Single-exponential algorithms and linear kernels via protrusion decompositions

Eun Jung Kim[1] Christophe Paul[2] Ignasi Sau[2]

Alexander Langer[3] Felix Reidl[3] Peter Rossmanith[3] Somnath Sikdar[3]

arXiv/1207.0835, 2013

[1] CNRS, LAMSADE, Paris (France)

[2] CNRS, LIRMM, Montpellier (France)

[3] Department of Computer Science, RWTH Aachen University (Germany)

Outline of the talk

1. Preliminaries
2. Protrusion decompositions
   - Definitions
   - A simple algorithm to compute them
3. Single-exponential algorithm for PLANAR-$\mathcal{F}$-DELETION
   - Motivation and our result
   - Sketch of proof
   - Further research
4. Linear kernels on graphs without topological minors
   - Motivation and our result
   - Idea of proof
   - Further research

Next section is...

1. Preliminaries

2. Protrusion decompositions
   - Definitions
   - A simple algorithm to compute them

3. Single-exponential algorithm for PLANAR-$\mathcal{F}$-DELETION
   - Motivation and our result
   - Sketch of proof
   - Further research

4. Linear kernels on graphs without topological minors
   - Motivation and our result
   - Idea of proof
   - Further research

Some words on parameterized complexity

- $\boxed{\text{Idea}}$ given an NP-hard problem with input size n, fix one parameter k of the input to see whether the problem gets more "tractable".

  Example: the size of a VERTEX COVER.

Some words on parameterized complexity

- $\boxed{\text{Idea}}$ given an NP-hard problem with input size n, fix one parameter k of the input to see whether the problem gets more "tractable".

  Example: the size of a Vertex Cover.

- Given a (NP-hard) problem with input of size n and a parameter k, a fixed-parameter tractable (FPT) algorithm runs in time

$$f(k) \cdot n^{O(1)}, \quad \text{for some function } f.$$

  Examples: k-Vertex Cover, k-Longest Path.

Some words on parameterized complexity

- $\boxed{\text{Idea}}$ given an NP-hard problem with input size n, fix one parameter k of the input to see whether the problem gets more "tractable".

  Example: the size of a Vertex Cover.

- Given a (NP-hard) problem with input of size n and a parameter k, a fixed-parameter tractable (FPT) algorithm runs in time

$$f(k) \cdot n^{O(1)}, \quad \text{for some function } f.$$

  Examples: k-Vertex Cover, k-Longest Path.

- A single-exponential parameterized algorithm is an FPT algo s.t.

$$f(k) = 2^{O(k)}.$$

The decomposition paradigm — “Divide et impera”

Many hard algorithmic graph problems become easier if one is able to find a suitable decomposition of the input graph.

The decomposition paradigm — "Divide et impera"

Many hard algorithmic graph problems become easier if one is able to find a suitable decomposition of the input graph.

Some famous examples:

- PTAS and exact subexponential algorithms based on finding separators of size $O(\sqrt{n})$ on planar graphs. [Baker's approach]

The decomposition paradigm — "Divide et impera"

Many hard algorithmic graph problems become easier if one is able to find a suitable decomposition of the input graph.

Some famous examples:

- PTAS and exact subexponential algorithms based on finding separators of size $O(\sqrt{n})$ on planar graphs. [Baker's approach]
- Linear-time algorithms for problems expressible in MSOL on graphs of bounded treewidth. [Coucelle's theorem]

The decomposition paradigm — "Divide et impera"

Many hard algorithmic graph problems become easier if one is able to find a suitable decomposition of the input graph.

Some famous examples:

- PTAS and exact subexponential algorithms based on finding separators of size $O(\sqrt{n})$ on planar graphs. [Baker's approach]
- Linear-time algorithms for problems expressible in MSOL on graphs of bounded treewidth. [Coucelle's theorem]
- FPT algorithms based on the structural decomposition result of H-minor-free graphs. [Graph Minors theory by Robertson and Seymour]

The decomposition paradigm — "Divide et impera"

Many hard algorithmic graph problems become easier if one is able to find a suitable decomposition of the input graph.

Some famous examples:

- PTAS and exact subexponential algorithms based on finding separators of size $O(\sqrt{n})$ on planar graphs. [Baker's approach]
- Linear-time algorithms for problems expressible in MSOL on graphs of bounded treewidth. [Coucelle's theorem]
- FPT algorithms based on the structural decomposition result of H-minor-free graphs. [Graph Minors theory by Robertson and Seymour]
- Linear-time algorithms based on modular decompositions.

Next section is...

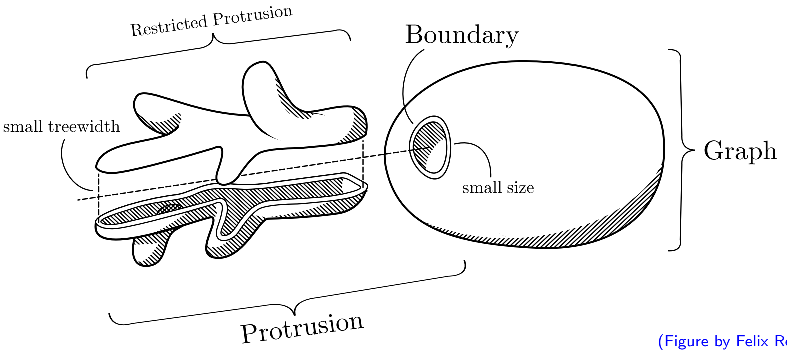
Next subsection is...

Protrusions

[Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh, Thilikos '09]

- Given a graph G, a set $W \subseteq V(G)$ is a t-protrusion of G if

$$\boxed{|\partial_G(W)| \leqslant t \quad \text{and} \quad \mathrm{tw}(G[W]) \leqslant t}$$



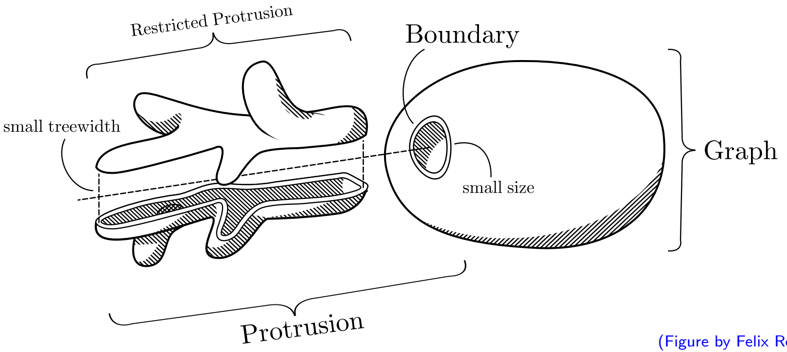
(Figure by Felix Reidl)

Protrusions

[Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh, Thilikos '09]

- Given a graph G, a set $W \subseteq V(G)$ is a t-protrusion of G if

$$\boxed{|\partial_G(W)| \leqslant t \quad \text{and} \quad \text{tw}(G[W]) \leqslant t}$$



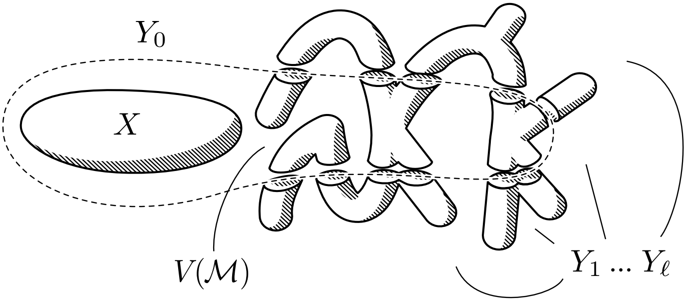
(Figure by Felix Reidl)

- The vertex set $W' = W \setminus \partial_G(W)$ is the restricted protrusion of W.
- We call $\partial_G(W)$ the boundary and $|W|$ the size of W.

