baste, S., Thilikos. 2019: arXiv 1907.04442]

- FPT algorithms parameterized by the solution size (# modifications).

  [Kim, Langer, Paul, Reidl, Rossmanith, S., Sikdar. 2012: arXiv 1207.0835]

  [S., Stamoulis, Thilikos. 2020: arXiv 2004.12692]

  [S., Stamoulis, Thilikos. 2021: arXiv 2103.00882]

This talk: forbidden minors

$\mathcal{M}$ = vertex deletion (or more), $\mathcal{C}$ = excluded minors.

- Linear kernels on sparse graph classes.

  [Garnero, Paul, S., Thilikos. 2014: arXiv 1312.6585]

  [Garnero, Paul, S., Thilikos. 2016: arXiv 1610.06131]

★ FPT algorithms parameterized by treewidth.

[Baste, S., Thilikos. 2017: arXiv 1704.07284]

[Baste, S., Thilikos. 2018: arXiv 2103.06536]

[Baste, S., Thilikos. 2019: arXiv 2103.06614]

[Baste, S., Thilikos. 2019: arXiv 1907.04442]

★ FPT algorithms parameterized by the solution size (# modifications).

[Kim, Langer, Paul, Reidl, Rossmanith, S., Sikdar. 2012: arXiv 1207.0835]

[S., Stamoulis, Thilikos. 2020: arXiv 2004.12692]

[S., Stamoulis, Thilikos. 2021: arXiv 2103.00882]

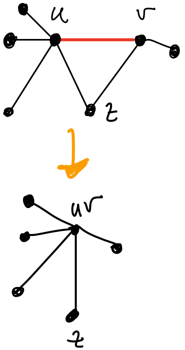
★ More general modification operations.

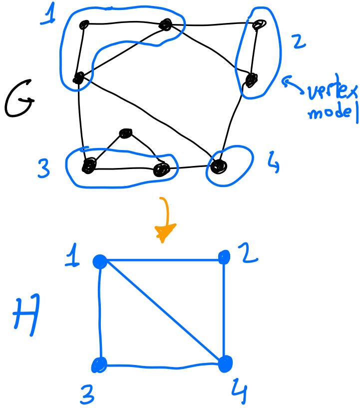
[Fomin, Golovach, S., Stamoulis, Thilikos. 2021: arXiv 2111.02755]

[Morelle, S., Stamoulis, Thilikos. 2022: arXiv 2210.02167]

Graph minors

A graph H is a minor of a graph G, denoted by $H \leqslant_m G$, if H can be obtained from a subgraph of G by contracting edges.





Minor-closed graph classes

A graph class $\mathcal{C}$ is minor-closed (or closed under minors) if

$$G \in \mathcal{C} \;\Rightarrow\; H \in \mathcal{C} \text{ for every } H \leqslant_m G.$$

Minor-closed graph classes

A graph class $\mathcal{C}$ is minor-closed (or closed under minors) if

$$G \in \mathcal{C} \;\Rightarrow\; H \in \mathcal{C} \text{ for every } H \leqslant_m G.$$

Let $\mathcal{F}$ be a (possibly infinite) family of graphs. We define $\text{exc}(\mathcal{F})$ as the class of all graphs that do not contain any of the graphs in $\mathcal{F}$ as a minor.

Minor-closed graph classes

A graph class $\mathcal{C}$ is minor-closed (or closed under minors) if

$$G \in \mathcal{C} \;\Rightarrow\; H \in \mathcal{C} \text{ for every } H \leqslant_m G.$$

Let $\mathcal{F}$ be a (possibly infinite) family of graphs. We define $\text{exc}(\mathcal{F})$ as the class of all graphs that do not contain any of the graphs in $\mathcal{F}$ as a minor.

Every minor-closed graph class $\mathcal{C}$ can be characterized by excluded minors:

> List all the graphs $\mathcal{F}_{\mathcal{C}} := \{G_1, G_2, \ldots\}$ that do not belong to $\mathcal{C}$, and then $\mathcal{C} = \text{exc}(\mathcal{F}_{\mathcal{C}})$.

Minor-closed graph classes

A graph class $\mathcal{C}$ is minor-closed (or closed under minors) if

$$G \in \mathcal{C} \;\Rightarrow\; H \in \mathcal{C} \text{ for every } H \leqslant_m G.$$

Let $\mathcal{F}$ be a (possibly infinite) family of graphs. We define $\text{exc}(\mathcal{F})$ as the class of all graphs that do not contain any of the graphs in $\mathcal{F}$ as a minor.

Every minor-closed graph class $\mathcal{C}$ can be characterized by excluded minors:

> List all the graphs $\mathcal{F}_{\mathcal{C}} := \{G_1, G_2, \ldots\}$ that do not belong to $\mathcal{C}$, and then $\mathcal{C} = \text{exc}(\mathcal{F}_{\mathcal{C}})$.

Note that, in general, this list $\mathcal{F}_{\mathcal{C}} = \{G_1, G_2, \ldots\}$ may be infinite.

Forbidden minors for some minor-closed graph classes

- If $\mathcal{C}$ = independent sets, then $\mathcal{C} = \text{exc}(K_2)$.

Forbidden minors for some minor-closed graph classes

- If $\mathcal{C}$ = independent sets, then $\mathcal{C} = \text{exc}(K_2)$.
- If $\mathcal{C}$ = forests, then $\mathcal{C} = \text{exc}(K_3)$.

Forbidden minors for some minor-closed graph classes

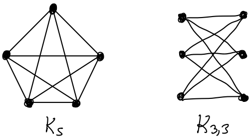
- If $\mathcal{C}$ = independent sets, then $\mathcal{C} = \text{exc}(K_2)$.
- If $\mathcal{C}$ = forests, then $\mathcal{C} = \text{exc}(K_3)$.
- If $\mathcal{C}$ = series-parallel graphs, then $\mathcal{C} = \text{exc}(K_4)$.

Forbidden minors for some minor-closed graph classes

- If $\mathcal{C}$ = independent sets, then $\mathcal{C} = \text{exc}(K_2)$.
- If $\mathcal{C}$ = forests, then $\mathcal{C} = \text{exc}(K_3)$.
- If $\mathcal{C}$ = series-parallel graphs, then $\mathcal{C} = \text{exc}(K_4)$.
- If $\mathcal{C}$ = outerplanar graphs, then $\mathcal{C} = \text{exc}(K_4, K_{2,3})$.

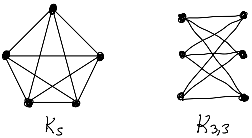
Forbidden minors for some minor-closed graph classes

- If $\mathcal{C}$ = independent sets, then $\mathcal{C} = \text{exc}(K_2)$.
- If $\mathcal{C}$ = forests, then $\mathcal{C} = \text{exc}(K_3)$.
- If $\mathcal{C}$ = series-parallel graphs, then $\mathcal{C} = \text{exc}(K_4)$.
- If $\mathcal{C}$ = outerplanar graphs, then $\mathcal{C} = \text{exc}(K_4, K_{2,3})$.
- If $\mathcal{C}$ = planar graphs, then $\mathcal{C} = \text{exc}(K_5, K_{3,3})$. [Kuratowski. 1930]



Forbidden minors for some minor-closed graph classes

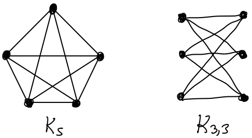
- If $\mathcal{C}$ = independent sets, then $\mathcal{C} = \text{exc}(K_2)$.
- If $\mathcal{C}$ = forests, then $\mathcal{C} = \text{exc}(K_3)$.
- If $\mathcal{C}$ = series-parallel graphs, then $\mathcal{C} = \text{exc}(K_4)$.
- If $\mathcal{C}$ = outerplanar graphs, then $\mathcal{C} = \text{exc}(K_4, K_{2,3})$.
- If $\mathcal{C}$ = planar graphs, then $\mathcal{C} = \text{exc}(K_5, K_{3,3})$. [Kuratowski. 1930]



- If $\mathcal{C}$ = graphs embeddable in the projective plane, then $|\mathcal{F}_{\mathcal{C}}| = 35$.

Forbidden minors for some minor-closed graph classes

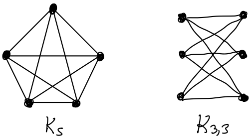
- If $\mathcal{C}$ = independent sets, then $\mathcal{C} = \text{exc}(K_2)$.
- If $\mathcal{C}$ = forests, then $\mathcal{C} = \text{exc}(K_3)$.
- If $\mathcal{C}$ = series-parallel graphs, then $\mathcal{C} = \text{exc}(K_4)$.
- If $\mathcal{C}$ = outerplanar graphs, then $\mathcal{C} = \text{exc}(K_4, K_{2,3})$.
- If $\mathcal{C}$ = planar graphs, then $\mathcal{C} = \text{exc}(K_5, K_{3,3})$. [Kuratowski. 1930]



- If $\mathcal{C}$ = graphs embeddable in the projective plane, then $|\mathcal{F}_{\mathcal{C}}| = 35$.
- If $\mathcal{C}$ = graphs embeddable in a fixed non-orientable surface, then $\mathcal{F}_{\mathcal{C}}$ is finite. [Archdeacon, Huneke. 1989]

Forbidden minors for some minor-closed graph classes

- If $\mathcal{C}$ = independent sets, then $\mathcal{C} = \text{exc}(K_2)$.
- If $\mathcal{C}$ = forests, then $\mathcal{C} = \text{exc}(K_3)$.
- If $\mathcal{C}$ = series-parallel graphs, then $\mathcal{C} = \text{exc}(K_4)$.
- If $\mathcal{C}$ = outerplanar graphs, then $\mathcal{C} = \text{exc}(K_4, K_{2,3})$.
- If $\mathcal{C}$ = planar graphs, then $\mathcal{C} = \text{exc}(K_5, K_{3,3})$. [Kuratowski. 1930]



- If $\mathcal{C}$ = graphs embeddable in the projective plane, then $|\mathcal{F}_{\mathcal{C}}| = 35$.
- If $\mathcal{C}$ = graphs embeddable in a fixed non-orientable surface, then $\mathcal{F}_{\mathcal{C}}$ is finite. [Archdeacon, Huneke. 1989]
- If $\mathcal{C}$ = graphs embeddable in a fixed orientable surface, then $\mathcal{F}_{\mathcal{C}}$ is finite. [Robertson, Seymour. 1990]

Wagner's conjecture

Conjecture (Wagner. 1970)

For every minor-closed graph class $\mathcal{C}$, there exists a finite set of graphs $\mathcal{F}_{\mathcal{C}}$ such that $\mathcal{C} = \text{exc}(\mathcal{F}_{\mathcal{C}})$.

Wagner's conjecture... now Robertson-Seymour's theorem

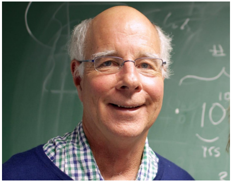
Theorem (Robertson, Seymour. 1983-2004)

For every minor-closed graph class $\mathcal{C}$, there exists a finite set of graphs $\mathcal{F}_{\mathcal{C}}$ such that $\mathcal{C} = \text{exc}(\mathcal{F}_{\mathcal{C}})$.

Parameterized complexity in a nutshell

Idea Measure the complexity of an algorithm in terms of the input size and an additional parameter.

This theory started in the late 80's, by Downey and Fellows:



Today, it is a well-established and very active area.

Parameterized problems

A parameterized problem is a language $L \subseteq \Sigma^* \times \mathbb{N}$,
where Σ is a fixed, finite alphabet.

For an instance $(x, k) \in \Sigma^* \times \mathbb{N}$, k is called the parameter.

Parameterized problems

A parameterized problem is a language $L \subseteq \Sigma^* \times \mathbb{N}$,
where Σ is a fixed, finite alphabet.

For an instance $(x, k) \in \Sigma^* \times \mathbb{N}$, k is called the parameter.

- k-Vertex Cover: Does a graph G contain a set $S \subseteq V(G)$, with $|S| \leq k$, containing at least an endpoint of every edge?
- k-Clique: Does a graph G contain a set $S \subseteq V(G)$, with $|S| \geq k$, of pairwise adjacent vertices?
- Vertex k-Coloring: Can the vertices of a graph be colored with $\leq k$ colors, so that any two adjacent vertices get different colors?

Parameterized problems

A parameterized problem is a language $L \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a fixed, finite alphabet.

For an instance $(x, k) \in \Sigma^* \times \mathbb{N}$, k is called the parameter.

- k-Vertex Cover: Does a graph G contain a set $S \subseteq V(G)$, with $|S| \leq k$, containing at least an endpoint of every edge?
- k-Clique: Does a graph G contain a set $S \subseteq V(G)$, with $|S| \geq k$, of pairwise adjacent vertices?
- Vertex k-Coloring: Can the vertices of a graph be colored with $\leq k$ colors, so that any two adjacent vertices get different colors?

These three problems are NP-hard, but are they equally hard?

They behave quite differently...

- k-Vertex Cover: Solvable in time $\mathcal{O}(2^k \cdot (m+n))$

- k-Clique: Solvable in time $\mathcal{O}(k^2 \cdot n^k)$

- Vertex k-Coloring: NP-hard for fixed $k = 3$.

They behave quite differently...

- k-VERTEX COVER: Solvable in time $\mathcal{O}(2^k \cdot (m+n)) = f(k) \cdot n^{\mathcal{O}(1)}$.

- k-CLIQUE: Solvable in time $\mathcal{O}(k^2 \cdot n^k) = f(k) \cdot n^{g(k)}$.

- VERTEX k-COLORING: NP-hard for fixed $k = 3$.

They behave quite differently...

- k-Vertex Cover: Solvable in time $\mathcal{O}(2^k \cdot (m+n)) = f(k) \cdot n^{\mathcal{O}(1)}$.

  The problem is FPT (fixed-parameter tractable)

- k-Clique: Solvable in time $\mathcal{O}(k^2 \cdot n^k) = f(k) \cdot n^{g(k)}$.

- Vertex k-Coloring: NP-hard for fixed $k = 3$.

They behave quite differently...

- k-Vertex Cover: Solvable in time $\mathcal{O}(2^k \cdot (m+n)) = f(k) \cdot n^{\mathcal{O}(1)}$.

  The problem is FPT (fixed-parameter tractable)

- k-Clique: Solvable in time $\mathcal{O}(k^2 \cdot n^k) = f(k) \cdot n^{g(k)}$.

  The problem is XP (slice-wise polynomial)

- Vertex k-Coloring: NP-hard for fixed $k = 3$.

They behave quite differently...

- k-VERTEX COVER: Solvable in time $\mathcal{O}(2^k \cdot (m+n)) = f(k) \cdot n^{\mathcal{O}(1)}$.

  The problem is FPT (fixed-parameter tractable)

- k-CLIQUE: Solvable in time $\mathcal{O}(k^2 \cdot n^k) = f(k) \cdot n^{g(k)}$.

  The problem is XP (slice-wise polynomial)

- VERTEX k-COLORING: NP-hard for fixed $k = 3$.

  The problem is para-NP-hard

Next section is...

Hitting forbidden minors

- If $\mathcal{C}$ = {edgeless graphs}, then $\mathcal{F} = \{K_2\}$.
- If $\mathcal{C}$ = {forests}, then $\mathcal{F} = \{K_3\}$.
- If $\mathcal{C}$ = {outerplanar graphs}, then $\mathcal{F} = \{K_4, K_{2,3}\}$.
- If $\mathcal{C}$ = {planar graphs}, then $\mathcal{F} = \{K_5, K_{3,3}\}$.

Hitting forbidden minors

- If $\mathcal{C}$ = {edgeless graphs}, then $\mathcal{F} = \{K_2\}$.
- If $\mathcal{C}$ = {forests}, then $\mathcal{F} = \{K_3\}$.
- If $\mathcal{C}$ = {outerplanar graphs}, then $\mathcal{F} = \{K_4, K_{2,3}\}$.
- If $\mathcal{C}$ = {planar graphs}, then $\mathcal{F} = \{K_5, K_{3,3}\}$.

Let $\mathcal{F}$ be a fixed finite collection of graphs.

Hitting forbidden minors

- If $\mathcal{C}$ = {edgeless graphs}, then $\mathcal{F} = \{K_2\}$.
- If $\mathcal{C}$ = {forests}, then $\mathcal{F} = \{K_3\}$.
- If $\mathcal{C}$ = {outerplanar graphs}, then $\mathcal{F} = \{K_4, K_{2,3}\}$.
- If $\mathcal{C}$ = {planar graphs}, then $\mathcal{F} = \{K_5, K_{3,3}\}$.

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-DELETION
Input: A graph G and an integer k.
Question: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

Hitting forbidden minors

- If $\mathcal{C}$ = {edgeless graphs}, then $\mathcal{F} = \{K_2\}$.
- If $\mathcal{C}$ = {forests}, then $\mathcal{F} = \{K_3\}$.
- If $\mathcal{C}$ = {outerplanar graphs}, then $\mathcal{F} = \{K_4, K_{2,3}\}$.
- If $\mathcal{C}$ = {planar graphs}, then $\mathcal{F} = \{K_5, K_{3,3}\}$.

Let $\mathcal{F}$ be a fixed finite collection of graphs.

> $\mathcal{F}$-M-Deletion
> **Input**: A graph G and an integer k.
> **Question**: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

- $\mathcal{F} = \{K_2\}$: Vertex Cover.
- $\mathcal{F} = \{K_3\}$: Feedback Vertex Set.
- $\mathcal{F} = \{K_5, K_{3,3}\}$: Vertex Planarization.
- $\mathcal{F}$ = {diamond}: Cactus Vertex Deletion.

Hitting forbidden minors

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-Deletion	
Input:	A graph G and an integer k.
Question:	Does G contain a set $S \subseteq V(G)$ with $\lvert S \rvert \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

Hitting forbidden minors

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-Deletion	
Input:	A graph G and an integer k.
Question:	Does G contain a set $S \subseteq V(G)$ with $\|S\| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

NP-hard if $\mathcal{F}$ contains a graph with some edge. [Lewis, Yannakakis. 1980]

Hitting forbidden minors

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-Deletion	
Input:	A graph G and an integer k.
Question:	Does G contain a set $S \subseteq V(G)$ with $\|S\| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

NP-hard if $\mathcal{F}$ contains a graph with some edge. [Lewis, Yannakakis. 1980]

We consider the following two parameterizations of $\mathcal{F}$-M-Deletion:

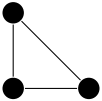
1. Structural parameter: $\mathrm{tw}(G)$.
2. Solution size: k.

Joint work with Julien Baste, Laure Morelle, Giannos Stamoulis, and Dimitrios M. Thilikos.

Next subsection is...

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

Example of a 2-tree:

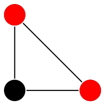


[Figure by Julien Baste]

Treewidth via k-trees

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

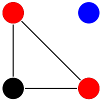
Example of a 2-tree:



[Figure by Julien Baste]

Treewidth via k-trees

Example of a 2-tree:



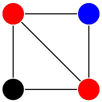
[Figure by Julien Baste]

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

Treewidth via k-trees

Example of a 2-tree:

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

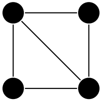


[Figure by Julien Baste]

Treewidth via k-trees

Example of a 2-tree:

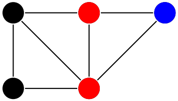
For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.