Protrusion decompositions

An (α, t)-protrusion decomposition of a graph G is a partition $\mathcal{P} = Y_0 \uplus Y_1 \uplus \cdots \uplus Y_\ell$ of $V(G)$ such that:

- for every $1 \leqslant i \leqslant \ell$, $N(Y_i) \subseteq Y_0$;
- for every $1 \leqslant i \leqslant \ell$, $Y_i \cup N_{Y_0}(Y_i)$ is a t-protrusion of G;
- $\max\{\ell, |Y_0|\} \leqslant \alpha$.



The set Y_0 is called the separating part of $\mathcal{P}$.

(Figure by Felix Reidl)

Next subsection is...

Main (informal) ideas of our algorithm

- Protrusion decompositions have already been used in the literature.

[Bodlaender, Fomin, Lokshtanov, Saurabh, Thilikos '09-12]

Main (informal) ideas of our algorithm

- Here we present a new algorithm to compute protrusion decompositions for graphs G that come equipped with a set

$$\boxed{X \subseteq V(G) \text{ s.t. } \mathrm{tw}(G - X) \leqslant t}$$

for some constant $t > 0$.

The set X is called a t-treewidth-modulator.

Main (informal) ideas of our algorithm

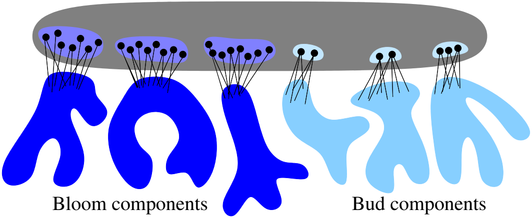
- Our algorithm marks the bags of a tree-decomposition of G.

Main (informal) ideas of our algorithm

- Our algorithm marks the bags of a tree-decomposition of G.
- Let r be an integer that is also given to the algorithm.

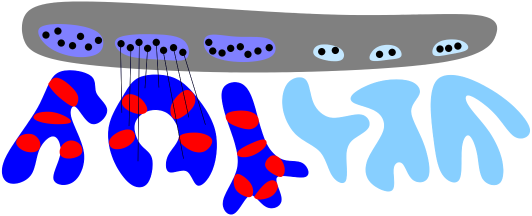
Main (informal) ideas of our algorithm

- Our algorithm marks the bags of a tree-decomposition of G.
- Let r be an integer that is also given to the algorithm.
- Given tree-decompositions of the conn. comp. of $G - X$ with $\geqslant r$ neighbors in X, we identify a set of bags $\mathcal{M}$ in a bottom-up manner.



Main (informal) ideas of our algorithm

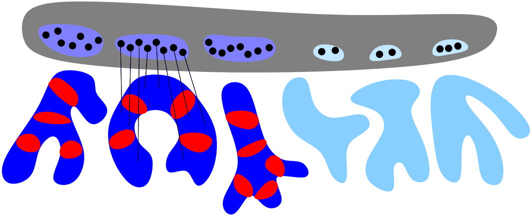
- Our algorithm marks the bags of a tree-decomposition of G.
- Let r be an integer that is also given to the algorithm.
- Given tree-decompositions of the conn. comp. of $G - X$ with $\geqslant r$ neighbors in X, we identify a set of bags $\mathcal{M}$ in a bottom-up manner.



- The set $V(\mathcal{M})$ of vertices contained in marked bags together with X will form the separating part Y_0 of the protrusion decomposition.

Main (informal) ideas of our algorithm

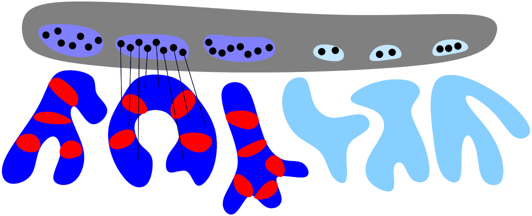
- Our algorithm marks the bags of a tree-decomposition of G.
- Let r be an integer that is also given to the algorithm.
- Given tree-decompositions of the conn. comp. of $G - X$ with $\geqslant r$ neighbors in X, we identify a set of bags $\mathcal{M}$ in a bottom-up manner.



- The set $V(\mathcal{M})$ of vertices contained in marked bags together with X will form the separating part Y_0 of the protrusion decomposition.
- Some marked bags will be mapped bijectively into pairwise vertex-disjoint connected subgraphs of $G - X$, each of which has $\geqslant r$ neighbors in X.

Main (informal) ideas of our algorithm

- Our algorithm marks the bags of a tree-decomposition of G.
- Let r be an integer that is also given to the algorithm.
- Given tree-decompositions of the conn. comp. of $G - X$ with $\geqslant r$ neighbors in X, we identify a set of bags $\mathcal{M}$ in a bottom-up manner.



- The set $V(\mathcal{M})$ of vertices contained in marked bags together with X will form the separating part Y_0 of the protrusion decomposition.
- Some marked bags will be mapped bijectively into pairwise vertex-disjoint connected subgraphs of $G - X$, each of which has $\geqslant r$ neighbors in X.
- Finally, to guarantee that the conn. comp. of $G - (X \cup V(\mathcal{M}))$ form protrusions with small boundary, the set $\mathcal{M}$ is closed under taking LCA.

Description of the bag marking algorithm

$\boxed{\text{Input}}$ G, $X \subseteq V(G)$ s.t. $\mathrm{tw}(G - X) \leqslant t$, and an integer $r > 0$.

Description of the bag marking algorithm

$\boxed{\text{Input}}$ G, $X \subseteq V(G)$ s.t. $\mathrm{tw}(G - X) \leqslant t$, and an integer $r > 0$.

- ⋆ Set $\mathcal{M} \leftarrow \emptyset$ as the set of marked bags.

Description of the bag marking algorithm

$\boxed{\text{Input}}$ G, $X \subseteq V(G)$ s.t. $\mathrm{tw}(G - X) \leqslant t$, and an integer $r > 0$.

- $\star$ Set $\mathcal{M} \leftarrow \emptyset$ as the set of marked bags.
- $\star$ Compute an optimal rooted tree-decomposition $\mathcal{T}_C = (T_C, \mathcal{B}_C)$ of every connected component C of $G - X$ such that $|N_X(C)| \geqslant r$.

Description of the bag marking algorithm

$\boxed{\text{Input}}$ G, $X \subseteq V(G)$ s.t. $\mathrm{tw}(G - X) \leqslant t$, and an integer $r > 0$.

- ⋆ Set $\mathcal{M} \leftarrow \emptyset$ as the set of marked bags.
- ⋆ Compute an optimal rooted tree-decomposition $\mathcal{T}_C = (T_C, \mathcal{B}_C)$ of every connected component C of $G - X$ such that $|N_X(C)| \geqslant r$.
- ⋆ Repeat the following loop for every rooted tree-decomposition $\mathcal{T}_C$:
  while $\mathcal{T}_C$ contains an unprocessed bag **do**:
  - ⋆ Let B be an unprocess. bag at farthest distance from the root of $\mathcal{T}_C$.

Description of the bag marking algorithm

$\boxed{\text{Input}}$ G, $X \subseteq V(G)$ s.t. $\mathrm{tw}(G - X) \leqslant t$, and an integer $r > 0$.

- $\star$ Set $\mathcal{M} \leftarrow \emptyset$ as the set of marked bags.
- $\star$ Compute an optimal rooted tree-decomposition $\mathcal{T}_C = (T_C, \mathcal{B}_C)$ of every connected component C of $G - X$ such that $|N_X(C)| \geqslant r$.
- $\star$ Repeat the following loop for every rooted tree-decomposition $\mathcal{T}_C$:

  while $\mathcal{T}_C$ contains an unprocessed bag **do**:
  - $\star$ Let B be an unprocess. bag at farthest distance from the root of $\mathcal{T}_C$.
  - $\star$ $\boxed{\text{LCA marking step}}$

    if B is the LCA of two marked bags of $\mathcal{M}$:

    $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices of B from every bag of $\mathcal{T}_C$.

Description of the bag marking algorithm

Input G, $X \subseteq V(G)$ s.t. $\mathrm{tw}(G - X) \leqslant t$, and an integer $r > 0$.

- ⋆ Set $\mathcal{M} \leftarrow \emptyset$ as the set of marked bags.
- ⋆ Compute an optimal rooted tree-decomposition $\mathcal{T}_C = (T_C, \mathcal{B}_C)$ of every connected component C of $G - X$ such that $|N_X(C)| \geqslant r$.
- ⋆ Repeat the following loop for every rooted tree-decomposition $\mathcal{T}_C$:

while $\mathcal{T}_C$ contains an unprocessed bag **do**:

- ⋆ Let B be an unprocess. bag at farthest distance from the root of $\mathcal{T}_C$.
- ⋆ LCA marking step

  if B is the LCA of two marked bags of $\mathcal{M}$:

  $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices of B from every bag of $\mathcal{T}_C$.
- ⋆ Bloom-subgraph marking step

  else if G_B contains a connected component C_B s.t. $|N_X(C_B)| \geqslant r$:

  $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices of B from every bag of $\mathcal{T}_C$.

Description of the bag marking algorithm

Input G, $X \subseteq V(G)$ s.t. $\mathrm{tw}(G - X) \leqslant t$, and an integer $r > 0$.

- Set $\mathcal{M} \leftarrow \emptyset$ as the set of marked bags.
- Compute an optimal rooted tree-decomposition $\mathcal{T}_C = (T_C, \mathcal{B}_C)$ of every connected component C of $G - X$ such that $|N_X(C)| \geqslant r$.
- Repeat the following loop for every rooted tree-decomposition $\mathcal{T}_C$:
  while $\mathcal{T}_C$ contains an unprocessed bag **do**:
  - Let B be an unprocess. bag at farthest distance from the root of $\mathcal{T}_C$.
  - LCA marking step

    if B is the LCA of two marked bags of $\mathcal{M}$:
    $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices of B from every bag of $\mathcal{T}_C$.
  - Bloom-subgraph marking step

    else if G_B contains a connected component C_B s.t. $|N_X(C_B)| \geqslant r$:
    $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices of B from every bag of $\mathcal{T}_C$.
  - Bag B is now processed.

Description of the bag marking algorithm

Input G, $X \subseteq V(G)$ s.t. $\mathrm{tw}(G - X) \leqslant t$, and an integer $r > 0$.

- Set $\mathcal{M} \leftarrow \emptyset$ as the set of marked bags.
- Compute an optimal rooted tree-decomposition $\mathcal{T}_C = (T_C, \mathcal{B}_C)$ of every connected component C of $G - X$ such that $|N_X(C)| \geqslant r$.
- Repeat the following loop for every rooted tree-decomposition $\mathcal{T}_C$:

  while $\mathcal{T}_C$ contains an unprocessed bag **do**:
  - Let B be an unprocess. bag at farthest distance from the root of $\mathcal{T}_C$.
  - LCA marking step

    if B is the LCA of two marked bags of $\mathcal{M}$:

    $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices of B from every bag of $\mathcal{T}_C$.
  - Bloom-subgraph marking step

    else if G_B contains a connected component C_B s.t. $|N_X(C_B)| \geqslant r$:

    $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices of B from every bag of $\mathcal{T}_C$.
  - Bag B is now processed.

Return $Y_0 = X \cup V(\mathcal{M})$.

Some properties of the bag marking algorithm

Lemma

The bag marking algorithm can be implemented to run in $O(n)$ time, where the hidden constant depends only on t and r.

Some properties of the bag marking algorithm

Given a graph G and a subset $S \subseteq V(G)$, a cluster of $G - S$ is a maximal collection of connected components of $G - S$ with the same neighborhood in S.

Some properties of the bag marking algorithm

Given a graph G and a subset $S \subseteq V(G)$, a cluster of $G - S$ is a maximal collection of connected components of $G - S$ with the same neighborhood in S.

Proposition

- *Let* r, t *be two positive integers,*
- *let* G *be a graph and* $X \subseteq V(G)$ *such that* $\mathrm{tw}(G - X) \leqslant t$,
- *let* $Y_0 \subseteq V(G)$ *be the output of the algorithm with input* (G, X, r), *and*
- *let* $Y_1, \ldots, Y_\ell$ *be the set of clusters of* $G - Y_0$.

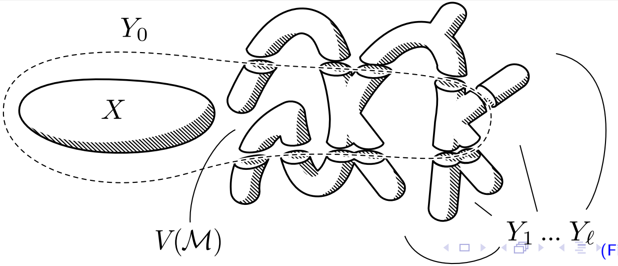
Some properties of the bag marking algorithm

Given a graph G and a subset $S \subseteq V(G)$, a cluster of $G - S$ is a maximal collection of connected components of $G - S$ with the same neighborhood in S.

Proposition

- *Let r, t be two positive integers,*
- *let G be a graph and $X \subseteq V(G)$ such that $\mathrm{tw}(G - X) \leqslant t$,*
- *let $Y_0 \subseteq V(G)$ be the output of the algorithm with input (G, X, r), and*
- *let $Y_1, \ldots, Y_\ell$ be the set of clusters of $G - Y_0$.*

Then $\mathcal{P} := Y_0 \uplus Y_1 \uplus \cdots \uplus Y_\ell$ is a $(\max\{\ell, |Y_0|\}, 2t + r)$-protrusion decomp. of G.



(Figure by Felix Reidl)

Next section is...

Next subsection is...

1. Preliminaries
2. Protrusion decompositions
   - Definitions
   - A simple algorithm to compute them
3. Single-exponential algorithm for Planar-$\mathcal{F}$-Deletion
   - Motivation and our result
   - Sketch of proof
   - Further research
4. Linear kernels on graphs without topological minors
   - Motivation and our result
   - Idea of proof
   - Further research

The (parameterized) PLANAR-$\mathcal{F}$-DELETION problem

Let $\mathcal{F}$ be a finite family of graphs containing at least one planar graph.

The (parameterized) Planar-$\mathcal{F}$-Deletion problem

Let $\mathcal{F}$ be a finite family of graphs containing at least one planar graph.

Planar-$\mathcal{F}$-Deletion

Input:	A graph G and a non-negative integer k.
Parameter:	The integer k.
Question:	Does G have a set $X \subseteq V(G)$ such that $\lvert X \rvert \leqslant k$ and $G - X$ is H-minor-free for every $H \in \mathcal{F}$?