[Figure by Julien Baste]

Treewidth via k-trees

Example of a 2-tree:

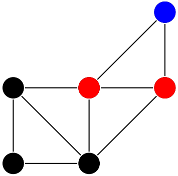


[Figure by Julien Baste]

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

Treewidth via k-trees

Example of a 2-tree:

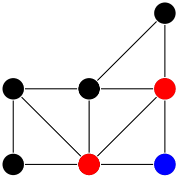


[Figure by Julien Baste]

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

Treewidth via k-trees

Example of a 2-tree:

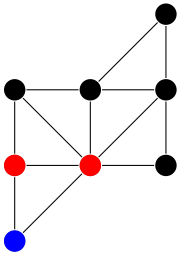


[Figure by Julien Baste]

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

Treewidth via k-trees

Example of a 2-tree:

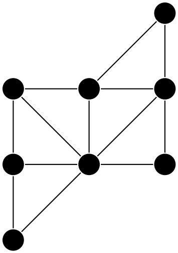


[Figure by Julien Baste]

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

Treewidth via k-trees

Example of a 2-tree:

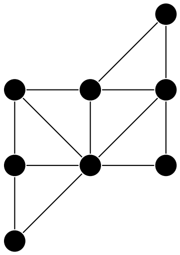


[Figure by Julien Baste]

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

Treewidth via k-trees

Example of a 2-tree:



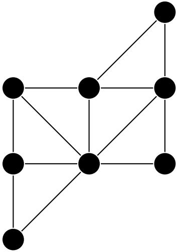
[Figure by Julien Baste]

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

A partial k-tree is a subgraph of a k-tree.

Treewidth via k-trees

Example of a 2-tree:



[Figure by Julien Baste]

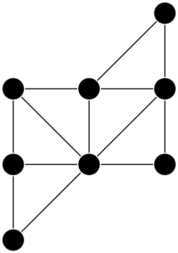
For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

A partial k-tree is a subgraph of a k-tree.

Treewidth of a graph G, denoted $\text{tw}(G)$:
smallest integer k such that G is a partial k-tree.

Treewidth via k-trees

Example of a 2-tree:



[Figure by Julien Baste]

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

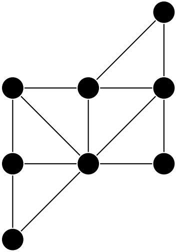
A partial k-tree is a subgraph of a k-tree.

Treewidth of a graph G, denoted $\text{tw}(G)$: smallest integer k such that G is a partial k-tree.

Invariant that measures the topological resemblance of a graph to a forest.

Treewidth via k-trees

Example of a 2-tree:



[Figure by Julien Baste]

For $k \geq 1$, a k-tree is a graph that can be built starting from a $(k+1)$-clique and then iteratively adding a vertex connected to a k-clique.

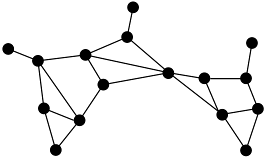
A partial k-tree is a subgraph of a k-tree.

Treewidth of a graph G, denoted $\mathrm{tw}(G)$: smallest integer k such that G is a partial k-tree.

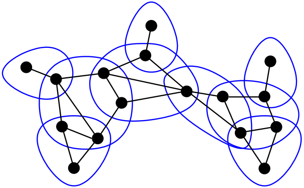
Invariant that measures the topological resemblance of a graph to a forest.

Construction suggests the notion of tree decomposition: small separators.

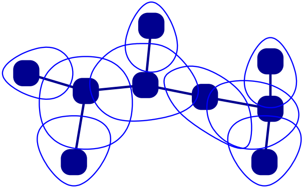
Treewidth measures the tree-likeness of a graph



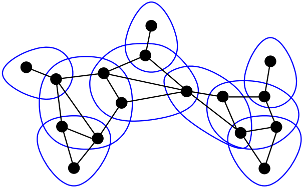
Treewidth measures the tree-likeness of a graph



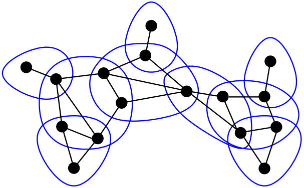
Treewidth measures the tree-likeness of a graph

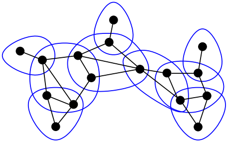


Treewidth measures the tree-likeness of a graph

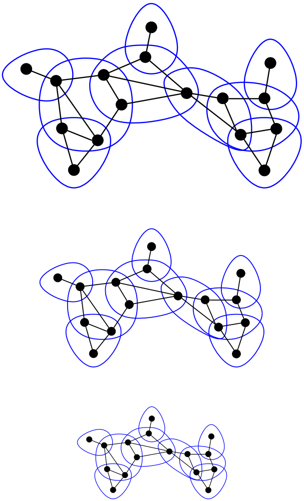


Treewidth measures the tree-likeness of a graph

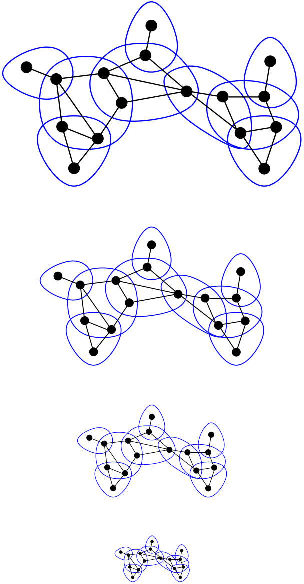




Treewidth measures the tree-likeness of a graph



Treewidth measures the tree-likeness of a graph



Theorem (Courcelle. 1990)

Every problem expressible in MSOL can be solved in time $f_{\mathcal{F}}(\mathrm{tw}) \cdot n$ *on graphs on* n *vertices and treewidth at most* tw.

Theorem (Courcelle. 1990)

Every problem expressible in MSOL can be solved in time $f_{\mathcal{F}}(\text{tw}) \cdot n$ on graphs on n vertices and treewidth at most tw.

It is not difficult to see that can $\mathcal{F}$-M-Deletion be expressed in MSOL:

$\mathcal{F}$-M-Deletion is FPT parameterized by tw...

Theorem (Courcelle. 1990)

Every problem expressible in MSOL can be solved in time $f_{\mathcal{F}}(\text{tw}) \cdot n$ on graphs on n vertices and treewidth at most tw.

It is not difficult to see that can $\mathcal{F}$-M-DELETION be expressed in MSOL:

$\mathcal{F}$-M-DELETION is FPT parameterized by tw...

$$f_{\mathcal{F}}(\text{tw}) \cdot n$$

Theorem (Courcelle. 1990)

Every problem expressible in MSOL can be solved in time $f_{\mathcal{F}}(\mathrm{tw}) \cdot n$ on graphs on n vertices and treewidth at most tw.

It is not difficult to see that can $\mathcal{F}$-M-DELETION be expressed in MSOL:

$\mathcal{F}$-M-DELETION is FPT parameterized by tw...

$$f_{\mathcal{F}}(\mathrm{tw}) \cdot n = 2^{3^{4^{5^{6^{7^{8^{\mathrm{tw}}}}}}}} \cdot n$$

Theorem (Courcelle. 1990)

Every problem expressible in MSOL can be solved in time $f_{\mathcal{F}}(\mathrm{tw}) \cdot n$ on graphs on n vertices and treewidth at most tw.

It is not difficult to see that can $\mathcal{F}$-M-Deletion be expressed in MSOL:

$\mathcal{F}$-M-Deletion is FPT parameterized by tw...

$$f_{\mathcal{F}}(\mathrm{tw}) \cdot n = 2^{3^{4^{5^{6^{7^{8^{\mathrm{tw}}}}}}}} \cdot n$$

Goal For every $\mathcal{F}$, find the smallest possible function $f_{\mathcal{F}}(\mathrm{tw})$.

Theorem (Courcelle. 1990)

Every problem expressible in MSOL can be solved in time $f_{\mathcal{F}}(\text{tw}) \cdot n$ on graphs on n vertices and treewidth at most tw.

It is not difficult to see that can $\mathcal{F}$-M-Deletion be expressed in MSOL:

$\mathcal{F}$-M-Deletion is FPT parameterized by tw...

$$f_{\mathcal{F}}(\text{tw}) \cdot n = 2^{3^{4^{5^{6^{7^{8^{\text{tw}}}}}}}} \cdot n$$

Goal For every $\mathcal{F}$, find the smallest possible function $f_{\mathcal{F}}(\text{tw})$.

ETH: The 3-SAT problem on n variables cannot be solved in time $2^{o(n)}$.

[Impagliazzo, Paturi. 1999]

What was known for particular collections $\mathcal{F}$

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-DELETION
Input: A graph G and an integer k.
Parameter: The treewidth tw of G.
Question: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

What was known for particular collections $\mathcal{F}$

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-Deletion	
Input:	A graph G and an integer k.
Parameter:	The treewidth tw of G.
Question:	Does G contain a set $S \subseteq V(G)$ with $\lvert S \rvert \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

- $\mathcal{F} = \{K_2\}$: Vertex Cover.

What was known for particular collections $\mathcal{F}$

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-DELETION

Input: A graph G and an integer k.

Parameter: The treewidth tw of G.

Question: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

- $\mathcal{F} = \{K_2\}$: VERTEX COVER.
  Easily solvable in time $2^{\Theta(\text{tw})} \cdot n^{\mathcal{O}(1)}$.

What was known for particular collections $\mathcal{F}$

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-Deletion	
Input:	A graph G and an integer k.
Parameter:	The treewidth tw of G.
Question:	Does G contain a set $S \subseteq V(G)$ with $\lvert S\rvert \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

- $\mathcal{F} = \{K_2\}$: Vertex Cover.
  Easily solvable in time $2^{\Theta(\mathrm{tw})} \cdot n^{\mathcal{O}(1)}$.
- $\mathcal{F} = \{K_3\}$: Feedback Vertex Set.

What was known for particular collections $\mathcal{F}$

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-Deletion
Input: A graph G and an integer k.
Parameter: The treewidth tw of G.
Question: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

- $\mathcal{F} = \{K_2\}$: Vertex Cover.
  Easily solvable in time $2^{\Theta(\text{tw})} \cdot n^{\mathcal{O}(1)}$.

- $\mathcal{F} = \{K_3\}$: Feedback Vertex Set.
  "Hardly" solvable in time $2^{\Theta(\text{tw})} \cdot n^{\mathcal{O}(1)}$.
  [**Cut&Count**: Cygan, Nederlof, Pilipczuk, Pilipczuk, van Rooij, Wojtaszczyk. 2011]

What was known for particular collections $\mathcal{F}$

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-DELETION	
Input:	A graph G and an integer k.
Parameter:	The treewidth tw of G.
Question:	Does G contain a set $S \subseteq V(G)$ with $\|S\| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

- $\mathcal{F} = \{K_2\}$: VERTEX COVER.
  Easily solvable in time $2^{\Theta(\text{tw})} \cdot n^{\mathcal{O}(1)}$.
- $\mathcal{F} = \{K_3\}$: FEEDBACK VERTEX SET.
  "Hardly" solvable in time $2^{\Theta(\text{tw})} \cdot n^{\mathcal{O}(1)}$.
  [**Cut&Count**: Cygan, Nederlof, Pilipczuk, Pilipczuk, van Rooij, Wojtaszczyk. 2011]
- $\mathcal{F} = \{K_5, K_{3,3}\}$: VERTEX PLANARIZATION.

What was known for particular collections $\mathcal{F}$

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-Deletion	
Input:	A graph G and an integer k.
Parameter:	The treewidth tw of G.
Question:	Does G contain a set $S \subseteq V(G)$ with $\lvert S\rvert \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

- $\mathcal{F} = \{K_2\}$: Vertex Cover.
  Easily solvable in time $2^{\Theta(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.

- $\mathcal{F} = \{K_3\}$: Feedback Vertex Set.
  "Hardly" solvable in time $2^{\Theta(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.
  [**Cut&Count**: Cygan, Nederlof, Pilipczuk, Pilipczuk, van Rooij, Wojtaszczyk. 2011]

- $\mathcal{F} = \{K_5, K_{3,3}\}$: Vertex Planarization.
  Solvable in time $2^{\Theta(\mathsf{tw} \cdot \log \mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.
  [Jansen, Lokshtanov, Saurabh. 2014 + Pilipczuk. 2015]

With Julien Baste and Dimitrios M. Thilikos (2016-2020)

Objective

Determine, for every fixed $\mathcal{F}$, the (asymptotically) smallest function $f_{\mathcal{F}}$ such that $\mathcal{F}$-M-Deletion on n-vertex graphs can be solved in time

$$f_{\mathcal{F}}(\text{tw}) \cdot n^{\mathcal{O}(1)}.$$

With Julien Baste and Dimitrios M. Thilikos (2016-2020)

Objective

Determine, for every fixed $\mathcal{F}$, the (asymptotically) smallest function $f_{\mathcal{F}}$ such that $\mathcal{F}$-M-Deletion on n-vertex graphs can be solved in time

$$f_{\mathcal{F}}(\mathsf{tw}) \cdot n^{\mathcal{O}(1)}.$$

- We do not want to optimize the degree of the polynomial factor.
- We do not want to optimize the constants.
- Our hardness results hold under the ETH.

[Baste, S., Thilikos. **Hitting minors on bounded treewidth graphs. I. General upper bounds**. 2020]

[Baste, S., Thilikos. **Hitting minors on bounded treewidth graphs. II. Single-exponential algorithms**. 2020]

[Baste, S., Thilikos. **Hitting minors on bounded treewidth graphs. III. Lower bounds**. 2020]

[Baste, S., Thilikos. **Hitting minors on bounded treewidth graphs. IV. An optimal algorithm**. 2021]

Summary of our results

[1] Planar collection $\mathcal{F}$: contains at least one planar graph.

Summary of our results

- For every $\mathcal{F}$: $\mathcal{F}$-M-DELETION in time $2^{2^{\mathcal{O}(\mathrm{tw} \cdot \log \mathrm{tw})}} \cdot n^{\mathcal{O}(1)}$.

[1] Planar collection $\mathcal{F}$: contains at least one planar graph.

Summary of our results

- For every $\mathcal{F}$: $\mathcal{F}$-M-DELETION in time $2^{2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})}} \cdot n^{\mathcal{O}(1)}$.
- For every planar[1] $\mathcal{F}$: $\mathcal{F}$-M-DELETION in time $2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.

[1] Planar collection $\mathcal{F}$: contains at least one planar graph.

Summary of our results

- For every $\mathcal{F}$: $\mathcal{F}$-M-Deletion in time $2^{2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})}} \cdot n^{\mathcal{O}(1)}$.
- For every ~~planar~~[1] $\mathcal{F}$: $\mathcal{F}$-M-Deletion in time $2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.

[1] Planar collection $\mathcal{F}$: contains at least one planar graph.

Summary of our results

- For every $\mathcal{F}$: $\mathcal{F}$-M-Deletion in time $2^{2^{\mathcal{O}(\mathsf{tw}\cdot\log \mathsf{tw})}} \cdot n^{\mathcal{O}(1)}$.
- For every ~~planar~~[1] $\mathcal{F}$: $\mathcal{F}$-M-Deletion in time $2^{\mathcal{O}(\mathsf{tw}\cdot\log \mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.
- G planar: $\mathcal{F}$-M-Deletion in time $2^{\mathcal{O}(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.

[1] Planar collection $\mathcal{F}$: contains at least one planar graph.

Summary of our results

- For every $\mathcal{F}$: $\mathcal{F}$-M-Deletion in time $2^{2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})}} \cdot n^{\mathcal{O}(1)}$.
- For every ~~planar~~[1] $\mathcal{F}$: $\mathcal{F}$-M-Deletion in time $2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.
- G planar: $\mathcal{F}$-M-Deletion in time $2^{\mathcal{O}(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.
- For every $\mathcal{F}$: $\mathcal{F}$-M-Deletion not solvable in time $2^{o(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$ unless the ETH fails, even if G planar.

[1] Planar collection $\mathcal{F}$: contains at least one planar graph.

Summary of our results

- For every $\mathcal{F}$: $\mathcal{F}$-M-Deletion in time $2^{2^{\mathcal{O}(\mathsf{tw}\cdot\log \mathsf{tw})}} \cdot n^{\mathcal{O}(1)}$.
- For every ~~planar~~[1] $\mathcal{F}$: $\mathcal{F}$-M-Deletion in time $2^{\mathcal{O}(\mathsf{tw}\cdot\log \mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.
- G planar: $\mathcal{F}$-M-Deletion in time $2^{\mathcal{O}(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.
- For every $\mathcal{F}$: $\mathcal{F}$-M-Deletion not solvable in time $2^{o(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$ unless the ETH fails, even if G planar.
- $\mathcal{F} = \{H\}$, H connected:

[1] Planar collection $\mathcal{F}$: contains at least one planar graph.

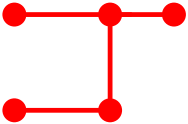
Summary of our results

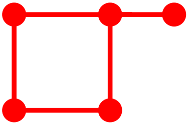
- For every $\mathcal{F}$: $\mathcal{F}$-M-DELETION in time $2^{2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})}} \cdot n^{\mathcal{O}(1)}$.
- For every ~~planar~~[1] $\mathcal{F}$: $\mathcal{F}$-M-DELETION in time $2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.
- G planar: $\mathcal{F}$-M-DELETION in time $2^{\mathcal{O}(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.
- For every $\mathcal{F}$: $\mathcal{F}$-M-DELETION not solvable in time $2^{o(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$ unless the ETH fails, even if G planar.
- $\mathcal{F} = \{H\}$, H connected: complete tight dichotomy...

[1] Planar collection $\mathcal{F}$: contains at least one planar graph.

A dichotomy for hitting a connected minor

A dichotomy for hitting a connected minor

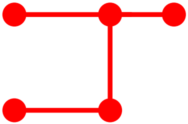


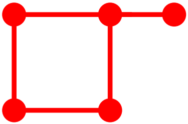


Theorem (Baste, S., Thilikos. 2016-2020)

Let H be a connected graph.

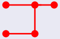
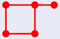
A dichotomy for hitting a connected minor



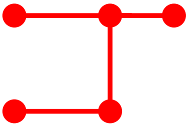


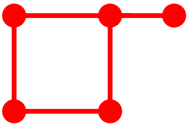
Theorem (Baste, S., Thilikos. 2016-2020)

Let H be a connected graph.
The $\{H\}$-M-Deletion problem is solvable in time

- $2^{\mathcal{O}(\mathrm{tw})} \cdot n^{\mathcal{O}(1)}$, *if* $H \leqslant_c$  *or* $H \leqslant_c$ .