Let $\mathcal{F}$ be a finite family of graphs containing at least one planar graph.

PLANAR-$\mathcal{F}$-DELETION
Input: A graph G and a non-negative integer k.
Parameter: The integer k.
Question: Does G have a set $X \subseteq V(G)$ such that $|X| \leqslant k$ and $G - X$ is H-minor-free for every $H \in \mathcal{F}$?

Some particular cases:

1. $\mathcal{F} = \{K_2\}$: $\equiv$ VERTEX COVER
   $\equiv$ TREEWIDTH-ZERO VERTEX DELETION
2. $\mathcal{F} = \{K_3\}$: $\equiv$ FEEDBACK VERTEX SET
   $\equiv$ TREEWIDTH-ONE VERTEX DELETION
3. $\mathcal{F} = \{K_4\}$: $\equiv$ TREEWIDTH-TWO VERTEX DELETION

How fast can Planar-$\mathcal{F}$-Deletion be solved?

How fast can PLANAR-$\mathcal{F}$-DELETION be solved?

Particular cases:

- $\mathcal{F} = \{K_2\}$ $O^*(1.2738^k)$ [Chen, Fernau, Kanj, Xia '10]
- $\mathcal{F} = \{K_3\}$ $O^*(3.83^k)$ [Cao, Chen, Liu '10]
- $\mathcal{F} = \{\theta_r\}$ $O^*(c^k)$ [Joret, Paul, S., Saurabh, Thomassé '11]
- $\mathcal{F} = \{K_4\}$ $O^*(c^k)$ [Kim, Paul, Philip '12]

How fast can PLANAR-$\mathcal{F}$-DELETION be solved?

Particular cases:

- $\mathcal{F} = \{K_2\}$ $O^*(1.2738^k)$ [Chen, Fernau, Kanj, Xia '10]
- $\mathcal{F} = \{K_3\}$ $O^*(3.83^k)$ [Cao, Chen, Liu '10]
- $\mathcal{F} = \{\theta_r\}$ $O^*(c^k)$ [Joret, Paul, S., Saurabh, Thomassé '11]
- $\mathcal{F} = \{K_4\}$ $O^*(c^k)$ [Kim, Paul, Philip '12]

General case:

- PLANAR-$\mathcal{F}$-DELETION is FPT. [Roberston and Seymour's Graph Minors theory]

How fast can PLANAR-$\mathcal{F}$-DELETION be solved?

Particular cases:

- $\mathcal{F} = \{K_2\}$ $O^*(1.2738^k)$ [Chen, Fernau, Kanj, Xia '10]
- $\mathcal{F} = \{K_3\}$ $O^*(3.83^k)$ [Cao, Chen, Liu '10]
- $\mathcal{F} = \{\theta_r\}$ $O^*(c^k)$ [Joret, Paul, S., Saurabh, Thomassé '11]
- $\mathcal{F} = \{K_4\}$ $O^*(c^k)$ [Kim, Paul, Philip '12]

General case:

- PLANAR-$\mathcal{F}$-DELETION is FPT. [Roberston and Seymour's Graph Minors theory]
- $2^{2^{O(k \log k)}} \cdot n^{O(1)}$ -time algorithm based on standard DP.

How fast can Planar-$\mathcal{F}$-Deletion be solved?

Particular cases:

- $\mathcal{F} = \{K_2\}$ $O^*(1.2738^k)$ [Chen, Fernau, Kanj, Xia '10]
- $\mathcal{F} = \{K_3\}$ $O^*(3.83^k)$ [Cao, Chen, Liu '10]
- $\mathcal{F} = \{\theta_r\}$ $O^*(c^k)$ [Joret, Paul, S., Saurabh, Thomassé '11]
- $\mathcal{F} = \{K_4\}$ $O^*(c^k)$ [Kim, Paul, Philip '12]

General case:

- Planar-$\mathcal{F}$-Deletion is FPT. [Roberston and Seymour's Graph Minors theory]
- $2^{2^{O(k \log k)}} \cdot n^{O(1)}$ -time algorithm based on standard DP.
- $2^{O(k \log k)} \cdot n^2$ -time algorithm. [Fomin, Lokshtanov, Misra, Saurabh '11]

How fast can PLANAR-$\mathcal{F}$-DELETION be solved?

Particular cases:

- $\mathcal{F} = \{K_2\}$ $O^*(1.2738^k)$ [Chen, Fernau, Kanj, Xia '10]
- $\mathcal{F} = \{K_3\}$ $O^*(3.83^k)$ [Cao, Chen, Liu '10]
- $\mathcal{F} = \{\theta_r\}$ $O^*(c^k)$ [Joret, Paul, S., Saurabh, Thomassé '11]
- $\mathcal{F} = \{K_4\}$ $O^*(c^k)$ [Kim, Paul, Philip '12]

General case:

- PLANAR-$\mathcal{F}$-DELETION is FPT. [Roberston and Seymour's Graph Minors theory]
- $2^{2^{O(k \log k)}} \cdot n^{O(1)}$ -time algorithm based on standard DP.
- $2^{O(k \log k)} \cdot n^2$ -time algorithm. [Fomin, Lokshtanov, Misra, Saurabh '11]
- $2^{O(k)} \cdot n \log^2 n$ -time algorithm for PLANAR-CONNECTED-$\mathcal{F}$-DELETION. [Fomin, Lokshtanov, Misra, Saurabh '12]

Our result

Theorem

The PLANAR-$\mathcal{F}$-DELETION *problem can be solved in time* $2^{O(k)} \cdot n^2$.

- This result unifies a number of algorithms in the literature.

Our result

Theorem

The PLANAR-$\mathcal{F}$-DELETION *problem can be solved in time* $2^{O(k)} \cdot n^2$.

- This result unifies a number of algorithms in the literature.
- No hope for a $2^{o(k)} \cdot n^{O(1)}$-time algorithm (under ETH). [Chen *et al.* '05]

  That is, the function $2^{O(k)}$ in our theorem is best possible.

Next subsection is...

First step: use iterative compression

Using iterative compression the PLANAR-$\mathcal{F}$-DELETION problem can be reduced in single-exponential time to the following problem:

First step: use iterative compression

Using iterative compression the Planar-$\mathcal{F}$-Deletion problem can be reduced in single-exponential time to the following problem:

Disjoint Planar-$\mathcal{F}$-Deletion
Input: A graph G, a non-negative integer k, and a set $X \subseteq V(G)$ with $|X| = k$ s.t. $G - X$ is $\mathcal{F}$-minor-free.

First step: use iterative compression

Using iterative compression the PLANAR-$\mathcal{F}$-DELETION problem can be reduced in single-exponential time to the following problem:

DISJOINT PLANAR-$\mathcal{F}$-DELETION

Input: A graph G, a non-negative integer k, and a set $X \subseteq V(G)$ with $|X| = k$ s.t. $G - X$ is $\mathcal{F}$-minor-free.

Parameter: The integer k.

Question: Does G have a set $\boxed{\tilde{X} \subseteq V(G) \setminus X}$ such that $|\tilde{X}| < k$ and $G - \tilde{X}$ is H-minor-free for every $H \in \mathcal{F}$?

We call $\tilde{X}$ an alternative solution.

First step: use iterative compression

Using iterative compression the PLANAR-$\mathcal{F}$-DELETION problem can be reduced in single-exponential time to the following problem:

DISJOINT PLANAR-$\mathcal{F}$-DELETION

Input: A graph G, a non-negative integer k, and a set $X \subseteq V(G)$ with $|X| = k$ s.t. $G - X$ is $\mathcal{F}$-minor-free.

Parameter: The integer k.

Question: Does G have a set $\boxed{\tilde{X} \subseteq V(G) \setminus X}$ such that $|\tilde{X}| < k$ and $G - \tilde{X}$ is H-minor-free for every $H \in \mathcal{F}$?

We call $\tilde{X}$ an alternative solution.

Lemma (well-kwown)

If DISJOINT PLANAR-$\mathcal{F}$-DELETION *can be solved in time* $O^*(c^k)$ *for some* $c \in \mathbb{N}^+$, *then* PLANAR-$\mathcal{F}$-DELETION *can be solved in* $O^*((c+1)^k)$.

Working hypothesis: an alternative solution $\tilde{X}$ does exist in $G - X$.

Working hypothesis: an alternative solution $\tilde{X}$ does exist in $G - X$.

Observation:

If (G, X, k) is a Yes-instance of Disjoint Planar-$\mathcal{F}$-Deletion, then

- $G[X]$ is $\mathcal{F}$-minor-free
- $G[V \setminus X]$ is $\mathcal{F}$-minor-free

Working hypothesis: an alternative solution $\tilde{X}$ does exist in $G - X$.

Observation:

If (G, X, k) is a YES-instance of DISJOINT PLANAR-$\mathcal{F}$-DELETION, then

- $G[X]$ is $\mathcal{F}$-minor-free $\Rightarrow$ $G[X]$ has bounded tw!!
- $G[V \setminus X]$ is $\mathcal{F}$-minor-free $\Rightarrow$ $G[V \setminus X]$ has bounded tw!!

Working hypothesis: an alternative solution $\tilde{X}$ does exist in $G - X$.

Observation:

If (G, X, k) is a YES-instance of DISJOINT PLANAR-$\mathcal{F}$-DELETION, then

- $G[X]$ is $\mathcal{F}$-minor-free $\Rightarrow$ $G[X]$ has bounded tw!!
- $G[V \setminus X]$ is $\mathcal{F}$-minor-free $\Rightarrow$ $G[V \setminus X]$ has bounded tw!!

$\star$ Let $r := |V(H)|$ for H being some planar graph in the family $\mathcal{F}$.

Working hypothesis: an alternative solution $\tilde{X}$ does exist in $G - X$.

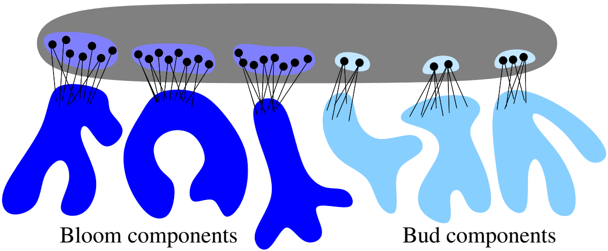
Observation:

If (G, X, k) is a YES-instance of DISJOINT PLANAR-$\mathcal{F}$-DELETION, then

- $G[X]$ is $\mathcal{F}$-minor-free $\Rightarrow$ $G[X]$ has bounded tw!!
- $G[V \setminus X]$ is $\mathcal{F}$-minor-free $\Rightarrow$ $G[V \setminus X]$ has bounded tw!!

⋆ Let $r := |V(H)|$ for H being some planar graph in the family $\mathcal{F}$.

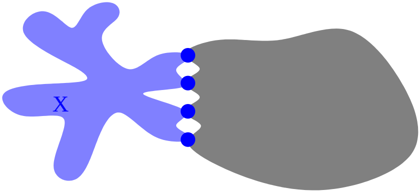
⋆ A connected component C of $G - X$ is called a bloom component if $|N_X(C)| \geqslant r$, and a bud component otherwise.



Linear protrusion decompositions

⋆ Recall that a β-protrusion in a graph G is a subset $Y \subseteq V(G)$ such that

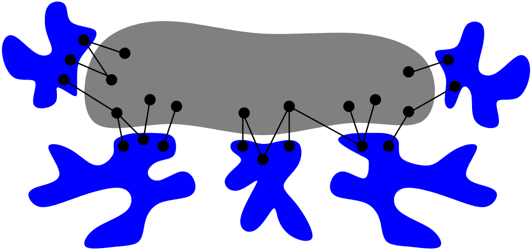
$|\partial(Y)| \leqslant \beta$ and $\mathrm{tw}(G[Y]) \leqslant \beta$



Linear protrusion decompositions

⋆ Recall that a β-protrusion in a graph G is a subset $Y \subseteq V(G)$ such that

$$|\partial(Y)| \leqslant \beta \text{ and } \mathrm{tw}(G[Y]) \leqslant \beta$$



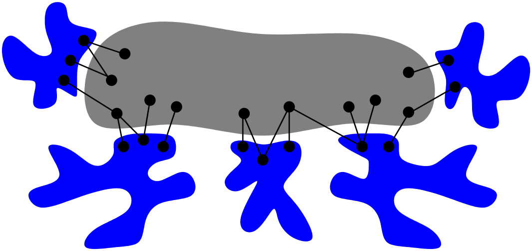
⋆ A partition $\mathcal{P} = Y_0 \uplus Y_1 \uplus \cdots \uplus Y_\ell$ of $V(G)$ with $\max\{\ell, |Y_0|\} \leqslant \alpha$ is an (α, β)-protrusion decomposition if for every $1 \leqslant i \leqslant \ell$,

$$N(Y_i) \subseteq Y_0 \quad \text{and} \quad Y_i \cup N_{Y_0}(Y_i) \text{ is a } \beta\text{-protrusion}$$

Linear protrusion decompositions

⋆ Recall that a β-protrusion in a graph G is a subset $Y \subseteq V(G)$ such that

$$|\partial(Y)| \leqslant \beta \text{ and } \mathrm{tw}(G[Y]) \leqslant \beta$$



⋆ A partition $\mathcal{P} = Y_0 \uplus Y_1 \uplus \cdots \uplus Y_\ell$ of $V(G)$ with $\max\{\ell, |Y_0|\} \leqslant \alpha$ is an (α, β)-protrusion decomposition if for every $1 \leqslant i \leqslant \ell$,

$$N(Y_i) \subseteq Y_0 \quad \text{and} \quad Y_i \cup N_{Y_0}(Y_i) \text{ is a } \beta\text{-protrusion}$$

⋆ $\mathcal{P}$ is $\boxed{\text{linear}}$ with respect to a parameter k whenever $\alpha = O(k)$.

Algorithm to solve Disjoint Planar-$\mathcal{F}$-Deletion

⋆ We will use our algorithm to compute protrusion decompositions.

Algorithm to solve Disjoint Planar-$\mathcal{F}$-Deletion

⋆ Recall that $r = |V(H)|$,

Algorithm to solve DISJOINT PLANAR-$\mathcal{F}$-DELETION

$\star$ Recall that $r = |V(H)|$, and that $\mathrm{tw}(G[V \setminus X]) \leqslant t_{\mathcal{F}}$,

Algorithm to solve DISJOINT PLANAR-$\mathcal{F}$-DELETION

$\star$ Recall that $r = |V(H)|$, and that $\mathrm{tw}(G[V \setminus X]) \leqslant t_{\mathcal{F}}$, so the set $X \subseteq V(G)$ will be the treewidth-bounding set which is given to the algorithm.