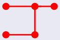
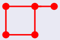
A dichotomy for hitting a connected minor



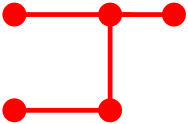


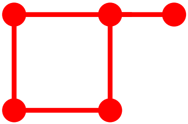
Theorem (Baste, S., Thilikos. 2016-2020)

Let H *be a connected graph.*
The $\{H\}$-M-DELETION *problem is solvable in time*

- $2^{\mathcal{O}(\text{tw})} \cdot n^{\mathcal{O}(1)}$, *if* $H \leqslant_{\mathsf{c}}$  *or* $H \leqslant_{\mathsf{c}}$ .
- $2^{\mathcal{O}(\text{tw} \cdot \log \text{tw})} \cdot n^{\mathcal{O}(1)}$, *otherwise.*

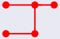
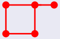
A dichotomy for hitting a connected minor





Theorem (Baste, S., Thilikos. 2016-2020)

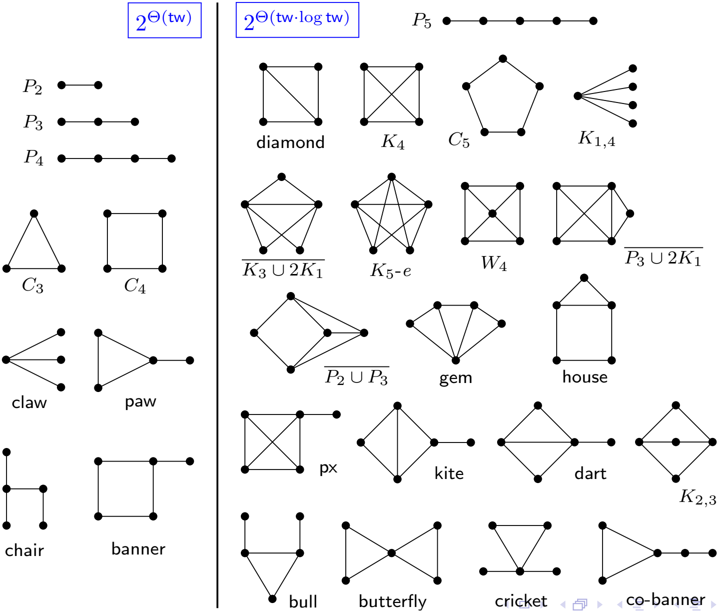
Let H *be a connected graph.*
The $\{H\}$-M-Deletion *problem is solvable in time*

- $2^{\mathcal{O}(\mathrm{tw})} \cdot n^{\mathcal{O}(1)}$, *if* $H \leqslant_c$  *or* $H \leqslant_c$  .
- $2^{\mathcal{O}(\mathrm{tw} \cdot \log \mathrm{tw})} \cdot n^{\mathcal{O}(1)}$, *otherwise.*

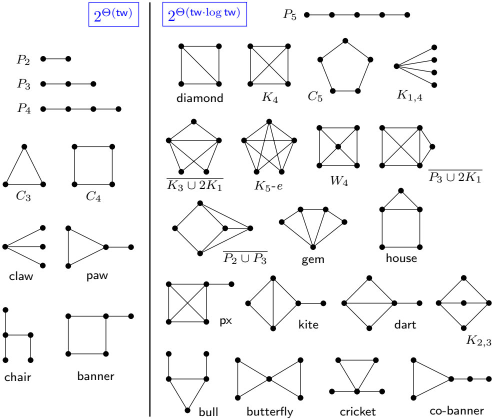
In both cases, the running time is asymptotically optimal under the ETH.

skip

Complexity of hitting a single connected minor H

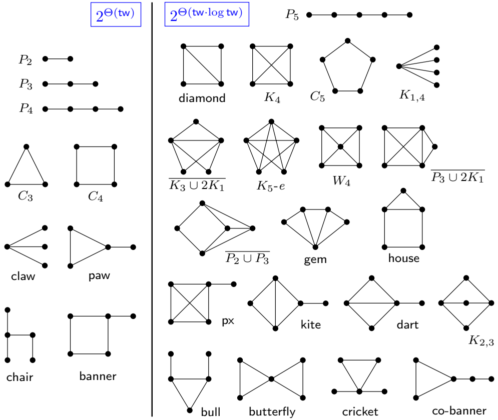


A compact statement for a single connected graph

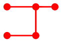
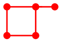


All these cases can be succinctly described as follows:

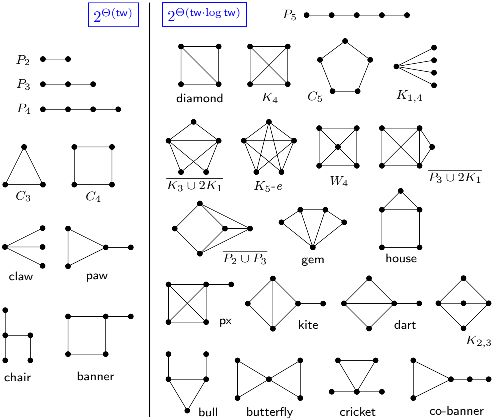
A compact statement for a single connected graph



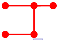
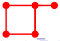
All these cases can be succinctly described as follows:

- All graphs on the left are contractions of  or 

A compact statement for a single connected graph



All these cases can be succinctly described as follows:

- All graphs on the left are contractions of  or 
- All graphs on the right are not contractions of  or 

Next subsection is...

We parameterize by the size of the desired solution

$\mathcal{F}$-M-Deletion
Input: A graph G and an integer k.
Parameter: k.
Question: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

We parameterize by the size of the desired solution

$\mathcal{F}$-M-Deletion
Input: A graph G and an integer k.
Parameter: k.
Question: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

It is easy to see that, for every $k \geqslant 1$, the class of graphs

$$\mathcal{C}_k = \{G \mid (G, k) \text{ is a positive instance of } \mathcal{F}\text{-M-Deletion}\}$$

is minor-closed.

We parameterize by the size of the desired solution

$\mathcal{F}$-M-Deletion
Input: A graph G and an integer k.
Parameter: k.
Question: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

It is easy to see that, for every $k \geqslant 1$, the class of graphs

$$\mathcal{C}_k = \{G \mid (G, k) \text{ is a positive instance of } \mathcal{F}\text{-M-Deletion}\}$$

is minor-closed.

Theorem (Robertson and Seymour. 1983-2004)

For every minor-closed graph class $\mathcal{C}$, deciding whether an n-vertex graph G belongs to $\mathcal{C}$ can be solved in time $f(\mathcal{C}) \cdot n^2$.

We parameterize by the size of the desired solution

$\mathcal{F}$-M-DELETION
Input: A graph G and an integer k.
Parameter: k.
Question: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

It is easy to see that, for every $k \geqslant 1$, the class of graphs

$$\mathcal{C}_k = \{G \mid (G, k) \text{ is a positive instance of } \mathcal{F}\text{-M-DELETION}\}$$

is minor-closed.

Theorem (Robertson and Seymour. 1983-2004)

For every minor-closed graph class $\mathcal{C}$, deciding whether an n-vertex graph G belongs to $\mathcal{C}$ can be solved in time $f(\mathcal{C}) \cdot n^2$.

For every $k \geq 1$, there exists an FPT algorithm for $\mathcal{F}$-M-DELETION.

We parameterize by the size of the desired solution

$\mathcal{F}$-M-Deletion
Input: A graph G and an integer k.
Parameter: k.
Question: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

It is easy to see that, for every $k \geqslant 1$, the class of graphs

$$\mathcal{C}_k = \{G \mid (G,k) \text{ is a positive instance of } \mathcal{F}\text{-M-Deletion}\}$$

is minor-closed.

Theorem (Robertson and Seymour. 1983-2004)

For every minor-closed graph class $\mathcal{C}$, deciding whether an n-vertex graph G belongs to $\mathcal{C}$ can be solved in time $f(\mathcal{C}) \cdot n^2$.

For every $k \geq 1$, there exists an FPT algorithm for $\mathcal{F}$-M-Deletion.

But... only existential, non-uniform, $f(\mathcal{C}_k)$ astronomical.

Can we do better?

- The function $f(\mathcal{C}_k)$ is constructible. [Adler, Grohe, Kreutzer. 2008]

Can we do better?

- The function $f(\mathcal{C}_k)$ is constructible. [Adler, Grohe, Kreutzer. 2008]
- If $\mathcal{F}$ contains a planar graph: $2^{\mathcal{O}_{\mathcal{F}}(k)} \cdot n^{\mathcal{O}(1)}$.

[Fomin, Lokshtanov, Misra, Saurabh. 2012]

[Kim, Langer, Paul, Reidl, Rossmanith, S., Sikdar. 2013]

Can we do better?

- The function $f(\mathcal{C}_k)$ is constructible. [Adler, Grohe, Kreutzer. 2008]
- If $\mathcal{F}$ contains a planar graph: $2^{\mathcal{O}_{\mathcal{F}}(k)} \cdot n^{\mathcal{O}(1)}$.

  [Fomin, Lokshtanov, Misra, Saurabh. 2012]

  [Kim, Langer, Paul, Reidl, Rossmanith, S., Sikdar. 2013]
- For some non-planar collections $\mathcal{F}$:
  - $\mathcal{F} = \{K_5, K_{3,3}\}$: $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$. [Jansen, Lokshtanov, Saurabh. 2014]

Can we do better?

- The function $f(\mathcal{C}_k)$ is constructible. [Adler, Grohe, Kreutzer. 2008]
- If $\mathcal{F}$ contains a planar graph: $2^{\mathcal{O}_{\mathcal{F}}(k)} \cdot n^{\mathcal{O}(1)}$.

  [Fomin, Lokshtanov, Misra, Saurabh. 2012]

  [Kim, Langer, Paul, Reidl, Rossmanith, S., Sikdar. 2013]
- For some non-planar collections $\mathcal{F}$:
  - $\mathcal{F} = \{K_5, K_{3,3}\}$: $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$. [Jansen, Lokshtanov, Saurabh. 2014]
  - Deletion to genus at most g: $2^{\mathcal{O}_g(k^2 \log k)} \cdot n^{\mathcal{O}(1)}$. [Kociumaka, Ma. Pilipczuk. 2019]

Can we do better?

- The function $f(\mathcal{C}_k)$ is constructible. [Adler, Grohe, Kreutzer. 2008]
- If $\mathcal{F}$ contains a planar graph: $2^{\mathcal{O}_{\mathcal{F}}(k)} \cdot n^{\mathcal{O}(1)}$.

  [Fomin, Lokshtanov, Misra, Saurabh. 2012]

  [Kim, Langer, Paul, Reidl, Rossmanith, S., Sikdar. 2013]
- For some non-planar collections $\mathcal{F}$:
  - $\mathcal{F} = \{K_5, K_{3,3}\}$: $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$. [Jansen, Lokshtanov, Saurabh. 2014]
  - Deletion to genus at most g: $2^{\mathcal{O}_g(k^2 \log k)} \cdot n^{\mathcal{O}(1)}$. [Kociumaka, Ma. Pilipczuk. 2019]
- For every $\mathcal{F}$, some enormous explicit function $f_{\mathcal{F}}(k)$ can be derived from an FPT algorithm for hitting topological minors:

$$f_{\mathcal{F}}(k) \cdot n^{\mathcal{O}(1)}.$$

[Fomin, Lokshtanov, Panolan, Saurabh, Zehavi. 2020]

Our results

Theorem (S., Stamoulis, Thilikos. 2020)

For all $\mathcal{F}$, the $\mathcal{F}$-M-DELETION problem can be solved in time $2^{\mathsf{poly}(k)} \cdot n^3$.

Here, $\mathsf{poly}(k)$ is a polynomial whose degree depends on $\mathcal{F}$.

Our results

Theorem (S., Stamoulis, Thilikos. 2020)

For all $\mathcal{F}$, the $\mathcal{F}$-M-Deletion problem can be solved in time $2^{\mathsf{poly}(k)} \cdot n^3$.

Here, $\mathsf{poly}(k)$ is a polynomial whose degree depends on $\mathcal{F}$.

Theorem (S., Stamoulis, Thilikos. 2020)

If $\mathcal{F}$ contains an apex graph, the $\mathcal{F}$-M-Deletion problem can be solved in time $2^{\mathsf{poly}(k)} \cdot n^2$.

Again, $\mathsf{poly}(k)$ is a polynomial whose degree depends on $\mathcal{F}$.

Our results

Theorem (S., Stamoulis, Thilikos. 2020)

For all $\mathcal{F}$, the $\mathcal{F}$-M-DELETION problem can be solved in time $2^{\mathsf{poly}(k)} \cdot n^3$.

Here, $\mathsf{poly}(k)$ is a polynomial whose degree depends on $\mathcal{F}$.

Theorem (S., Stamoulis, Thilikos. 2020)

If $\mathcal{F}$ contains an apex graph, the $\mathcal{F}$-M-DELETION problem can be solved in time $2^{\mathsf{poly}(k)} \cdot n^2$.

Again, $\mathsf{poly}(k)$ is a polynomial whose degree depends on $\mathcal{F}$.

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

For all $\mathcal{F}$, the $\mathcal{F}$-M-DELETION problem can be solved in time $2^{\mathsf{poly}(k)} \cdot n^2$.

skip

Next section is...

Next subsection is...

Recall the statement of the problem

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-DELETION
Input: A graph G and an integer k.
Parameter: The treewidth tw of G.
Question: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

We have three types of results

We have three types of results

1. General algorithms
   - For every $\mathcal{F}$: time $2^{2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})}} \cdot n^{\mathcal{O}(1)}$.
   - $\mathcal{F}$ planar: time $2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.
   - $\mathcal{F}$ ~~planar~~: time $2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.
   - G planar: time $2^{\mathcal{O}(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$.

We have three types of results

1. General algorithms
   - For every $\mathcal{F}$: time $2^{2^{\mathcal{O}(\mathrm{tw} \cdot \log \mathrm{tw})}} \cdot n^{\mathcal{O}(1)}$.
   - $\mathcal{F}$ planar: time $2^{\mathcal{O}(\mathrm{tw} \cdot \log \mathrm{tw})} \cdot n^{\mathcal{O}(1)}$.
   - $\mathcal{F}$ ~~planar~~: time $2^{\mathcal{O}(\mathrm{tw} \cdot \log \mathrm{tw})} \cdot n^{\mathcal{O}(1)}$.
   - G planar: time $2^{\mathcal{O}(\mathrm{tw})} \cdot n^{\mathcal{O}(1)}$.

2. Ad-hoc single-exponential algorithms
   - Some use "typical" dynamic programming.
   - Some use the rank-based approach. [Bodlaender, Cygan, Kratsch, Nederlof. 2013]

We have three types of results

1. General algorithms
   - For every $\mathcal{F}$: time $2^{2^{\mathcal{O}(\text{tw} \cdot \log \text{tw})}} \cdot n^{\mathcal{O}(1)}$.
   - $\mathcal{F}$ planar: time $2^{\mathcal{O}(\text{tw} \cdot \log \text{tw})} \cdot n^{\mathcal{O}(1)}$.
   - $\mathcal{F}$ ~~planar~~: time $2^{\mathcal{O}(\text{tw} \cdot \log \text{tw})} \cdot n^{\mathcal{O}(1)}$.
   - G planar: time $2^{\mathcal{O}(\text{tw})} \cdot n^{\mathcal{O}(1)}$.

2. Ad-hoc single-exponential algorithms
   - Some use "typical" dynamic programming.
   - Some use the rank-based approach. [Bodlaender, Cygan, Kratsch, Nederlof. 2013]

3. Lower bounds under the ETH
   - $2^{o(\text{tw})}$ is "easy".
   - $2^{o(\text{tw} \cdot \log \text{tw})}$ is much more involved and we get ideas from:

[Lokshtanov, Marx, Saurabh. 2011] [Marcin Pilipczuk. 2017] [Bonnet, Brettell, Kwon, Marx. 2017]

skip

We have three types of results

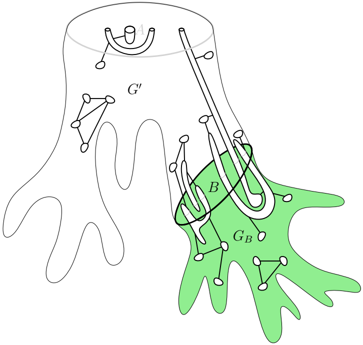
1. General algorithms
   - For every $\mathcal{F}$: time $2^{2^{\mathcal{O}(\text{tw}\cdot\log\text{tw})}} \cdot n^{\mathcal{O}(1)}$.
   - $\mathcal{F}$ planar: time $2^{\mathcal{O}(\text{tw}\cdot\log\text{tw})} \cdot n^{\mathcal{O}(1)}$.
   - ★ $\mathcal{F}$ ~~planar~~: time $2^{\mathcal{O}(\text{tw}\cdot\log\text{tw})} \cdot n^{\mathcal{O}(1)}$.
   - G planar: time $2^{\mathcal{O}(\text{tw})} \cdot n^{\mathcal{O}(1)}$.
2. Ad-hoc single-exponential algorithms
   - Some use "typical" dynamic programming.
   - Some use the rank-based approach. [Bodlaender, Cygan, Kratsch, Nederlof. 2013]
3. Lower bounds under the ETH
   - $2^{o(\text{tw})}$ is "easy".
   - $2^{o(\text{tw}\cdot\log\text{tw})}$ is much more involved and we get ideas from:

     [Lokshtanov, Marx, Saurabh. 2011] [Marcin Pilipczuk. 2017] [Bonnet, Brettell, Kwon, Marx. 2017]

skip

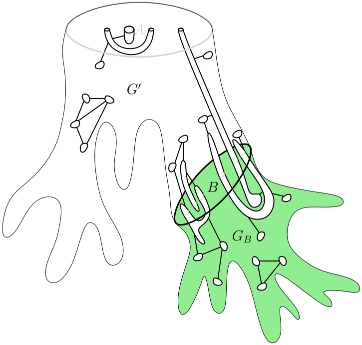
Algorithm in time $2^{\mathcal{O}_{\mathcal{F}}(\mathsf{tw}\cdot\log \mathsf{tw})}\cdot n^{\mathcal{O}(1)}$ for any collection $\mathcal{F}$

[Fig. by Valentin Garnero]



Algorithm in time $2^{\mathcal{O}_{\mathcal{F}}(\text{tw} \cdot \log \text{tw})} \cdot n^{\mathcal{O}(1)}$ for any collection $\mathcal{F}$

[Fig. by Valentin Garnero]



- For a fixed $\mathcal{F}$, we define an equivalence relation $\equiv^{(\mathcal{F},t)}$ on t-boundaried graphs:

$$G_1 \equiv^{(\mathcal{F},t)} G_2 \quad \text{if } \forall G' \in \mathcal{B}^t,$$
$$\mathcal{F} \leqslant_{\text{m}} G' \oplus G_1 \iff \mathcal{F} \leqslant_{\text{m}} G' \oplus G_2.$$

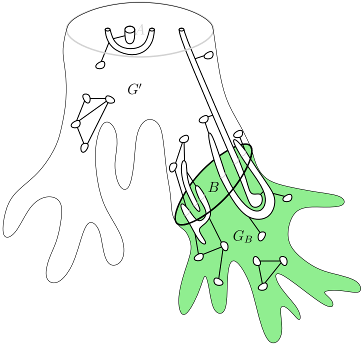
Algorithm in time $2^{\mathcal{O}_{\mathcal{F}}(\mathrm{tw} \cdot \log \mathrm{tw})} \cdot n^{\mathcal{O}(1)}$ for any collection $\mathcal{F}$

[Fig. by Valentin Garnero]

- For a fixed $\mathcal{F}$, we define an equivalence relation $\equiv^{(\mathcal{F},t)}$ on t-boundaried graphs:

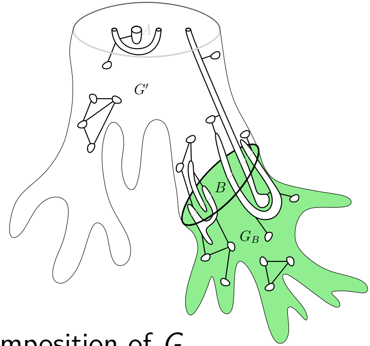
  $G_1 \equiv^{(\mathcal{F},t)} G_2$ if $\forall G' \in \mathcal{B}^t$,
  $\mathcal{F} \leqslant_{\mathrm{m}} G' \oplus G_1 \iff \mathcal{F} \leqslant_{\mathrm{m}} G' \oplus G_2$.

- $\mathcal{R}^{(\mathcal{F},t)}$: set of minimum-size representatives of $\equiv^{(\mathcal{F},t)}$.



Algorithm in time $2^{\mathcal{O}_{\mathcal{F}}(\text{tw}\cdot\log \text{tw})} \cdot n^{\mathcal{O}(1)}$ for any collection $\mathcal{F}$

[Fig. by Valentin Garnero]



- For a fixed $\mathcal{F}$, we define an equivalence relation $\equiv^{(\mathcal{F},t)}$ on t-boundaried graphs:

$$G_1 \equiv^{(\mathcal{F},t)} G_2 \quad \text{if } \forall G' \in \mathcal{B}^t,$$
$$\mathcal{F} \leqslant_{\text{m}} G' \oplus G_1 \iff \mathcal{F} \leqslant_{\text{m}} G' \oplus G_2.$$

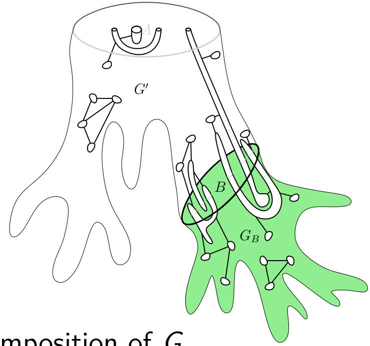
- $\mathcal{R}^{(\mathcal{F},t)}$: set of minimum-size representatives of $\equiv^{(\mathcal{F},t)}$.

- We compute, using DP over a tree decomposition of G, the following parameter for every representative $R \in \mathcal{R}^{(\mathcal{F},t)}$:

$$\mathbf{p}(G_B, R) = \min\{|S| \ : \ S \subseteq V(G_B) \ \wedge \ \text{rep}_{\mathcal{F},t}(G_B \setminus S) = R\}$$

Algorithm in time $2^{\mathcal{O}_{\mathcal{F}}(\mathrm{tw}\cdot\log \mathrm{tw})} \cdot n^{\mathcal{O}(1)}$ for any collection $\mathcal{F}$

[Fig. by Valentin Garnero]



- For a fixed $\mathcal{F}$, we define an equivalence relation $\equiv^{(\mathcal{F},t)}$ on t-boundaried graphs:

$$G_1 \equiv^{(\mathcal{F},t)} G_2 \quad \text{if } \forall G' \in \mathcal{B}^t,$$
$$\mathcal{F} \leqslant_{\mathrm{m}} G' \oplus G_1 \iff \mathcal{F} \leqslant_{\mathrm{m}} G' \oplus G_2.$$

- $\mathcal{R}^{(\mathcal{F},t)}$: set of minimum-size representatives of $\equiv^{(\mathcal{F},t)}$.

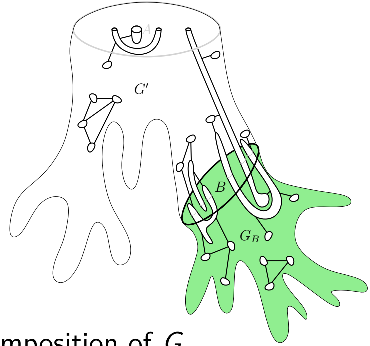
- We compute, using DP over a tree decomposition of G, the following parameter for every representative $R \in \mathcal{R}^{(\mathcal{F},t)}$:

$$\mathbf{p}(G_B, R) = \min\{|S| \; : \; S \subseteq V(G_B) \wedge \mathrm{rep}_{\mathcal{F},t}(G_B \setminus S) = R\}$$

- This gives an algorithm running in time $|\mathcal{R}^{(\mathcal{F},t)}|^{\mathcal{O}(1)} \cdot n^{\mathcal{O}(1)}$.

Algorithm in time $2^{\mathcal{O}_{\mathcal{F}}(\mathrm{tw}\cdot\log \mathrm{tw})} \cdot n^{\mathcal{O}(1)}$ for any collection $\mathcal{F}$

[Fig. by Valentin Garnero]



- For a fixed $\mathcal{F}$, we define an equivalence relation $\equiv^{(\mathcal{F},t)}$ on t-boundaried graphs:

$$G_1 \equiv^{(\mathcal{F},t)} G_2 \quad \text{if } \forall G' \in \mathcal{B}^t,$$
$$\mathcal{F} \leqslant_{\mathrm{m}} G' \oplus G_1 \iff \mathcal{F} \leqslant_{\mathrm{m}} G' \oplus G_2.$$

- $\mathcal{R}^{(\mathcal{F},t)}$: set of minimum-size representatives of $\equiv^{(\mathcal{F},t)}$.

- We compute, using DP over a tree decomposition of G, the following parameter for every representative $R \in \mathcal{R}^{(\mathcal{F},t)}$:

$$\mathbf{p}(G_B, R) = \min\{|S| \ : \ S \subseteq V(G_B) \ \wedge \ \mathrm{rep}_{\mathcal{F},t}(G_B \setminus S) = R\}$$

- This gives an algorithm running in time $|\mathcal{R}^{(\mathcal{F},t)}|^{\mathcal{O}(1)} \cdot n^{\mathcal{O}(1)}$.

- Goal: Bound the number of representatives: $|\mathcal{R}^{(\mathcal{F},t)}| = 2^{\mathcal{O}_{\mathcal{F}}(\mathrm{tw}\cdot\log \mathrm{tw})}$.

Bounding the set of representatives

- $\mathcal{R}^{(\mathcal{F},t)}$: set of minimum-size representatives of $\equiv^{(\mathcal{F},t)}$.

Bounding the set of representatives

- $\mathcal{R}^{(\mathcal{F},t)}$: set of minimum-size representatives of $\equiv^{(\mathcal{F},t)}$.
- Suppose that we can prove that, for every $R \in \mathcal{R}^{(\mathcal{F},t)}$,

$$|V(R)| = \mathcal{O}_{\mathcal{F}}(t).$$

Bounding the set of representatives

- $\mathcal{R}^{(\mathcal{F},t)}$: set of minimum-size representatives of $\equiv^{(\mathcal{F},t)}$.
- Suppose that we can prove that, for every $R \in \mathcal{R}^{(\mathcal{F},t)}$,

$$|V(R)| = \mathcal{O}_{\mathcal{F}}(t).$$

- Then, by the sparsity of the representatives,

$$|\mathcal{R}^{(\mathcal{F},t)}| = \mathcal{O}_{\mathcal{F}}(1) \cdot \binom{t^2}{t} = 2^{\mathcal{O}_{\mathcal{F}}(t \cdot \log t)},$$

and we are done!

Bounding the set of representatives

- $\mathcal{R}^{(\mathcal{F},t)}$: set of minimum-size representatives of $\equiv^{(\mathcal{F},t)}$.
- Suppose that we can prove that, for every $R \in \mathcal{R}^{(\mathcal{F},t)}$,
$$|V(R)| = \mathcal{O}_{\mathcal{F}}(t).$$
- Then, by the sparsity of the representatives,
$$|\mathcal{R}^{(\mathcal{F},t)}| = \mathcal{O}_{\mathcal{F}}(1) \cdot \binom{t^2}{t} = 2^{\mathcal{O}_{\mathcal{F}}(t \cdot \log t)},$$
and we are done!
- Flat Wall Theorem [Robertson, Seymour. GMXIII. 1995]

Bounding the set of representatives

- $\mathcal{R}^{(\mathcal{F},t)}$: set of minimum-size representatives of $\equiv^{(\mathcal{F},t)}$.
- Suppose that we can prove that, for every $R \in \mathcal{R}^{(\mathcal{F},t)}$,

$$|V(R)| = \mathcal{O}_{\mathcal{F}}(t).$$

- Then, by the sparsity of the representatives,

$$|\mathcal{R}^{(\mathcal{F},t)}| = \mathcal{O}_{\mathcal{F}}(1) \cdot \binom{t^2}{t} = 2^{\mathcal{O}_{\mathcal{F}}(t \cdot \log t)},$$

and we are done!

- Flat Wall Theorem [Robertson, Seymour. GMXIII. 1995]

As a representative R is $\mathcal{F}$-minor-free, if $\mathrm{tw}(R \setminus B) > c_{\mathcal{F}}$,

Bounding the set of representatives

- $\mathcal{R}^{(\mathcal{F},t)}$: set of minimum-size representatives of $\equiv^{(\mathcal{F},t)}$.
- Suppose that we can prove that, for every $R \in \mathcal{R}^{(\mathcal{F},t)}$,
$$|V(R)| = \mathcal{O}_{\mathcal{F}}(t).$$
- Then, by the sparsity of the representatives,
$$|\mathcal{R}^{(\mathcal{F},t)}| = \mathcal{O}_{\mathcal{F}}(1) \cdot \binom{t^2}{t} = 2^{\mathcal{O}_{\mathcal{F}}(t \cdot \log t)},$$
and we are done!
- Flat Wall Theorem [Robertson, Seymour. GMXIII. 1995]

  As a representative R is $\mathcal{F}$-minor-free, if $\text{tw}(R \setminus B) > c_{\mathcal{F}}$, $R \setminus B$ contains a large flat wall,

Bounding the set of representatives

- $\mathcal{R}^{(\mathcal{F},t)}$: set of minimum-size representatives of $\equiv^{(\mathcal{F},t)}$.
- Suppose that we can prove that, for every $R \in \mathcal{R}^{(\mathcal{F},t)}$,

$$|V(R)| = \mathcal{O}_{\mathcal{F}}(t).$$

- Then, by the sparsity of the representatives,

$$|\mathcal{R}^{(\mathcal{F},t)}| = \mathcal{O}_{\mathcal{F}}(1) \cdot \binom{t^2}{t} = 2^{\mathcal{O}_{\mathcal{F}}(t \cdot \log t)},$$

  and we are done!

- Flat Wall Theorem [Robertson, Seymour. GMXIII. 1995]

  As a representative R is $\mathcal{F}$-minor-free, if $\text{tw}(R \setminus B) > c_{\mathcal{F}}$, $R \setminus B$ contains a large flat wall, where we can find an irrelevant vertex.

Next subsection is...

Basic principle of the irrelevant vertex technique

This technique was invented in [Robertson and Seymour. 1995]

Basic principle of the irrelevant vertex technique

This technique was invented in [Robertson and Seymour. 1995]

Disjoint Paths
Input: a graph G and k pairs of vertices $T = \{s_1, \ldots, s_k, t_1, \ldots, t_k\}$.
Question: does G contain k vertex-disjoint paths $P_1, \ldots, P_k$ such that P_i connects s_i to t_i?

Basic principle of the irrelevant vertex technique

This technique was invented in [Robertson and Seymour. 1995]

Disjoint Paths
Input: a graph G and k pairs of vertices $T = \{s_1, \ldots, s_k, t_1, \ldots, t_k\}$.
Question: does G contain k vertex-disjoint paths $P_1, \ldots, P_k$ such that P_i connects s_i to t_i?

Strategy:

1. If $\text{tw}(G) > f(k)$, find an irrelevant vertex:

   A vertex $v \in V(G)$ such that (G, T, k) and $(G \setminus v, T, k)$ are equivalent instances.

Basic principle of the irrelevant vertex technique

This technique was invented in [Robertson and Seymour. 1995]

Disjoint Paths
Input: a graph G and k pairs of vertices $T = \{s_1, \ldots, s_k, t_1, \ldots, t_k\}$.
Question: does G contain k vertex-disjoint paths $P_1, \ldots, P_k$ such that P_i connects s_i to t_i?

Strategy:

1. If $\text{tw}(G) > f(k)$, find an irrelevant vertex:

   A vertex $v \in V(G)$ such that (G, T, k) and $(G \setminus v, T, k)$ are equivalent instances.

2. Otherwise, if $\text{tw}(G) \leq f(k)$, solve the problem using dynamic programming (by Courcelle).

How to find an irrelevant vertex when the treewidth is large?

How to find an irrelevant vertex when the treewidth is large?

By using the Grid Exclusion Theorem!

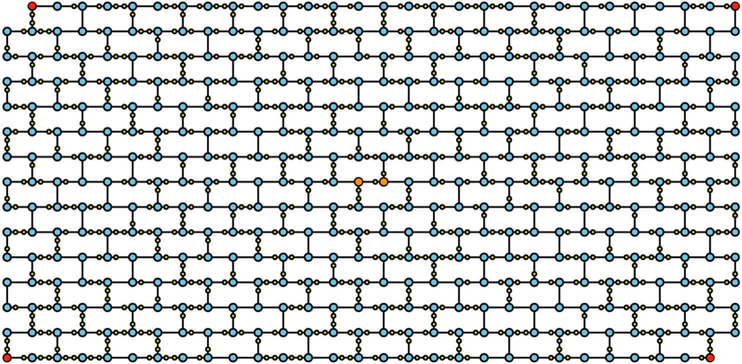
How to find an irrelevant vertex when the treewidth is large?

By using the Wall Exclusion Theorem!

How to find an irrelevant vertex when the treewidth is large?

Theorem (Robertson and Seymour. 1986)

For every integer $\ell > 0$, there is an integer $c(\ell)$ such that every graph of treewidth $\geq c(\ell)$ contains an ℓ-wall as a minor.

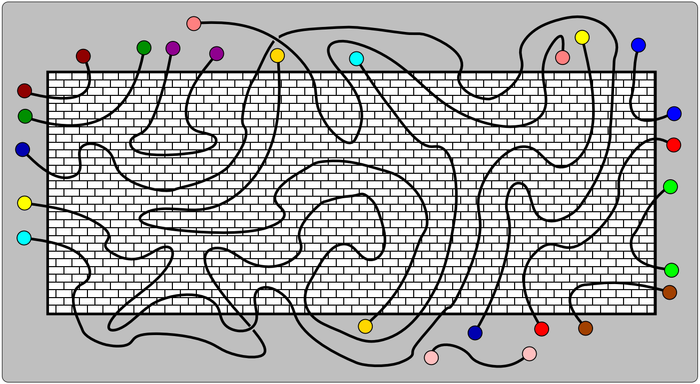


[Figure by Dimitrios M. Thilikos]

How to find an irrelevant vertex when the treewidth is large?

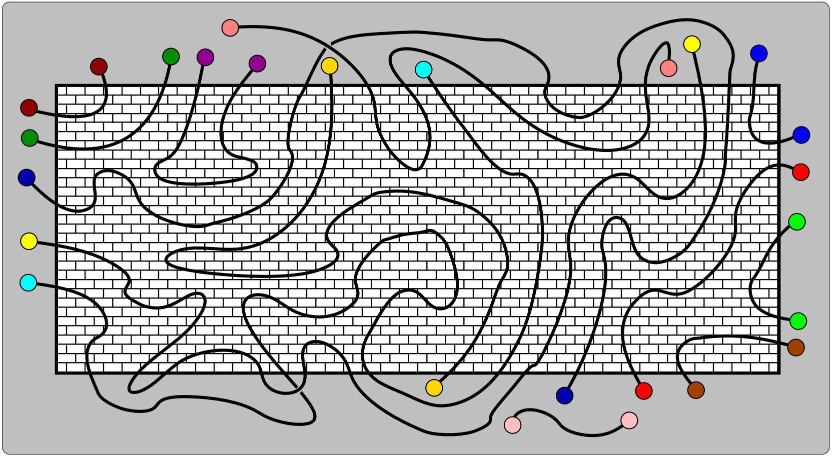
Theorem (Robertson and Seymour. 1986)

For every integer $\ell > 0$, there is an integer $c(\ell)$ such that every graph of treewidth $\geq c(\ell)$ contains an ℓ-wall as a minor.

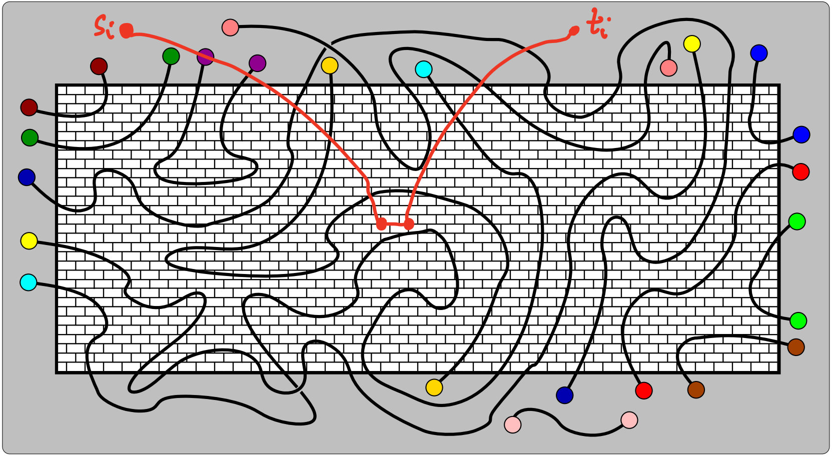


[Figure by Dimitrios M. Thilikos]

Goal: declare one of the central vertices of the wall irrelevant.



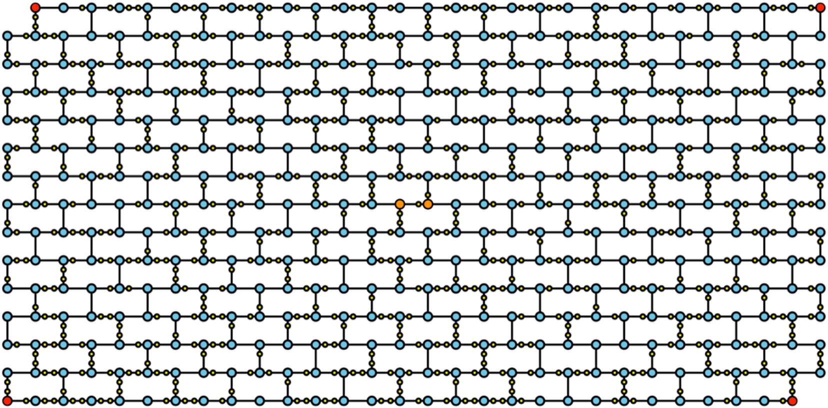
Goal: declare one of the central vertices of the wall irrelevant.



This is only possible if the wall is insulated from the exterior!

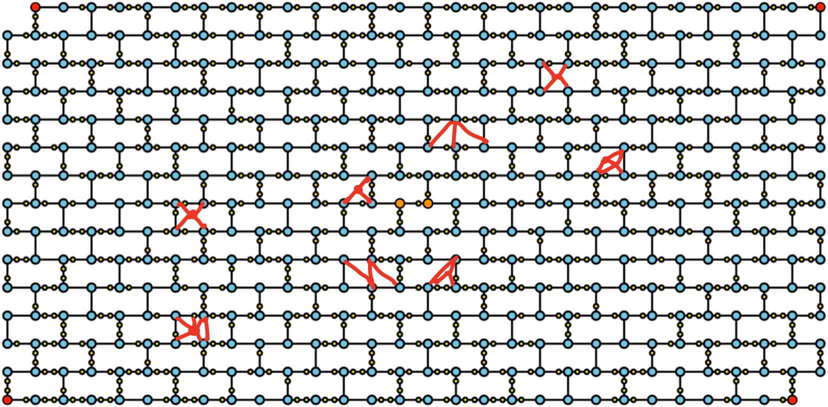
Flat walls

Goal: enrich the notion of wall so that we can insulate it from the exterior.



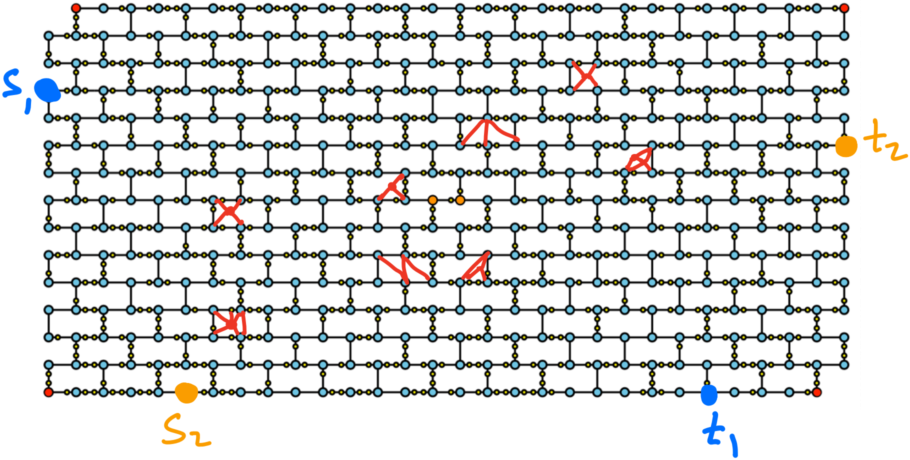
Flat walls

We need to allow some extra edges in the interior of the wall.