Algorithm to solve Disjoint Planar-$\mathcal{F}$-Deletion

⋆ Recall that $r = |V(H)|$, and that $\mathrm{tw}(G[V \setminus X]) \leqslant t_{\mathcal{F}}$, so the set $X \subseteq V(G)$ will be the treewidth-bounding set which is given to the algorithm.

⋆ But it turns out that, with input (G, X, r), the set Y_0 output by our algorithm does not define a linear protrusion decomposition of G, which is crucial for us...

Algorithm to solve DISJOINT PLANAR-$\mathcal{F}$-DELETION

$\star$ Recall that $r = |V(H)|$, and that $\mathrm{tw}(G[V \setminus X]) \leqslant t_{\mathcal{F}}$, so the set $X \subseteq V(G)$ will be the treewidth-bounding set which is given to the algorithm.

$\star$ But it turns out that, with input (G, X, r), the set Y_0 output by our algorithm does not define a linear protrusion decomposition of G, which is crucial for us...

$\boxed{1}$ Guess the intersection $I = \tilde{X} \cap Y_0$ of the alt. solution $\tilde{X}$ with Y_0 s.t.:

- $G - I$ has a linear protrusion decomposition
$$\mathcal{P} = Y_0 \uplus Y_1 \uplus \cdots \uplus Y_\ell$$
- with $X \subseteq Y_0$ and $\tilde{X} \setminus I \subseteq V(G) \setminus Y_0$.

Algorithm to solve DISJOINT PLANAR-$\mathcal{F}$-DELETION

$\star$ Recall that $r = |V(H)|$, and that $\mathrm{tw}(G[V \setminus X]) \leqslant t_{\mathcal{F}}$, so the set $X \subseteq V(G)$ will be the treewidth-bounding set which is given to the algorithm.

$\star$ But it turns out that, with input (G, X, r), the set Y_0 output by our algorithm does not define a linear protrusion decomposition of G, which is crucial for us...

$\boxed{1}$ Guess the intersection $I = \tilde{X} \cap Y_0$ of the alt. solution $\tilde{X}$ with Y_0 s.t.:

- $G - I$ has a linear protrusion decomposition

$$\mathcal{P} = Y_0 \uplus Y_1 \uplus \cdots \uplus Y_\ell$$

- with $X \subseteq Y_0$ and $\tilde{X} \setminus I \subseteq V(G) \setminus Y_0$.

By carefully analyzing the output of our bag marking algorithm

Algorithm to solve Disjoint Planar-$\mathcal{F}$-Deletion

$\star$ Recall that $r = |V(H)|$, and that $\mathrm{tw}(G[V \setminus X]) \leqslant t_{\mathcal{F}}$, so the set $X \subseteq V(G)$ will be the treewidth-bounding set which is given to the algorithm.

$\star$ But it turns out that, with input (G, X, r), the set Y_0 output by our algorithm does not define a linear protrusion decomposition of G, which is crucial for us...

$\boxed{1}$ Guess the intersection $I = \tilde{X} \cap Y_0$ of the alt. solution $\tilde{X}$ with Y_0 s.t.:

- $G - I$ has a linear protrusion decomposition

$$\mathcal{P} = Y_0 \uplus Y_1 \uplus \cdots \uplus Y_\ell$$

- with $X \subseteq Y_0$ and $\tilde{X} \setminus I \subseteq V(G) \setminus Y_0$.

$\boxed{\text{By carefully analyzing the output of our bag marking algorithm}}$

$\boxed{2}$ Finally, compute $\tilde{X} \setminus I$, given a linear protrusion decomposition.

Algorithm to solve Disjoint Planar-$\mathcal{F}$-Deletion

$\star$ Recall that $r = |V(H)|$, and that $\mathrm{tw}(G[V \setminus X]) \leqslant t_{\mathcal{F}}$, so the set $X \subseteq V(G)$ will be the treewidth-bounding set which is given to the algorithm.

$\star$ But it turns out that, with input (G, X, r), the set Y_0 output by our algorithm does not define a linear protrusion decomposition of G, which is crucial for us...

1. Guess the intersection $I = \tilde{X} \cap Y_0$ of the alt. solution $\tilde{X}$ with Y_0 s.t.:
   - $G - I$ has a linear protrusion decomposition
   $$\mathcal{P} = Y_0 \uplus Y_1 \uplus \cdots \uplus Y_\ell$$
   - with $X \subseteq Y_0$ and $\tilde{X} \setminus I \subseteq V(G) \setminus Y_0$.

By carefully analyzing the output of our bag marking algorithm

2. Finally, compute $\tilde{X} \setminus I$, given a linear protrusion decomposition.

Based on the finite index of MSO-definable properties (automaton theory)

Algorithm to solve DISJOINT PLANAR-$\mathcal{F}$-DELETION

⋆ Recall that $r = |V(H)|$, and that $\mathrm{tw}(G[V \setminus X]) \leqslant t_{\mathcal{F}}$, so the set $X \subseteq V(G)$ will be the treewidth-bounding set which is given to the algorithm.

⋆ But it turns out that, with input (G, X, r), the set Y_0 output by our algorithm does not define a linear protrusion decomposition of G, which is crucial for us...

$\boxed{1}$ Guess the intersection $I = \tilde{X} \cap Y_0$ of the alt. solution $\tilde{X}$ with Y_0 s.t.:

- $G - I$ has a linear protrusion decomposition

$$\mathcal{P} = Y_0 \uplus Y_1 \uplus \cdots \uplus Y_\ell$$

- with $X \subseteq Y_0$ and $\tilde{X} \setminus I \subseteq V(G) \setminus Y_0$.

By carefully analyzing the output of our bag marking algorithm

$\boxed{2}$ Finally, compute $\tilde{X} \setminus I$, given a linear protrusion decomposition.

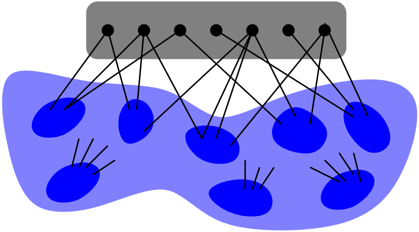
Based on the finite index of MSO-definable properties (automaton theory)

⋆ Both steps can be done in single-exponential time.

First step: analysis of the bag marking algorithm

Lemma (edge simulation to chop bloom components)

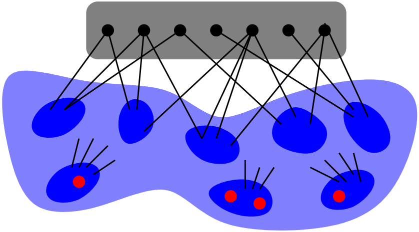
If $C_1, \ldots, C_\ell$ is a collection of connected pairwise vertex-disjoint subgraphs of $G - X$ such that $|N_X(C_i)| \geqslant r$ for $1 \leqslant i \leqslant \ell$, then $\boxed{\ell \leqslant (1 + \alpha_r) \cdot k}$.



First step: analysis of the bag marking algorithm

Lemma (edge simulation to chop bloom components)

If $C_1, \ldots, C_\ell$ is a collection of connected pairwise vertex-disjoint subgraphs of $G - X$ such that $|N_X(C_i)| \geqslant r$ for $1 \leqslant i \leqslant \ell$, then $\boxed{\ell \leqslant (1 + \alpha_r) \cdot k}$.



Proposition (Thomason '01)

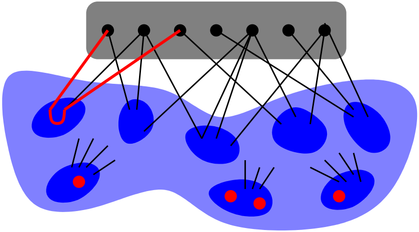
There exists a constant $\alpha < 0.320$ such that any n-vertex graph with no K_r-minor has at most $\alpha_r \cdot n = (\alpha \cdot r\sqrt{\log r}) \cdot n$ edges.