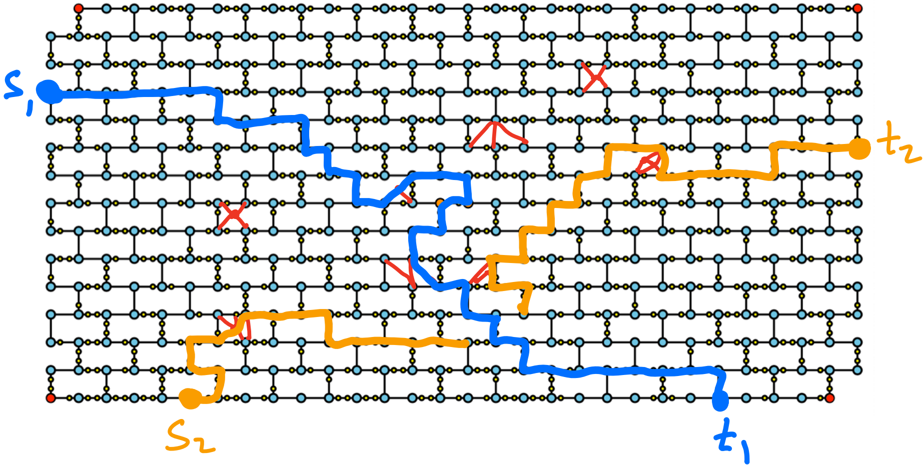
Flat walls

We impose a topological property that defines the "flatness" of the wall.



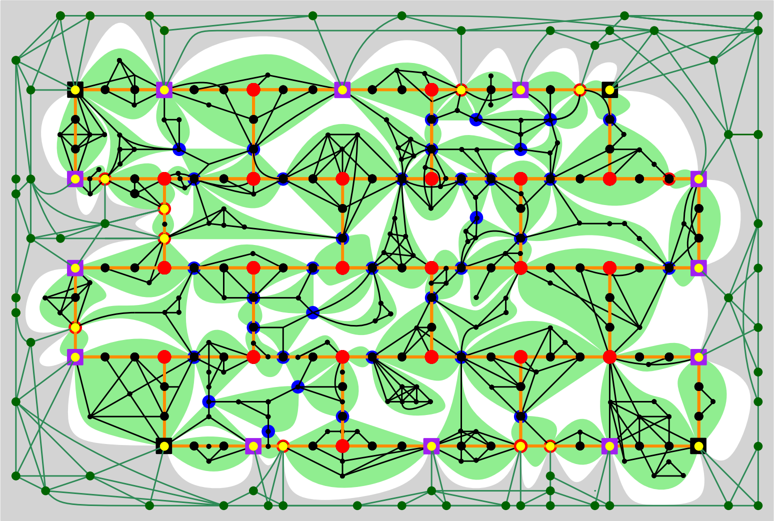
Flat walls

There are no crossing paths $s_1 - t_1$ and $s_2 - t_2$ from/to the perimeter.

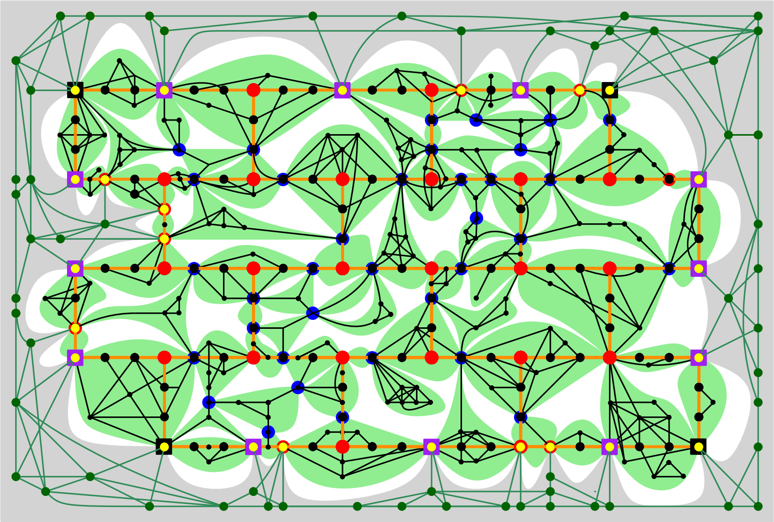


Flat walls

A real flat wall can be quite wild... [Figure by Dimitrios M. Thilikos]

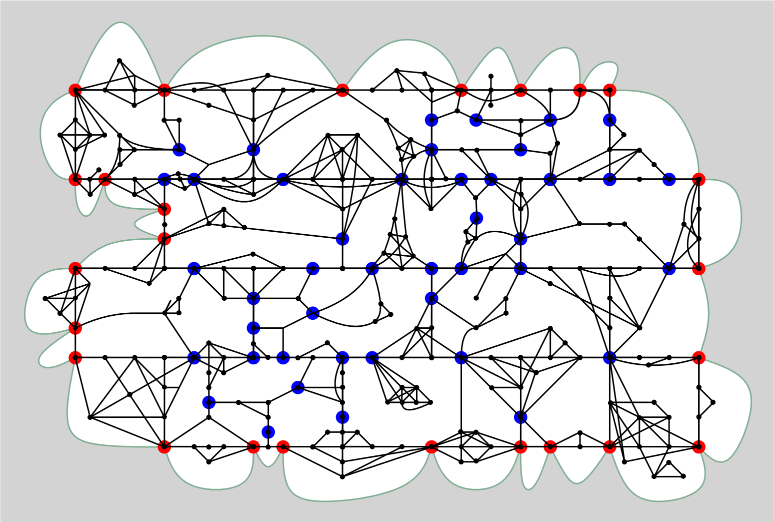


Flat walls: a bit more formal



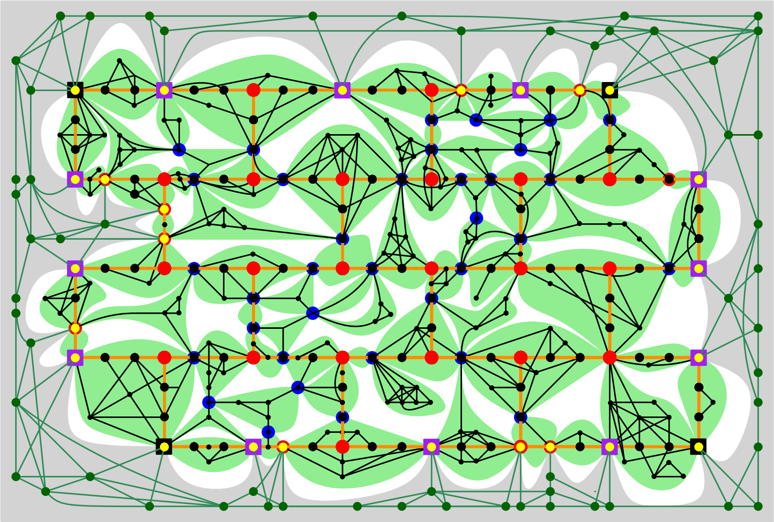
[Figures by Dimitrios M. Thilikos]

Flat walls: a bit more formal



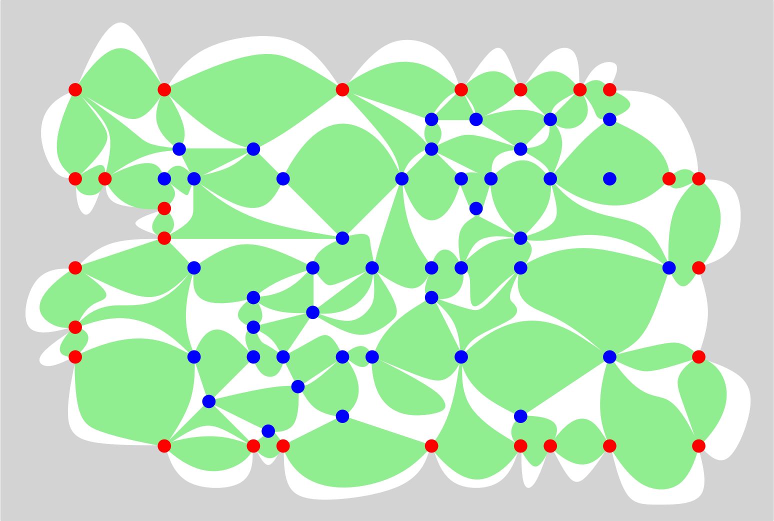
[Figures by Dimitrios M. Thilikos]

Flat walls: a bit more formal



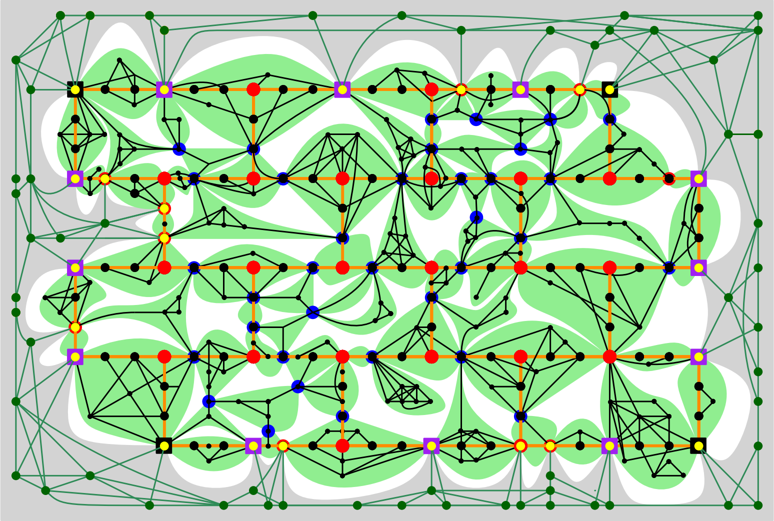
[Figures by Dimitrios M. Thilikos]

Flat walls: a bit more formal



[Figures by Dimitrios M. Thilikos]

Flat walls: a bit more formal



[Figures by Dimitrios M. Thilikos]

The Flat Wall Theorem

Theorem (Robertson and Seymour. 1995)

There exist recursive functions $f_1 : \mathbb{N}^2 \to \mathbb{N}$ *and* $f_2 : \mathbb{N} \to \mathbb{N}$, *such that for every graph* G *and every* $q, r \in \mathbb{N}$, *one of the following holds:*

The Flat Wall Theorem

Theorem (Robertson and Seymour. 1995)

There exist recursive functions $f_1 : \mathbb{N}^2 \to \mathbb{N}$ *and* $f_2 : \mathbb{N} \to \mathbb{N}$, *such that for every graph* G *and every* $q, r \in \mathbb{N}$, *one of the following holds:*

1. K_q *is a minor of* G.

The Flat Wall Theorem

Theorem (Robertson and Seymour. 1995)

There exist recursive functions $f_1 : \mathbb{N}^2 \to \mathbb{N}$ and $f_2 : \mathbb{N} \to \mathbb{N}$, such that for every graph G and every $q, r \in \mathbb{N}$, one of the following holds:

1. K_q *is a minor of* G.
2. *The treewidth of* G *is at most* $f_1(q, r)$.

The Flat Wall Theorem

Theorem (Robertson and Seymour. 1995)

There exist recursive functions $f_1 : \mathbb{N}^2 \rightarrow \mathbb{N}$ *and* $f_2 : \mathbb{N} \rightarrow \mathbb{N}$*, such that for every graph* G *and every* $q, r \in \mathbb{N}$*, one of the following holds:*

1. K_q *is a minor of* G.
2. *The treewidth of* G *is at most* $f_1(q, r)$.
3. *There exists* $A \subseteq V(G)$ *(apices) with* $|A| \leq f_2(q)$ *such that* $G \setminus A$ *contains as a subgraph a flat wall* W *of height* r.

The Flat Wall Theorem

Theorem (Robertson and Seymour. 1995)

There exist recursive functions $f_1 : \mathbb{N}^2 \to \mathbb{N}$ *and* $f_2 : \mathbb{N} \to \mathbb{N}$, *such that for every graph* G *and every* $q, r \in \mathbb{N}$, *one of the following holds:*

1. K_q *is a minor of* G.
2. *The treewidth of* G *is at most* $f_1(q, r)$.
3. *There exists* $A \subseteq V(G)$ *(apices) with* $|A| \leq f_2(q)$ *such that* $G \setminus A$ *contains as a subgraph a flat wall* W *of height* r.

There are several different variants and optimizations of this theorem...

[Chuzhoy. 2015]

[Kawarabayashi, Thomas, Wollan. 2018]

[S., Stamoulis, Thilikos. 2021]

The Flat Wall Theorem

Theorem (Robertson and Seymour. 1995)

There exist recursive functions $f_1 : \mathbb{N}^2 \to \mathbb{N}$ and $f_2 : \mathbb{N} \to \mathbb{N}$, such that for every graph G and every $q, r \in \mathbb{N}$, one of the following holds:

1. K_q *is a minor of* G.
2. *The treewidth of* G *is at most* $f_1(q, r)$.
3. *There exists* $A \subseteq V(G)$ *(apices) with* $|A| \leq f_2(q)$ *such that* $G \setminus A$ *contains as a subgraph a flat wall* W *of height* r.

There are several different variants and optimizations of this theorem...

[Chuzhoy. 2015]

[Kawarabayashi, Thomas, Wollan. 2018]

[S., Stamoulis, Thilikos. 2021]

Important: possible to find one of the outputs in time $f(q, r) \cdot |V(G)|$.

Back to the Disjoint Paths problem

Disjoint Paths
Input: a graph G and k pairs of vertices $T = \{s_1, \ldots, s_k, t_1, \ldots, t_k\}$.
Question: does G contain k vertex-disjoint paths $P_1, \ldots, P_k$ such that P_i connects s_i to t_i?

Back to the Disjoint Paths problem

Disjoint Paths

Input: a graph G and k pairs of vertices $T = \{s_1, \ldots, s_k, t_1, \ldots, t_k\}$.

Question: does G contain k vertex-disjoint paths $P_1, \ldots, P_k$ such that P_i connects s_i to t_i?

By the Flat Wall Theorem:

- If $\mathrm{tw}(G) \leq f(k)$: solve using dynamic programming.

Back to the DISJOINT PATHS problem

DISJOINT PATHS
Input: a graph G and k pairs of vertices $T = \{s_1, \ldots, s_k, t_1, \ldots, t_k\}$.
Question: does G contain k vertex-disjoint paths $P_1, \ldots, P_k$ such that P_i connects s_i to t_i?

By the Flat Wall Theorem:

- If $\text{tw}(G) \leq f(k)$: solve using dynamic programming.
- If G contains a $K_{g(k)}$-minor: "easy" to find an irrelevant vertex.

Back to the Disjoint Paths problem

Disjoint Paths

Input: a graph G and k pairs of vertices $T = \{s_1, \ldots, s_k, t_1, \ldots, t_k\}$.

Question: does G contain k vertex-disjoint paths $P_1, \ldots, P_k$ such that P_i connects s_i to t_i?

By the Flat Wall Theorem:

- If $\text{tw}(G) \leq f(k)$: solve using dynamic programming.
- If G contains a $K_{g(k)}$-minor: "easy" to find an irrelevant vertex.
- If G contains a "small" apex set A and a flat wall W in $G \setminus A$ of size at least $h(k)$: declare the central vertex of the flat wall irrelevant.

Back to the DISJOINT PATHS problem

DISJOINT PATHS

Input: a graph G and k pairs of vertices $T = \{s_1, \ldots, s_k, t_1, \ldots, t_k\}$.

Question: does G contain k vertex-disjoint paths $P_1, \ldots, P_k$ such that P_i connects s_i to t_i?

By the Flat Wall Theorem:

- If $\text{tw}(G) \le f(k)$: solve using dynamic programming.
- If G contains a $K_{g(k)}$-minor: "easy" to find an irrelevant vertex.
- If G contains a "small" apex set A and a flat wall W in $G \setminus A$ of size at least $h(k)$: declare the central vertex of the flat wall irrelevant.

The irrelevant vertex technique has been applied to many problems...

Back to the DISJOINT PATHS problem

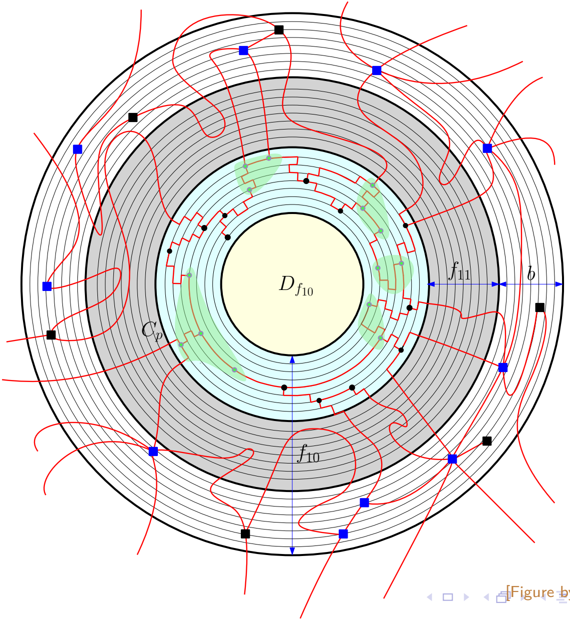
DISJOINT PATHS
Input: a graph G and k pairs of vertices $T = \{s_1, \ldots, s_k, t_1, \ldots, t_k\}$.
Question: does G contain k vertex-disjoint paths $P_1, \ldots, P_k$ such that P_i connects s_i to t_i?

By the Flat Wall Theorem:

- If $\text{tw}(G) \leq f(k)$: solve using dynamic programming.
- If G contains a $K_{g(k)}$-minor: "easy" to find an irrelevant vertex.
- If G contains a "small" apex set A and a flat wall W in $G \setminus A$ of size at least $h(k)$: declare the central vertex of the flat wall irrelevant.

The irrelevant vertex technique has been applied to many problems... usually with a lot of technical pain.

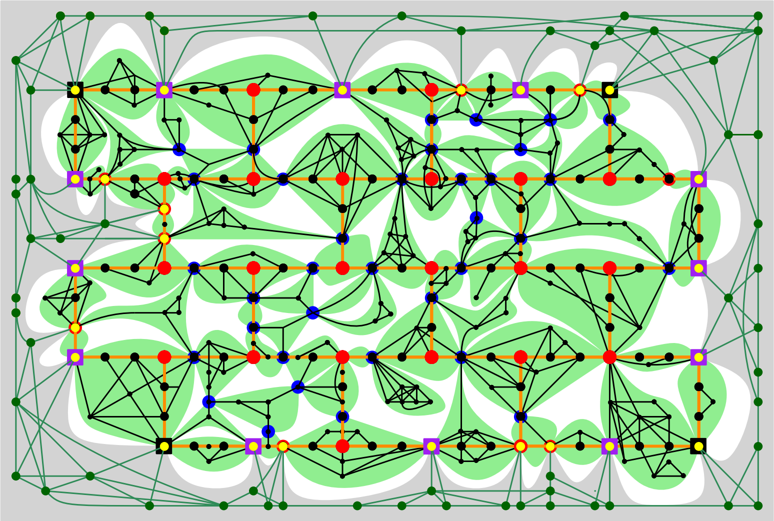
Rerouting inside a big flat wall...



[Figure by Dimitrios M. Thilikos]

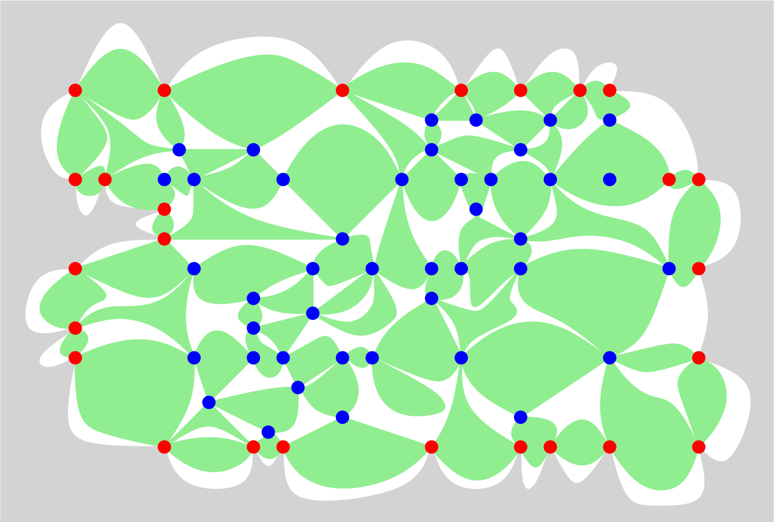
Crucial notion: homogeneity

In order to declare a vertex irrelevant for some problem, usually we need to consider a homogenous flat wall, which we proceed to define.



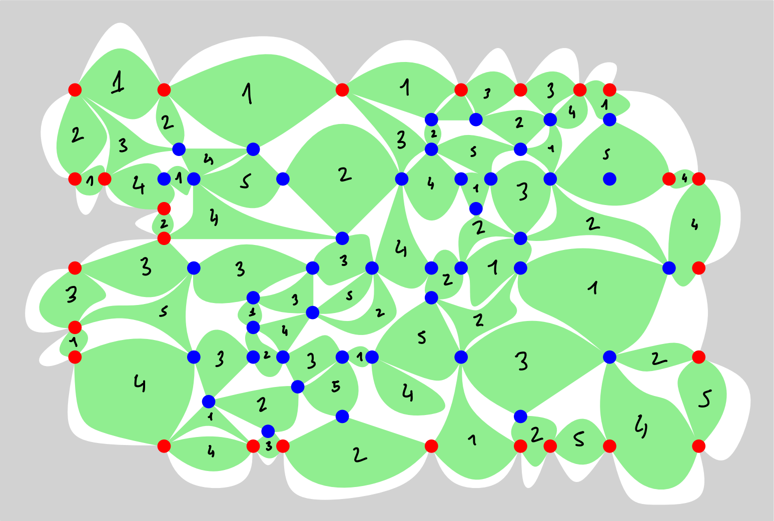
Crucial notion: homogeneity

We consider a flap-coloring encoding the relevant information of our favorite problem inside each flap (similar to tables of DP).



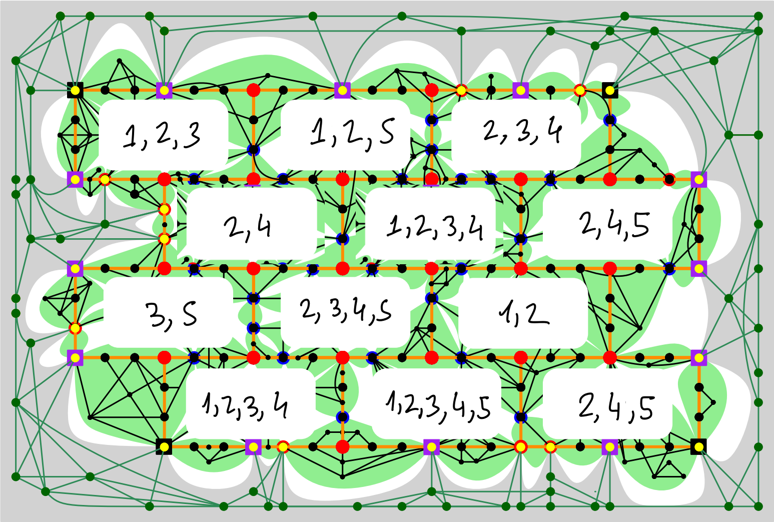
Crucial notion: homogeneity

We consider a flap-coloring encoding the relevant information of our favorite problem inside each flap (similar to tables of DP).



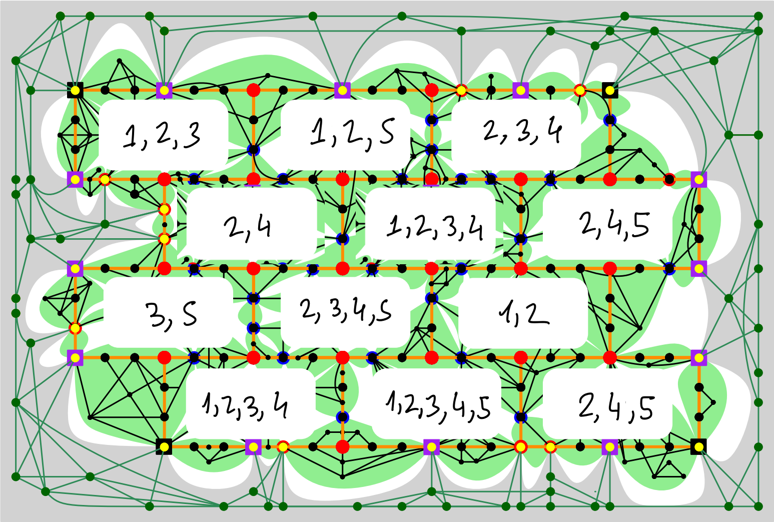
Crucial notion: homogeneity

For every brick of the wall, we define its palette as the colors appearing in the flaps it contains.



Crucial notion: homogeneity

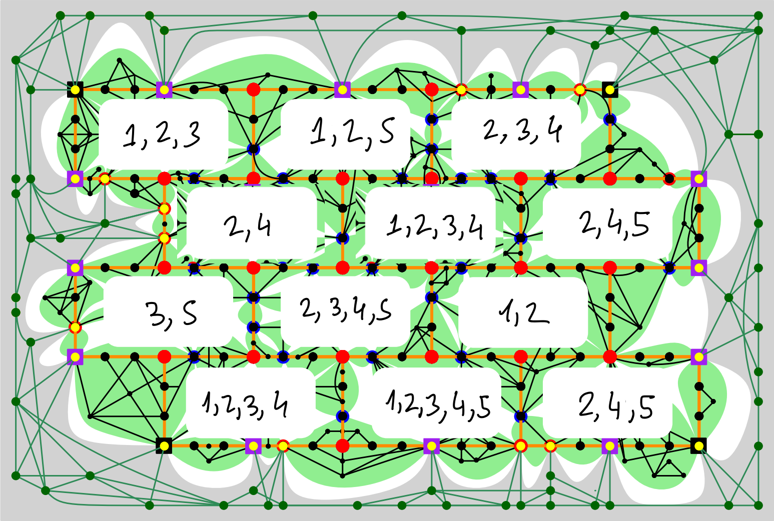
A flat wall is homogenous if every (internal) brick has the same palette.
Fact: every brick of a homogenous flat wall has the same "behavior".



Crucial notion: homogeneity

Price of homogeneity to obtain a homogenous flat r-wall (zooming):

If we have c colors, we need to start with a flat r^c-wall. (why?)



Next subsection is...

Recall the statement of the problem

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-Deletion
Input: A graph G and an integer k.
Parameter: k.
Question: Does G contain a set $S \subseteq V(G)$ with $|S| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

Recall the statement of the problem

Let $\mathcal{F}$ be a fixed finite collection of graphs.

$\mathcal{F}$-M-DELETION	
Input:	A graph G and an integer k.
Parameter:	k.
Question:	Does G contain a set $S \subseteq V(G)$ with $\|S\| \leqslant k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor?

Theorem (S., Stamoulis, Thilikos. 2020)

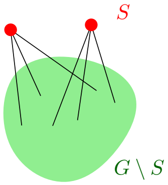
For all $\mathcal{F}$, the $\mathcal{F}$-M-DELETION problem can be solved in time $2^{\text{poly}(k)} \cdot n^3$.

General scheme of the algorithm:

[whole slide shamelessly borrowed from Giannos Stamoulis]

General scheme of the algorithm:

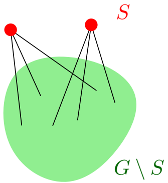
[whole slide shamelessly borrowed from Giannos Stamoulis]



Iterative compression: given solution S of size $k+1$, search solution of size k.

General scheme of the algorithm:

[whole slide shamelessly borrowed from Giannos Stamoulis]

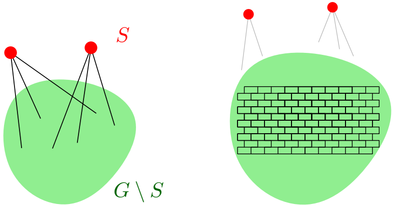


Iterative compression: given solution S of size $k+1$, search solution of size k.

If treewidth of $G \setminus S$ is "large enough" (as a polynomial function of k):

General scheme of the algorithm:

[whole slide shamelessly borrowed from Giannos Stamoulis]



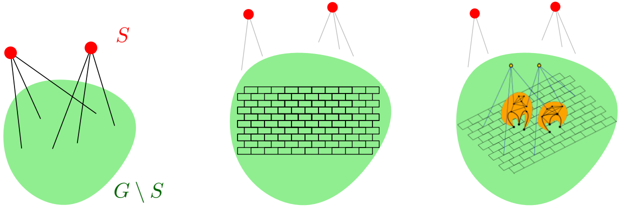
Iterative compression: given solution S of size $k+1$, search solution of size k.

If treewidth of $G \setminus S$ is "large enough" (as a polynomial function of k):

1. Find a "very very large" wall in $G \setminus S$.

General scheme of the algorithm:

[whole slide shamelessly borrowed from Giannos Stamoulis]



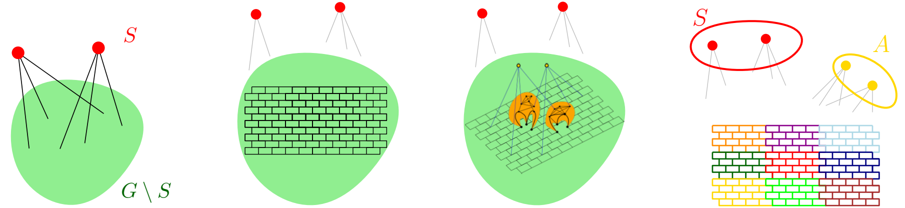
Iterative compression: given solution S of size $k+1$, search solution of size k.

If treewidth of $G \setminus S$ is "large enough" (as a polynomial function of k):

1. Find a "very very large" wall in $G \setminus S$.
2. Find a "very large" flat wall W of $G \setminus S$ with few apices A.

General scheme of the algorithm:

[whole slide shamelessly borrowed from Giannos Stamoulis]



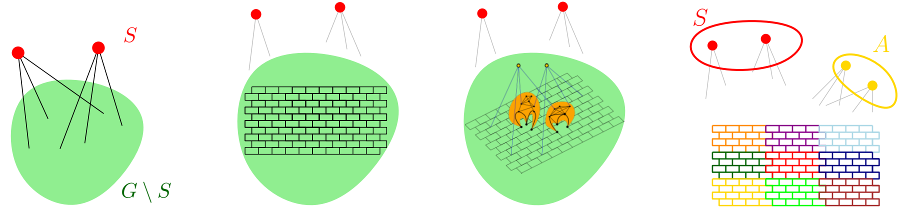
Iterative compression: given solution S of size $k+1$, search solution of size k.

If treewidth of $G \setminus S$ is "large enough" (as a polynomial function of k):

1. Find a "very very large" wall in $G \setminus S$.
2. Find a "very large" flat wall W of $G \setminus S$ with few apices A.
3. Find in W a packing of $\mathcal{O}_{\mathcal{F}}(k^4)$ disjoint "large" subwalls:

General scheme of the algorithm:

[whole slide shamelessly borrowed from Giannos Stamoulis]



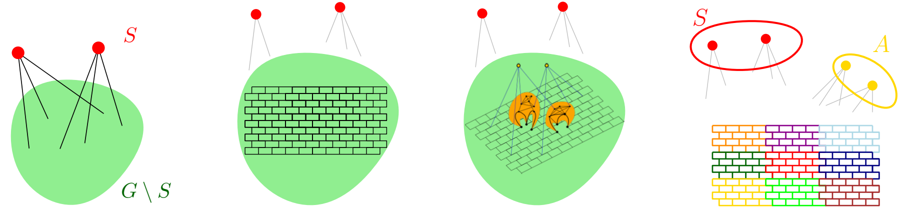
Iterative compression: given solution S of size $k+1$, search solution of size k.

If treewidth of $G \setminus S$ is "large enough" (as a polynomial function of k):

1. Find a "very very large" wall in $G \setminus S$.
2. Find a "very large" flat wall W of $G \setminus S$ with few apices A.
3. Find in W a packing of $\mathcal{O}_{\mathcal{F}}(k^4)$ disjoint "large" subwalls:
    - If every subwall has at least $|A|+1$ neighbors in $S \cup A$:

General scheme of the algorithm:

[whole slide shamelessly borrowed from Giannos Stamoulis]



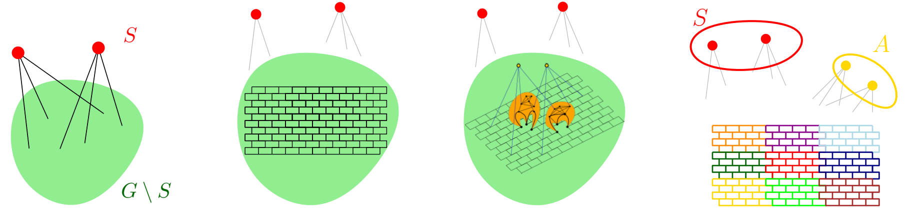
Iterative compression: given solution S of size $k+1$, search solution of size k.

If treewidth of $G \setminus S$ is "large enough" (as a polynomial function of k):

1. Find a "very very large" wall in $G \setminus S$.
2. Find a "very large" flat wall W of $G \setminus S$ with few apices A.
3. Find in W a packing of $\mathcal{O}_{\mathcal{F}}(k^4)$ disjoint "large" subwalls:
   - If every subwall has at least $|A|+1$ neighbors in $S \cup A$:
     Every solution intersects $S \cup A \to$ we can branch!

General scheme of the algorithm:

[whole slide shamelessly borrowed from Giannos Stamoulis]



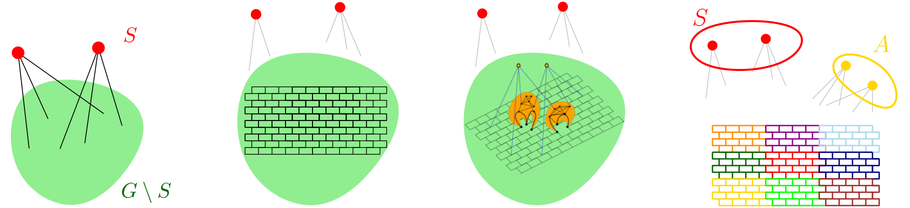
Iterative compression: given solution S of size $k+1$, search solution of size k.

If treewidth of $G \setminus S$ is "large enough" (as a polynomial function of k):

1. Find a "very very large" wall in $G \setminus S$.
2. Find a "very large" flat wall W of $G \setminus S$ with few apices A.
3. Find in W a packing of $\mathcal{O}_{\mathcal{F}}(k^4)$ disjoint "large" subwalls:
    - If every subwall has at least $|A|+1$ neighbors in $S \cup A$:
      Every solution intersects $S \cup A \rightarrow$ we can branch!
    - If one of these subwalls has at most $|A|$ neighbors in $S \cup A$:

General scheme of the algorithm:

[whole slide shamelessly borrowed from Giannos Stamoulis]



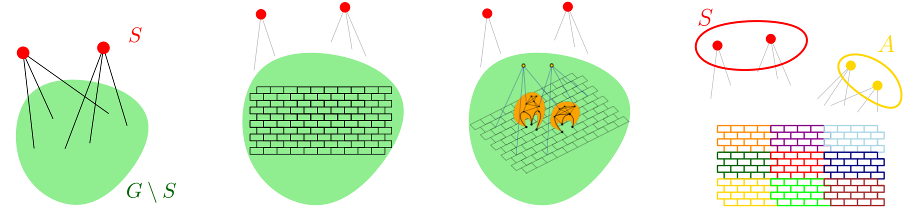
Iterative compression: given solution S of size $k+1$, search solution of size k.

If treewidth of $G \setminus S$ is "large enough" (as a polynomial function of k):

1. Find a "very very large" wall in $G \setminus S$.
2. Find a "very large" flat wall W of $G \setminus S$ with few apices A.
3. Find in W a packing of $\mathcal{O}_{\mathcal{F}}(k^4)$ disjoint "large" subwalls:
   - If every subwall has at least $|A|+1$ neighbors in $S \cup A$:
     Every solution intersects $S \cup A$ $\rightarrow$ we can branch!
   - If one of these subwalls has at most $|A|$ neighbors in $S \cup A$:
     Find an irrelevant vertex v inside this flat subwall.
     Update $G = G \setminus v$ and repeat.

General scheme of the algorithm:

[whole slide shamelessly borrowed from Giannos Stamoulis]



Iterative compression: given solution S of size $k+1$, search solution of size k.

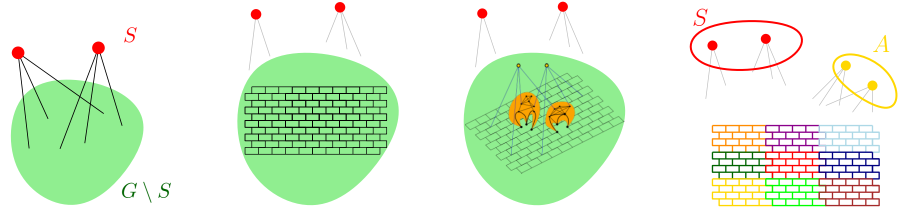
If treewidth of $G \setminus S$ is "large enough" (as a polynomial function of k):

1. Find a "very very large" wall in $G \setminus S$.
2. Find a "very large" flat wall W of $G \setminus S$ with few apices A.
3. Find in W a packing of $\mathcal{O}_{\mathcal{F}}(k^4)$ disjoint "large" subwalls:
    - If every subwall has at least $|A|+1$ neighbors in $S \cup A$:
      Every solution intersects $S \cup A \rightarrow$ we can branch!
    - If one of these subwalls has at most $|A|$ neighbors in $S \cup A$:
      Find an irrelevant vertex v inside this flat subwall.
      Update $G = G \setminus v$ and repeat.

Thus, $\mathrm{tw}(G \setminus S) = k^{\mathcal{O}_{\mathcal{F}}(1)}$:

General scheme of the algorithm:

[whole slide shamelessly borrowed from Giannos Stamoulis]



Iterative compression: given solution S of size $k+1$, search solution of size k.

If treewidth of $G \setminus S$ is "large enough" (as a polynomial function of k):

1. Find a "very very large" wall in $G \setminus S$.
2. Find a "very large" flat wall W of $G \setminus S$ with few apices A.
3. Find in W a packing of $\mathcal{O}_{\mathcal{F}}(k^4)$ disjoint "large" subwalls:
   - If every subwall has at least $|A|+1$ neighbors in $S \cup A$:
     Every solution intersects $S \cup A$ $\rightarrow$ we can branch!
   - If one of these subwalls has at most $|A|$ neighbors in $S \cup A$:
     Find an irrelevant vertex v inside this flat subwall.
     Update $G = G \setminus v$ and repeat.

Thus, $\text{tw}(G \setminus S) = k^{\mathcal{O}_{\mathcal{F}}(1)}$: our previous FPT algo gives $2^{k^{\mathcal{O}_{\mathcal{F}}(1)}} \cdot n^2$.

Main idea of our improved algorithm

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

For all $\mathcal{F}$, the $\mathcal{F}$-M-Deletion problem can be solved in time $2^{\text{poly}(k)} \cdot n^2$.

Improvement from n^3 to n^2: avoiding iterative compression.

Main idea of our improved algorithm

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

For all $\mathcal{F}$, the $\mathcal{F}$-M-Deletion problem can be solved in time $2^{\text{poly}(k)} \cdot n^2$.

Improvement from n^3 to n^2: avoiding iterative compression.

How to achieve it?

We are able to detect a vertex that must belong to every solution.