(Recall that $r = |V(H)|$, for H being any planar graph in $\mathcal{F}$)

First step: analysis of the bag marking algorithm

Lemma (edge simulation to chop bloom components)

If $C_1, \ldots, C_\ell$ is a collection of connected pairwise vertex-disjoint subgraphs of $G - X$ such that $|N_X(C_i)| \geqslant r$ for $1 \leqslant i \leqslant \ell$, then $\boxed{\ell \leqslant (1 + \alpha_r) \cdot k}$.



Proposition (Thomason '01)

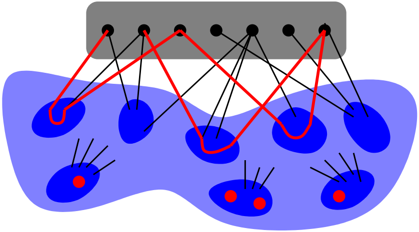
There exists a constant $\alpha < 0.320$ such that any n-vertex graph with no K_r-minor has at most $\alpha_r \cdot n = (\alpha \cdot r\sqrt{\log r}) \cdot n$ edges.

(Recall that $r = |V(H)|$, for H being any planar graph in $\mathcal{F}$)

First step: analysis of the bag marking algorithm

Lemma (edge simulation to chop bloom components)

If $C_1, \ldots, C_\ell$ is a collection of connected pairwise vertex-disjoint subgraphs of $G - X$ such that $|N_X(C_i)| \geqslant r$ for $1 \leqslant i \leqslant \ell$, then $\boxed{\ell \leqslant (1 + \alpha_r) \cdot k}$.



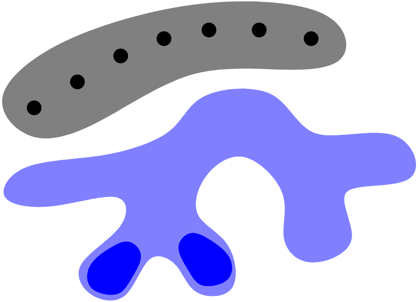
Proposition (Thomason '01)

There exists a constant $\alpha < 0.320$ such that any n-vertex graph with no K_r-minor has at most $\alpha_r \cdot n = (\alpha \cdot r\sqrt{\log r}) \cdot n$ edges.

(Recall that $r = |V(H)|$, for H being any planar graph in $\mathcal{F}$)

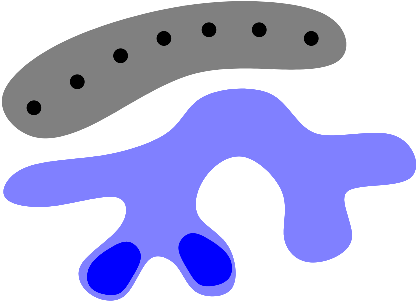
Chopping bloom components (2)

Consider an optimal tree-decomposition $\mathcal{T} = (T, \mathcal{B})$ of a "bloom" connected component C of $G - X$ (i.e., $|N_X(C)| \geqslant r$)



Chopping bloom components (2)

Consider an optimal tree-decomposition $\mathcal{T} = (T, \mathcal{B})$ of a "bloom" connected component C of $G - X$ (i.e., $|N_X(C)| \geqslant r$)



Recall our bottom-up Bag Marking algorithm:

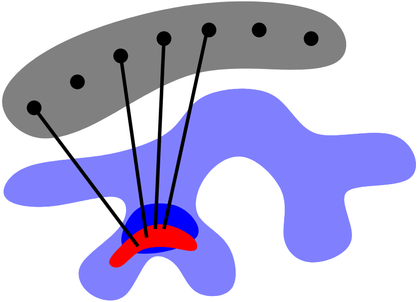
if a bag B is the LCA of two marked bags of $\mathcal{M}$, or

G_B contains a connected bloom component, then

- $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices in B from the bags of $\mathcal{T}$

Chopping bloom components (2)

Consider an optimal tree-decomposition $\mathcal{T} = (T, \mathcal{B})$ of a "bloom" connected component C of $G - X$ (i.e., $|N_X(C)| \geqslant r$)



Recall our bottom-up BAG MARKING algorithm:

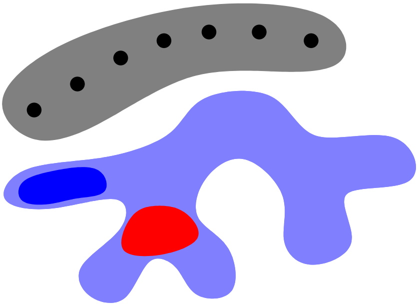
if a bag B is the LCA of two marked bags of $\mathcal{M}$, or

G_B contains a connected bloom component, then

- $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices in B from the bags of $\mathcal{T}$

Chopping bloom components (2)

Consider an optimal tree-decomposition $\mathcal{T} = (T, \mathcal{B})$ of a "bloom" connected component C of $G - X$ (i.e., $|N_X(C)| \geqslant r$)



Recall our bottom-up BAG MARKING algorithm:

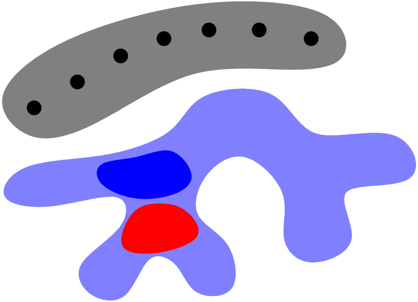
if a bag B is the LCA of two marked bags of $\mathcal{M}$, or

G_B contains a connected bloom component, then

- $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices in B from the bags of $\mathcal{T}$

Chopping bloom components (2)

Consider an optimal tree-decomposition $\mathcal{T} = (T, \mathcal{B})$ of a "bloom" connected component C of $G - X$ (i.e., $|N_X(C)| \geqslant r$)



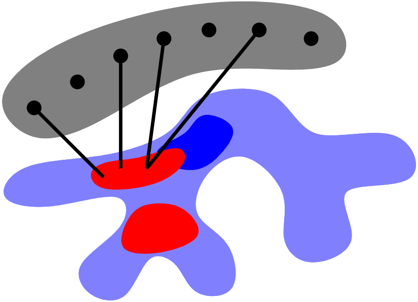
Recall our bottom-up BAG MARKING algorithm:

if a bag B is the LCA of two marked bags of $\mathcal{M}$, or G_B contains a connected bloom component, then

- $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices in B from the bags of $\mathcal{T}$

Chopping bloom components (2)

Consider an optimal tree-decomposition $\mathcal{T} = (T, \mathcal{B})$ of a "bloom" connected component C of $G - X$ (i.e., $|N_X(C)| \geqslant r$)



Recall our bottom-up Bag Marking algorithm:

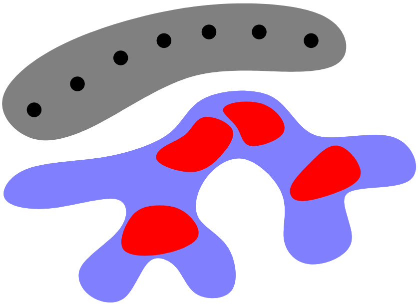
if a bag B is the LCA of two marked bags of $\mathcal{M}$, or

G_B contains a connected bloom component, then

- $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices in B from the bags of $\mathcal{T}$