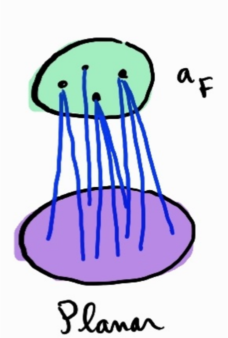
Approach inspired by [Marx, Schlotter. 2012] [S., Stamoulis, Thilikos. 2020]

skip

Finding a vertex belonging to every solution of size k

Let $\mathcal{F}$ be a finite collection of graphs.

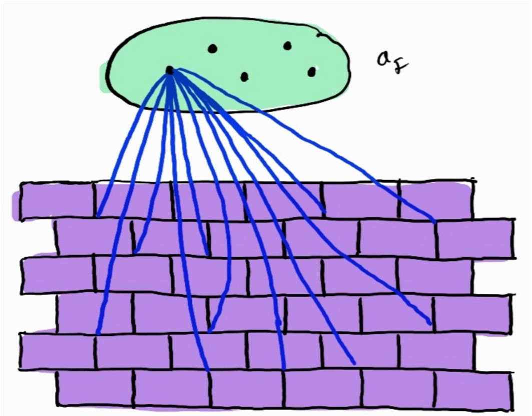
The **apex number** $a_{\mathcal{F}}$ is the smallest number of vertices that can be removed from a graph of $\mathcal{F}$ such that the remaining graph is planar.



[Figure by Laure Morelle]

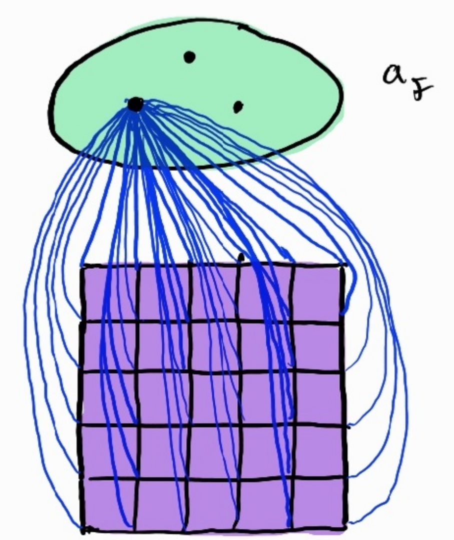
$a_{\mathcal{F}} = 1 \rightarrow$ apex graph

Finding a vertex belonging to every solution of size k



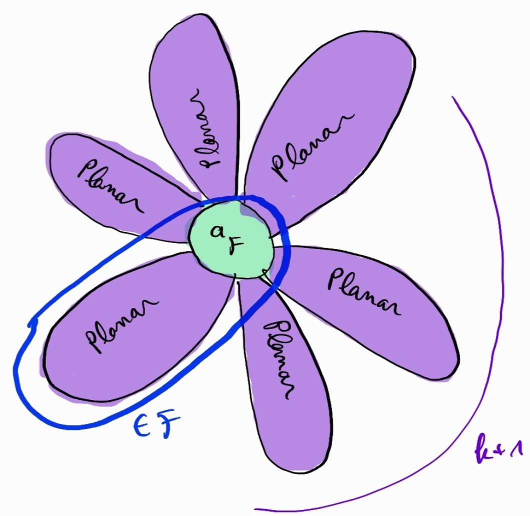
[Figure by Laure Morelle]

Finding a vertex belonging to every solution of size k



[Figure by Laure Morelle]

Finding a vertex belonging to every solution of size k



[Figure by Laure Morelle]

Strategy for solving $\mathcal{F}$-M-DELETION in time $2^{\text{poly}_{\mathcal{F}}(k)} \cdot n^2$:

Strategy for solving $\mathcal{F}$-M-Deletion in time $2^{\text{poly}_{\mathcal{F}}(k)} \cdot n^2$:

- If the treewidth of G is small (namely, $\text{tw} \leq \text{poly}_{\mathcal{F}}(k)$):

Strategy for solving $\mathcal{F}$-M-DELETION in time $2^{\text{poly}_{\mathcal{F}}(k)} \cdot n^2$:

- If the treewidth of G is small (namely, $\text{tw} \leq \text{poly}_{\mathcal{F}}(k)$):

  Dynamic programming using algorithm of [Baste, S., Thilikos. 2020]
  Solve in time $2^{\text{poly}_{\mathcal{F}}(\text{tw} \log \text{tw})} \cdot n$.

Strategy for solving $\mathcal{F}$-M-DELETION in time $2^{\text{poly}_{\mathcal{F}}(k)} \cdot n^2$:

- If the treewidth of G is small (namely, $\text{tw} \leq \text{poly}_{\mathcal{F}}(k)$):

  Dynamic programming using algorithm of [Baste, S., Thilikos. 2020]
  Solve in time $2^{\text{poly}_{\mathcal{F}}(\text{tw} \log \text{tw})} \cdot n$.

- If the treewidth of G is big, remove a vertex from G using one of the following approaches:

Strategy for solving $\mathcal{F}$-M-DELETION in time $2^{\text{poly}_{\mathcal{F}}(k)} \cdot n^2$:

- If the treewidth of G is small (namely, $\text{tw} \leq \text{poly}_{\mathcal{F}}(k)$):

  Dynamic programming using algorithm of [Baste, S., Thilikos. 2020]
  Solve in time $2^{\text{poly}_{\mathcal{F}}(\text{tw} \log \text{tw})} \cdot n$.

- If the treewidth of G is big, remove a vertex from G using one of the following approaches:

  - **Irrelevant vertex technique**: time $\mathcal{O}^*(n)$.
    Detect vertex v such that (G, k) and $(G \setminus \{v\}, k)$ are equivalent instances of $\mathcal{F}$-M-DELETION.

Strategy for solving $\mathcal{F}$-M-Deletion in time $2^{\text{poly}_{\mathcal{F}}(k)} \cdot n^2$:

- If the treewidth of G is small (namely, $\text{tw} \leq \text{poly}_{\mathcal{F}}(k)$):

  Dynamic programming using algorithm of [Baste, S., Thilikos. 2020]
  Solve in time $2^{\text{poly}_{\mathcal{F}}(\text{tw} \log \text{tw})} \cdot n$.

- If the treewidth of G is big, remove a vertex from G using one of the following approaches:

  - **Irrelevant vertex technique**: time $\mathcal{O}^*(n)$.
    Detect vertex v such that (G, k) and $(G \setminus \{v\}, k)$ are equivalent instances of $\mathcal{F}$-M-Deletion.

  - **Branching**: time $\mathcal{O}^*(n^2)$.
    Find set A of $a_{\mathcal{F}}$ vertices that intersects every k-apex set.
    "Guess" a vertex $v \in A$ in a k-apex set and solve $(G \setminus \{v\}, k-1)$.

Strategy for solving $\mathcal{F}$-M-DELETION in time $2^{\text{poly}_{\mathcal{F}}(k)} \cdot n^2$:

- If the treewidth of G is small (namely, $\text{tw} \leq \text{poly}_{\mathcal{F}}(k)$):

  Dynamic programming using algorithm of [Baste, S., Thilikos. 2020]
  Solve in time $2^{\text{poly}_{\mathcal{F}}(\text{tw} \log \text{tw})} \cdot n$.

- If the treewidth of G is big, remove a vertex from G using one of the following approaches:

  - **Irrelevant vertex technique**: time $\mathcal{O}^*(n)$.
    Detect vertex v such that (G, k) and $(G \setminus \{v\}, k)$ are equivalent instances of $\mathcal{F}$-M-DELETION.

  - **Branching**: time $\mathcal{O}^*(n^2)$.
    Find set A of $a_{\mathcal{F}}$ vertices that intersects every k-apex set.
    "Guess" a vertex $v \in A$ in a k-apex set and solve $(G \setminus \{v\}, k-1)$.

    (Branching tree is of size $a_{\mathcal{F}}^k$, so we do *not* get an extra factor n).

skip

Next section is...

Motivation: distance from triviality

Distance from triviality: [Guo, Hüffner, Niedermeier. 2004]

Concept to express the closeness of a graph G to a "trivial" graph class $\mathcal{H}$.

Motivation: distance from triviality

Distance from triviality: [Guo, Hüffner, Niedermeier. 2004]

Concept to express the closeness of a graph G to a "trivial" graph class $\mathcal{H}$.

Motivation: Solve problems parameterized by the distance to $\mathcal{H}$.

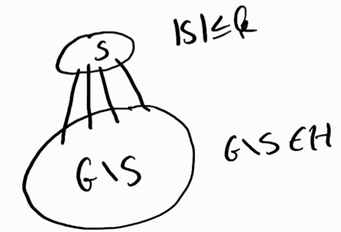
Motivation: distance from triviality

Distance from triviality: [Guo, Hüffner, Niedermeier. 2004]

Concept to express the closeness of a graph G to a "trivial" graph class $\mathcal{H}$.

Motivation: Solve problems parameterized by the distance to $\mathcal{H}$.

→ Vertex Deletion to $\mathcal{H}$



[Figure by Laure Morelle]

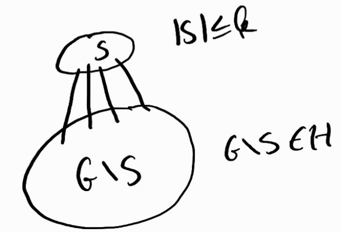
Motivation: distance from triviality

Distance from triviality: [Guo, Hüffner, Niedermeier. 2004]

Concept to express the closeness of a graph G to a "trivial" graph class $\mathcal{H}$.

Motivation: Solve problems parameterized by the distance to $\mathcal{H}$.

→ Vertex Deletion to $\mathcal{H}$



[Figure by Laure Morelle]

→ Elimination Distance to $\mathcal{H}$

Notion recently introduced by [Bulian, Dawar. 2016]

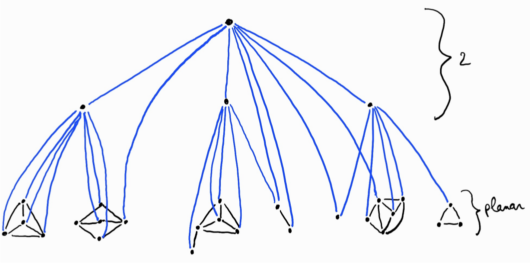
The **elimination distance** of a graph G to a graph class $\mathcal{H}$ is:

$$\mathrm{ed}_{\mathcal{H}}(G) = \begin{cases} 0 & \text{if } G \in \mathcal{H}, \\ 1 + \min\{\mathrm{ed}_{\mathcal{H}}(G \setminus \{v\}) \mid v \in V(G)\} & \text{if } G \text{ is connected}, \\ \max\{\mathrm{ed}_{\mathcal{H}}(H) \mid H \text{ is a connected component of } G\} & \text{otherwise.} \end{cases}$$

Notion recently introduced by [Bulian, Dawar. 2016]

The **elimination distance** of a graph G to a graph class $\mathcal{H}$ is:

$$\mathrm{ed}_{\mathcal{H}}(G) = \begin{cases} 0 & \text{if } G \in \mathcal{H}, \\ 1 + \min\{\mathrm{ed}_{\mathcal{H}}(G \setminus \{v\}) \mid v \in V(G)\} & \text{if } G \text{ is connected}, \\ \max\{\mathrm{ed}_{\mathcal{H}}(H) \mid H \text{ is a connected component of } G\} & \text{otherwise.} \end{cases}$$

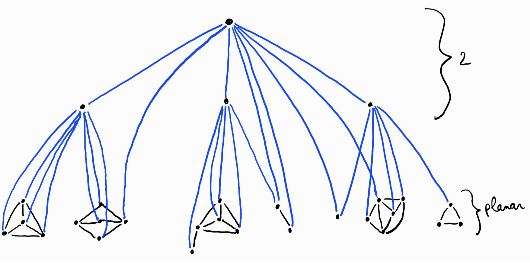


[Figure by Laure Morelle]

Notion recently introduced by [Bulian, Dawar. 2016]

The **elimination distance** of a graph G to a graph class $\mathcal{H}$ is:

$$\mathrm{ed}_{\mathcal{H}}(G) = \begin{cases} 0 & \text{if } G \in \mathcal{H}, \\ 1 + \min\{\mathrm{ed}_{\mathcal{H}}(G \setminus \{v\}) \mid v \in V(G)\} & \text{if } G \text{ is connected}, \\ \max\{\mathrm{ed}_{\mathcal{H}}(H) \mid H \text{ is a connected component of } G\} & \text{otherwise.} \end{cases}$$



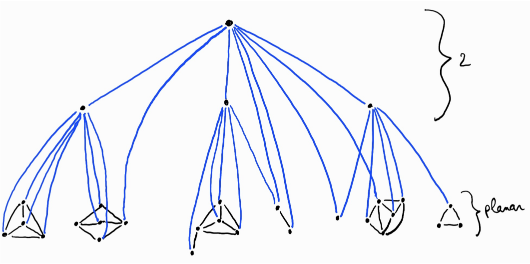
[Figure by Laure Morelle]

k-elimination set: set of removed vertices such that $\mathrm{ed}_{\mathcal{H}}(G) \leq k$.

Notion recently introduced by [Bulian, Dawar. 2016]

The **elimination distance** of a graph G to a graph class $\mathcal{H}$ is:

$$\mathrm{ed}_{\mathcal{H}}(G) = \begin{cases} 0 & \text{if } G \in \mathcal{H}, \\ 1 + \min\{\mathrm{ed}_{\mathcal{H}}(G \setminus \{v\}) \mid v \in V(G)\} & \text{if } G \text{ is connected}, \\ \max\{\mathrm{ed}_{\mathcal{H}}(H) \mid H \text{ is a connected component of } G\} & \text{otherwise.} \end{cases}$$



[Figure by Laure Morelle]

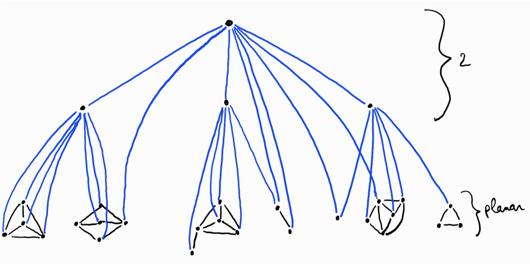
k-elimination set: set of removed vertices such that $\mathrm{ed}_{\mathcal{H}}(G) \leq k$.

Remark: the size of a k-elimination set is not necessarily a function of k!

Notion recently introduced by [Bulian, Dawar. 2016]

The **elimination distance** of a graph G to a graph class $\mathcal{H}$ is:

$$\mathrm{ed}_{\mathcal{H}}(G) = \begin{cases} 0 & \text{if } G \in \mathcal{H}, \\ 1 + \min\{\mathrm{ed}_{\mathcal{H}}(G \setminus \{v\}) \mid v \in V(G)\} & \text{if } G \text{ is connected}, \\ \max\{\mathrm{ed}_{\mathcal{H}}(H) \mid H \text{ is a connected component of } G\} & \text{otherwise.} \end{cases}$$



[Figure by Laure Morelle]

k-elimination set: set of removed vertices such that $\mathrm{ed}_{\mathcal{H}}(G) \leq k$.

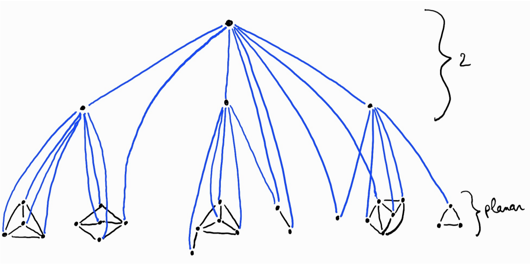
Remark: the size of a k-elimination set is not necessarily a function of k!

$\rightarrow \mathcal{H} = \{\emptyset\}$: treedepth

Notion recently introduced by [Bulian, Dawar. 2016]

The **elimination distance** of a graph G to a graph class $\mathcal{H}$ is:

$$\mathrm{ed}_{\mathcal{H}}(G) = \begin{cases} 0 & \text{if } G \in \mathcal{H}, \\ 1 + \min\{\mathrm{ed}_{\mathcal{H}}(G \setminus \{v\}) \mid v \in V(G)\} & \text{if } G \text{ is connected}, \\ \max\{\mathrm{ed}_{\mathcal{H}}(H) \mid H \text{ is a connected component of } G\} & \text{otherwise.} \end{cases}$$



[Figure by Laure Morelle]

k-elimination set: set of removed vertices such that $\mathrm{ed}_{\mathcal{H}}(G) \leq k$.

Remark: the size of a k-elimination set is not necessarily a function of k!

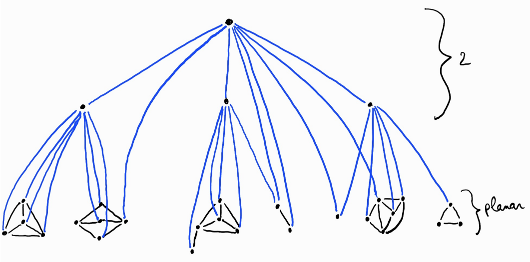
$\rightarrow \mathcal{H} = \{\emptyset\}$: treedepth

Stronger parameter than vertex deletion: $\mathrm{ed}_{\mathcal{H}}(G) \leq \mathrm{VertexDeletion}_{\mathcal{H}}(G)$

Notion recently introduced by [Bulian, Dawar. 2016]

The **elimination distance** of a graph G to a graph class $\mathcal{H}$ is:

$$\mathrm{ed}_{\mathcal{H}}(G) = \begin{cases} 0 & \text{if } G \in \mathcal{H}, \\ 1 + \min\{\mathrm{ed}_{\mathcal{H}}(G \setminus \{v\}) \mid v \in V(G)\} & \text{if } G \text{ is connected}, \\ \max\{\mathrm{ed}_{\mathcal{H}}(H) \mid H \text{ is a connected component of } G\} & \text{otherwise.} \end{cases}$$



[Figure by Laure Morelle]

ELIMINATION DISTANCE TO $\mathcal{H}$

Input: A graph G and a $k \in \mathbb{N}$.
Question: Is $\mathrm{ed}_{\mathcal{H}}(G) \leq k$?

What is known about ELIMINATION DISTANCE TO $\mathcal{H}$?

What is known about ELIMINATION DISTANCE TO $\mathcal{H}$?

Let $\mathcal{E}_k(\mathcal{H}) = \{G \mid \mathrm{ed}_{\mathcal{H}}(G) \leq k\}$.

What is known about ELIMINATION DISTANCE TO $\mathcal{H}$?

Let $\mathcal{E}_k(\mathcal{H}) = \{G \mid \mathrm{ed}_{\mathcal{H}}(G) \leq k\}$.

(G, k) yes-instance of ELIMINATION DISTANCE TO $\mathcal{H} \Leftrightarrow G \in \mathcal{E}_k(\mathcal{H})$.

What is known about Elimination Distance to $\mathcal{H}$?

Let $\mathcal{E}_k(\mathcal{H}) = \{G \mid \text{ed}_{\mathcal{H}}(G) \leq k\}$.

(G, k) yes-instance of Elimination Distance to $\mathcal{H}$ $\Leftrightarrow$ $G \in \mathcal{E}_k(\mathcal{H})$.

$\mathcal{H}$ minor-closed

What is known about ELIMINATION DISTANCE TO $\mathcal{H}$?

Let $\mathcal{E}_k(\mathcal{H}) = \{G \mid \mathrm{ed}_{\mathcal{H}}(G) \leq k\}$.

(G, k) yes-instance of ELIMINATION DISTANCE TO $\mathcal{H} \Leftrightarrow G \in \mathcal{E}_k(\mathcal{H})$.

$\mathcal{H}$ minor-closed $\Rightarrow \mathcal{E}_k(\mathcal{H})$ minor-closed

What is known about Elimination Distance to $\mathcal{H}$?

Let $\mathcal{E}_k(\mathcal{H}) = \{G \mid \text{ed}_{\mathcal{H}}(G) \leq k\}$.

(G, k) yes-instance of Elimination Distance to $\mathcal{H} \Leftrightarrow G \in \mathcal{E}_k(\mathcal{H})$.

$\mathcal{H}$ minor-closed $\Rightarrow \mathcal{E}_k(\mathcal{H})$ minor-closed $\Rightarrow$ non-constructive FPT-algo.

What is known about ELIMINATION DISTANCE TO $\mathcal{H}$?

Let $\mathcal{E}_k(\mathcal{H}) = \{G \mid \text{ed}_{\mathcal{H}}(G) \leq k\}$.

(G, k) yes-instance of ELIMINATION DISTANCE TO $\mathcal{H} \Leftrightarrow G \in \mathcal{E}_k(\mathcal{H})$.

$\mathcal{H}$ minor-closed $\Rightarrow \mathcal{E}_k(\mathcal{H})$ minor-closed $\Rightarrow$ non-constructive FPT-algo.

If we are given $\mathcal{F} = \text{Obs}(\mathcal{H})$, it is possible to construct $\text{Obs}(\mathcal{E}_k(\mathcal{H}))$.

[Bulian, Dawar. 2017]

What is known about ELIMINATION DISTANCE TO $\mathcal{H}$?

Let $\mathcal{E}_k(\mathcal{H}) = \{G \mid \mathrm{ed}_{\mathcal{H}}(G) \leq k\}$.

(G, k) yes-instance of ELIMINATION DISTANCE TO $\mathcal{H}$ $\Leftrightarrow$ $G \in \mathcal{E}_k(\mathcal{H})$.

$\mathcal{H}$ minor-closed $\Rightarrow$ $\mathcal{E}_k(\mathcal{H})$ minor-closed $\Rightarrow$ non-constructive FPT-algo.

If we are given $\mathcal{F} = \mathrm{Obs}(\mathcal{H})$, it is possible to construct $\mathrm{Obs}(\mathcal{E}_k(\mathcal{H}))$.
[Bulian, Dawar. 2017]

$\Rightarrow$ constructive FPT-algorithm: $f(k) \cdot n^2$

What is known about ELIMINATION DISTANCE TO $\mathcal{H}$?

Let $\mathcal{E}_k(\mathcal{H}) = \{G \mid \text{ed}_{\mathcal{H}}(G) \leq k\}$.

(G, k) yes-instance of ELIMINATION DISTANCE TO $\mathcal{H} \Leftrightarrow G \in \mathcal{E}_k(\mathcal{H})$.

$\mathcal{H}$ minor-closed $\Rightarrow \mathcal{E}_k(\mathcal{H})$ minor-closed $\Rightarrow$ non-constructive FPT-algo.

If we are given $\mathcal{F} = \text{Obs}(\mathcal{H})$, it is possible to construct $\text{Obs}(\mathcal{E}_k(\mathcal{H}))$.

[Bulian, Dawar. 2017]

$\Rightarrow$ constructive FPT-algorithm: $f(k) \cdot n^2$

Can we provide an explicit function $f(k)$?

Taking the treewidth as the parameter

If $\mathcal{H} = \{\emptyset\}$ (treedepth): [Reidl, Rossmanith, Sanchez Villaamil, Sikdar. 2014]

Dynamic programming algorithm parameterized by treewidth in $2^{\mathcal{O}(k \cdot \mathsf{tw})} \cdot n$.

Taking the treewidth as the parameter

If $\mathcal{H} = \{\emptyset\}$ (treedepth): [Reidl, Rossmanith, Sanchez Villaamil, Sikdar. 2014]

Dynamic programming algorithm parameterized by treewidth in $2^{\mathcal{O}(k \cdot \mathrm{tw})} \cdot n$.

Since $\mathrm{tw}(G) \leq \mathrm{td}(G) \leq \mathrm{tw}(G) \log n$

Taking the treewidth as the parameter

If $\mathcal{H} = \{\emptyset\}$ (treedepth): [Reidl, Rossmanith, Sanchez Villaamil, Sikdar. 2014]

Dynamic programming algorithm parameterized by treewidth in $2^{\mathcal{O}(k \cdot \mathrm{tw})} \cdot n$.

Since $\mathrm{tw}(G) \leq \mathrm{td}(G) \leq \mathrm{tw}(G) \log n \rightarrow$ time $n^{\mathcal{O}(\mathrm{tw}^2)}$

Taking the treewidth as the parameter

If $\mathcal{H} = \{\emptyset\}$ (treedepth): [Reidl, Rossmanith, Sanchez Villaamil, Sikdar. 2014]

Dynamic programming algorithm parameterized by treewidth in $2^{\mathcal{O}(k \cdot \mathrm{tw})} \cdot n$.

Since $\mathrm{tw}(G) \leq \mathrm{td}(G) \leq \mathrm{tw}(G) \log n \rightarrow$ time $n^{\mathcal{O}(\mathrm{tw}^2)}$ and $2^{\mathcal{O}(k^2)} \cdot n$.

Taking the treewidth as the parameter

If $\mathcal{H} = \{\emptyset\}$ (treedepth): [Reidl, Rossmanith, Sanchez Villaamil, Sikdar. 2014]

Dynamic programming algorithm parameterized by treewidth in $2^{\mathcal{O}(k \cdot \text{tw})} \cdot n$.

Since $\text{tw}(G) \leq \text{td}(G) \leq \text{tw}(G) \log n \rightarrow$ time $n^{\mathcal{O}(\text{tw}^2)}$ and $2^{\mathcal{O}(k^2)} \cdot n$.

Open problem: computing td parameterized by tw is FPT?

Taking the treewidth as the parameter

If $\mathcal{H} = \{\emptyset\}$ (treedepth): [Reidl, Rossmanith, Sanchez Villaamil, Sikdar. 2014]

Dynamic programming algorithm parameterized by treewidth in $2^{\mathcal{O}(k \cdot \mathsf{tw})} \cdot n$.

Since $\mathsf{tw}(G) \leq \mathsf{td}(G) \leq \mathsf{tw}(G) \log n \rightarrow$ time $n^{\mathcal{O}(\mathsf{tw}^2)}$ and $2^{\mathcal{O}(k^2)} \cdot n$.

Open problem: computing td parameterized by tw is FPT?

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

Given a graph G on n vertices and with treewidth at most tw, and $k \in \mathbb{N}$, there is an algorithm that solves ELIMINATION DISTANCE TO $\mathcal{H}$ *for the instance (G, k) in time $2^{\mathcal{O}_{\mathcal{H}}(k \cdot \mathsf{tw} + \mathsf{tw} \log \mathsf{tw})} \cdot n$.*

Taking the treewidth as the parameter

If $\mathcal{H} = \{\emptyset\}$ (treedepth): [Reidl, Rossmanith, Sanchez Villaamil, Sikdar. 2014]

Dynamic programming algorithm parameterized by treewidth in $2^{\mathcal{O}(k \cdot \mathsf{tw})} \cdot n$.

Since $\mathsf{tw}(G) \leq \mathsf{td}(G) \leq \mathsf{tw}(G) \log n \rightarrow$ time $n^{\mathcal{O}(\mathsf{tw}^2)}$ and $2^{\mathcal{O}(k^2)} \cdot n$.

Open problem: computing td parameterized by tw is FPT?

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

Given a graph G on n vertices and with treewidth at most tw, and $k \in \mathbb{N}$, there is an algorithm that solves Elimination Distance to $\mathcal{H}$ *for the instance (G, k) in time $2^{\mathcal{O}_{\mathcal{H}}(k \cdot \mathsf{tw} + \mathsf{tw} \log \mathsf{tw})} \cdot n$.*

$\rightarrow$ algorithm in time $n^{\mathcal{O}_{\mathcal{H}}(\mathsf{tw}^2)}$ for Elimination Distance to $\mathcal{H}$.

Our results for ELIMINATION DISTANCE TO $\mathcal{H}$

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

Given a graph G on n vertices and $k \in \mathbb{N}$, there is an algorithm that solves ELIMINATION DISTANCE TO $\mathcal{H}$ *for the instance (G, k) in time*

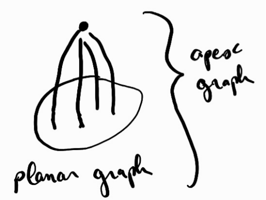
- $2^{2^{2^{\mathrm{poly}_{\mathcal{H}}(k)}}} \cdot n^2$ *for a general minor-closed class $\mathcal{H}$,*

Our results for ELIMINATION DISTANCE TO $\mathcal{H}$

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

Given a graph G on n vertices and $k \in \mathbb{N}$, there is an algorithm that solves ELIMINATION DISTANCE TO $\mathcal{H}$ *for the instance (G, k) in time*

- $2^{2^{2^{\text{poly}_{\mathcal{H}}(k)}}} \cdot n^2$ *for a general minor-closed class $\mathcal{H}$,*
- $2^{2^{\text{poly}_{\mathcal{H}}(k)}} \cdot n^2$ *if $\text{Obs}(\mathcal{H})$ contains an apex graph.*



[Figure by Laure Morelle]

Our results for ELIMINATION DISTANCE TO $\mathcal{H}$

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

Given a graph G on n vertices and $k \in \mathbb{N}$, there is an algorithm that solves ELIMINATION DISTANCE TO $\mathcal{H}$ *for the instance (G, k) in time*

- $2^{2^{2^{\text{poly}_{\mathcal{H}}(k)}}} \cdot n^2$ *for a general minor-closed class $\mathcal{H}$,*
- $2^{2^{\text{poly}_{\mathcal{H}}(k)}} \cdot n^2$ *if $\text{Obs}(\mathcal{H})$ contains an apex graph.*

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

If $\text{Obs}(\mathcal{H})$ contains an apex graph, given a graph G on n vertices and $k \in \mathbb{N}$, there is an algorithm that solves ELIMINATION DISTANCE TO $\mathcal{H}$ *for the instance (G, k) in time $2^{\text{poly}_{\mathcal{H}}(k)} \cdot n^3$.*

Our results for Elimination Distance to $\mathcal{H}$

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

Given a graph G on n vertices and $k \in \mathbb{N}$, there is an algorithm that solves Elimination Distance to $\mathcal{H}$ *for the instance (G, k) in time*

- $2^{2^{2^{\mathrm{poly}_{\mathcal{H}}(k)}}} \cdot n^2$ *for a general minor-closed class $\mathcal{H}$,*
- $2^{2^{\mathrm{poly}_{\mathcal{H}}(k)}} \cdot n^2$ *if $\mathrm{Obs}(\mathcal{H})$ contains an apex graph.*

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

If $\mathrm{Obs}(\mathcal{H})$ contains an apex graph, given a graph G on n vertices and $k \in \mathbb{N}$, there is an algorithm that solves Elimination Distance to $\mathcal{H}$ *for the instance (G, k) in time $2^{\mathrm{poly}_{\mathcal{H}}(k)} \cdot n^3$.*

Main challenge compared to Vertex Deletion to $\mathcal{H}$:

The size of a k-elimination set may be unbounded, so we cannot branch!

Our results for Elimination Distance to $\mathcal{H}$

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

Given a graph G on n vertices and $k \in \mathbb{N}$, there is an algorithm that solves Elimination Distance to $\mathcal{H}$ *for the instance (G, k) in time*

- $2^{2^{2^{\text{poly}_{\mathcal{H}}(k)}}} \cdot n^2$ *for a general minor-closed class $\mathcal{H}$,*
- $2^{2^{\text{poly}_{\mathcal{H}}(k)}} \cdot n^2$ *if $\text{Obs}(\mathcal{H})$ contains an apex graph.*

Theorem (Morelle, S., Stamoulis, Thilikos. 2022)

If $\text{Obs}(\mathcal{H})$ contains an apex graph, given a graph G on n vertices and $k \in \mathbb{N}$, there is an algorithm that solves Elimination Distance to $\mathcal{H}$ *for the instance (G, k) in time $2^{\text{poly}_{\mathcal{H}}(k)} \cdot n^3$.*

Main challenge compared to Vertex Deletion to $\mathcal{H}$:

The size of a k-elimination set may be unbounded, so we cannot branch!
We always have to find an irrelevant vertex: larger treewidth bounds.

Next section is...

What's next about $\mathcal{F}$-M-Vertex-Deletion?

What's next about $\mathcal{F}$-M-Vertex-Deletion?

With parameter tw | Classify the asymptotic complexity of $\mathcal{F}$-M-Deletion for every family $\mathcal{F}$?

What's next about $\mathcal{F}$-M-Vertex-Deletion?

With parameter tw: Classify the asymptotic complexity of $\mathcal{F}$-M-Deletion for every family $\mathcal{F}$?

- We obtained a tight dichotomy when $|\mathcal{F}| = 1$ (connected).

What's next about $\mathcal{F}$-M-Vertex-Deletion?

With parameter tw Classify the asymptotic complexity of $\mathcal{F}$-M-Deletion for every family $\mathcal{F}$?

- We obtained a tight dichotomy when $|\mathcal{F}| = 1$ (connected).
- Missing: When $|\mathcal{F}| \geq 2$ (connected): $2^{\Theta(\text{tw})}$ or $2^{\Theta(\text{tw} \cdot \log \text{tw})}$?

What's next about $\mathcal{F}$-M-Vertex-Deletion?

With parameter tw — Classify the asymptotic complexity of $\mathcal{F}$-M-Deletion for every family $\mathcal{F}$?

- We obtained a tight dichotomy when $|\mathcal{F}| = 1$ (connected).
- Missing: When $|\mathcal{F}| \geq 2$ (connected): $2^{\Theta(\mathrm{tw})}$ or $2^{\Theta(\mathrm{tw} \cdot \log \mathrm{tw})}$?

We can also consider the topological minor version:

What's next about $\mathcal{F}$-M-Vertex-Deletion?

With parameter tw Classify the asymptotic complexity of $\mathcal{F}$-M-Deletion for every family $\mathcal{F}$?

- We obtained a tight dichotomy when $|\mathcal{F}| = 1$ (connected).
- Missing: When $|\mathcal{F}| \geq 2$ (connected): $2^{\Theta(\text{tw})}$ or $2^{\Theta(\text{tw} \cdot \log \text{tw})}$?

We can also consider the topological minor version:

- Dichotomy for $\{H\}$-TM-Deletion when H connected (+planar)?

What's next about $\mathcal{F}$-M-Vertex-Deletion?

With parameter tw: Classify the asymptotic complexity of $\mathcal{F}$-M-Deletion for every family $\mathcal{F}$?

- We obtained a tight dichotomy when $|\mathcal{F}| = 1$ (connected).
- Missing: When $|\mathcal{F}| \geq 2$ (connected): $2^{\Theta(\mathsf{tw})}$ or $2^{\Theta(\mathsf{tw} \cdot \log \mathsf{tw})}$?

We can also consider the topological minor version:

- Dichotomy for $\{H\}$-TM-Deletion when H connected (+planar)?
- We do not know if there exists some $\mathcal{F}$ such that $\mathcal{F}$-TM-Deletion cannot be solved in time $2^{o(\mathsf{tw}^2)} \cdot n^{\mathcal{O}(1)}$ under the ETH.

What's next about $\mathcal{F}$-M-VERTEX-DELETION?

With parameter tw — Classify the asymptotic complexity of $\mathcal{F}$-M-DELETION for every family $\mathcal{F}$?

- We obtained a tight dichotomy when $|\mathcal{F}| = 1$ (connected).
- Missing: When $|\mathcal{F}| \geq 2$ (connected): $2^{\Theta(\mathsf{tw})}$ or $2^{\Theta(\mathsf{tw} \cdot \log \mathsf{tw})}$?

We can also consider the topological minor version:

- Dichotomy for $\{H\}$-TM-DELETION when H connected (+planar)?
- We do not know if there exists some $\mathcal{F}$ such that $\mathcal{F}$-TM-DELETION cannot be solved in time $2^{o(\mathsf{tw}^2)} \cdot n^{\mathcal{O}(1)}$ under the ETH.

With parameter k — We presented an algorithm in time $2^{k^{\mathcal{O}_{\mathcal{F}}(1)}} \cdot n^2$.

What's next about $\mathcal{F}$-M-Vertex-Deletion?

With parameter tw — Classify the asymptotic complexity of $\mathcal{F}$-M-Deletion for every family $\mathcal{F}$?

- We obtained a tight dichotomy when $|\mathcal{F}| = 1$ (connected).
- Missing: When $|\mathcal{F}| \geq 2$ (connected): $2^{\Theta(\text{tw})}$ or $2^{\Theta(\text{tw} \cdot \log \text{tw})}$?

We can also consider the topological minor version:

- Dichotomy for $\{H\}$-TM-Deletion when H connected (+planar)?
- We do not know if there exists some $\mathcal{F}$ such that $\mathcal{F}$-TM-Deletion cannot be solved in time $2^{o(\text{tw}^2)} \cdot n^{\mathcal{O}(1)}$ under the ETH.

With parameter k — We presented an algorithm in time $2^{k^{\mathcal{O}_{\mathcal{F}}(1)}} \cdot n^2$.
Is $2^{\mathcal{O}_{\mathcal{F}}(k^c)} \cdot n^{\mathcal{O}(1)}$ possible for some constant c?

What's next about $\mathcal{F}$-M-Vertex-Deletion?

With parameter tw — Classify the asymptotic complexity of $\mathcal{F}$-M-Deletion for every family $\mathcal{F}$?

- We obtained a tight dichotomy when $|\mathcal{F}| = 1$ (connected).
- Missing: When $|\mathcal{F}| \geq 2$ (connected): $2^{\Theta(\text{tw})}$ or $2^{\Theta(\text{tw} \cdot \log \text{tw})}$?

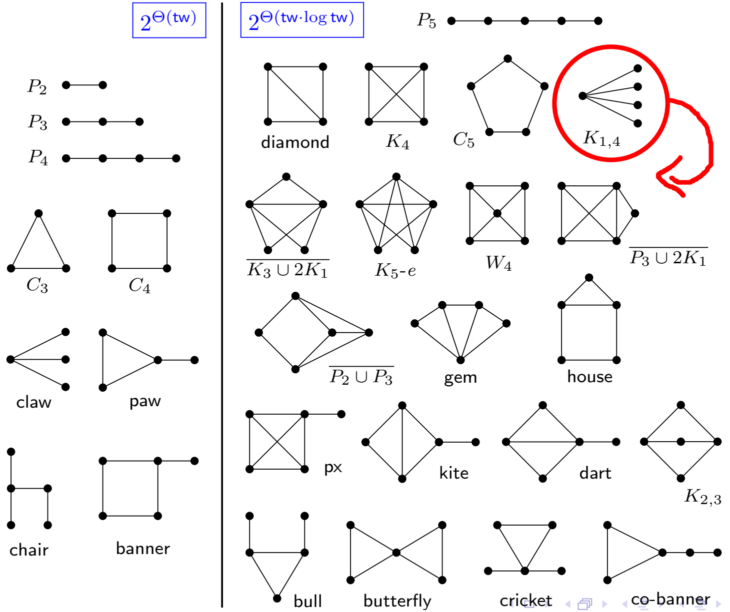
We can also consider the topological minor version:

- Dichotomy for $\{H\}$-TM-Deletion when H connected (+planar)?
- We do not know if there exists some $\mathcal{F}$ such that $\mathcal{F}$-TM-Deletion cannot be solved in time $2^{o(\text{tw}^2)} \cdot n^{\mathcal{O}(1)}$ under the ETH.

With parameter k — We presented an algorithm in time $2^{k^{\mathcal{O}_{\mathcal{F}}(1)}} \cdot n^2$.
Is $2^{\mathcal{O}_{\mathcal{F}}(k^c)} \cdot n^{\mathcal{O}(1)}$ possible for some constant c?
Is the price of homogeneity unavoidable?

skip

For topological minors, there is (at least) one change



Gràcies!