Chopping bloom components (2)

Consider an optimal tree-decomposition $\mathcal{T} = (T, \mathcal{B})$ of a "bloom" connected component C of $G - X$ (i.e., $|N_X(C)| \geqslant r$)



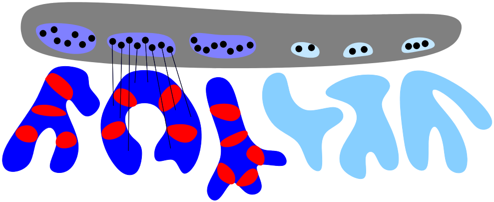
Recall our bottom-up BAG MARKING algorithm:

if a bag B is the LCA of two marked bags of $\mathcal{M}$, or

G_B contains a connected bloom component, then

- $\mathcal{M} \leftarrow \mathcal{M} \cup \{B\}$ and remove the vertices in B from the bags of $\mathcal{T}$

Chopping bloom components (3)



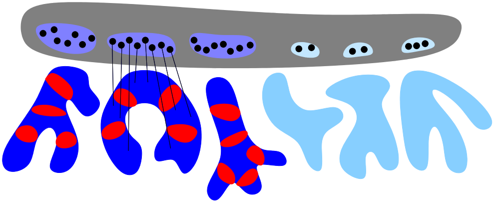
Lemma ($|Y_0| = O(k)$ and every component is a protrusion)

If (G, X, k) is a Yes*-instance of* Disjoint Planar-$\mathcal{F}$-Deletion*, then*

- $Y_0 = X \cup V(\mathcal{M})$ *has size at most* $\boxed{k + 2t_{\mathcal{F}} \cdot (1 + \alpha_r) \cdot k}$.
- *Every connected component C of $G - Y_0$ satisfies*

$$|N_X(C)| \leqslant r \text{ and } |N_{Y_0}(C)| \leqslant r + 2t_{\mathcal{F}}.$$

Chopping bloom components (3)



Lemma ($|Y_0| = O(k)$ and every component is a protrusion)

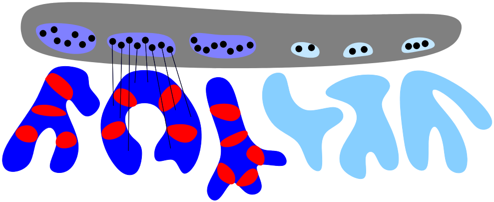
If (G, X, k) *is a* YES*-instance of* DISJOINT PLANAR-$\mathcal{F}$-DELETION, *then*

- $Y_0 = X \cup V(\mathcal{M})$ *has size at most* $\boxed{k + 2t_{\mathcal{F}} \cdot (1 + \alpha_r) \cdot k}$.
- *Every connected component* C *of* $G - Y_0$ *satisfies*

$$|N_X(C)| \leqslant r \text{ and } |N_{Y_0}(C)| \leqslant r + 2t_{\mathcal{F}}.$$

- Note that $k = |X|$,

Chopping bloom components (3)



Lemma ($|Y_0| = O(k)$ and every component is a protrusion)

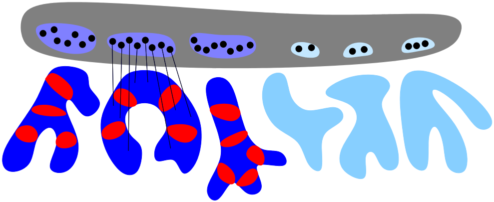
If (G, X, k) is a Yes*-instance of* Disjoint Planar-$\mathcal{F}$-Deletion*, then*

- $Y_0 = X \cup V(\mathcal{M})$ *has size at most* $\boxed{k + 2t_{\mathcal{F}} \cdot (1 + \alpha_r) \cdot k}$.
- *Every connected component* C *of* $G - Y_0$ *satisfies*

$$|N_X(C)| \leqslant r \text{ and } |N_{Y_0}(C)| \leqslant r + 2t_{\mathcal{F}}.$$

- Note that $k = |X|$,
- $\mathrm{tw}(G - X) \leqslant t_{\mathcal{F}}$, and

Chopping bloom components (3)



Lemma ($|Y_0| = O(k)$ and every component is a protrusion)

If (G, X, k) is a Yes*-instance of* Disjoint Planar-$\mathcal{F}$-Deletion, *then*

- $Y_0 = X \cup V(\mathcal{M})$ *has size at most* $\boxed{k + 2t_{\mathcal{F}} \cdot (1 + \alpha_r) \cdot k}$.
- *Every connected component* C *of* $G - Y_0$ *satisfies*

$$|N_X(C)| \leqslant r \text{ and } |N_{Y_0}(C)| \leqslant r + 2t_{\mathcal{F}}.$$

- Note that $k = |X|$,
- $\mathrm{tw}(G - X) \leqslant t_{\mathcal{F}}$, and
- $|\mathcal{M}| \leqslant (1 + \alpha_r) \cdot k$ (by the "edge simulation" Lemma)

Computing a linear protrusion decomposition

Remark: Therefore, Y_0 and the connected components of $G - Y_0$ form a protrusion decomposition of G... but not a linear one!

Computing a linear protrusion decomposition

Remark: Therefore, Y_0 and the connected components of $G - Y_0$ form a protrusion decomposition of G... but not a linear one!

We need that $\#\text{protrusions} = O(k)$.

Computing a linear protrusion decomposition

Remark: Therefore, Y_0 and the connected components of $G - Y_0$ form a protrusion decomposition of G... but not a linear one!

We need that $\#\text{protrusions} = O(k)$.

Branching step:

$$\boxed{\text{Guess } I = \tilde{X} \cap Y_0 \text{ among the } 2^{O(k)} \text{ subsets of } V(\mathcal{M})}$$

Computing a linear protrusion decomposition

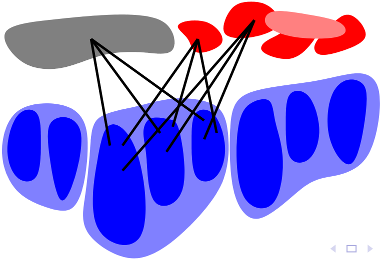
Remark: Therefore, Y_0 and the connected components of $G - Y_0$ form a protrusion decomposition of G... but not a linear one!

We need that $\#\text{protrusions} = O(k)$.

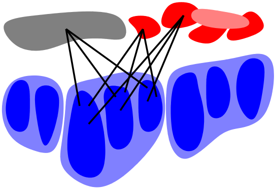
Branching step:

$$\boxed{\text{Guess } I = \tilde{X} \cap Y_0 \text{ among the } 2^{O(k)} \text{ subsets of } V(\mathcal{M})}$$

Let $G_I := G - I$. Recall that a cluster of $G_I - Y_0$ is a maximal set of connected components of $G_I - Y_0$ with the same neighborhood in Y_0.



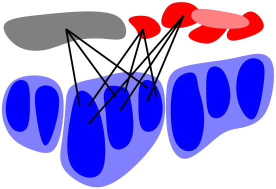
Linear protrusion decomposition (2)



Lemma (For some choice of I, $\#\text{clusters} = O(k)$)

If $(G_I, Y_0 \setminus I, k - |I|)$ is a Yes*-instance of* Disjoint Planar-$\mathcal{F}$-Deletion*, then the number ℓ of clusters of of $G_I - Y_0$ is at most $\boxed{(5t_{\mathcal{F}}\alpha_r\mu_r) \cdot k}$.*

Linear protrusion decomposition (2)



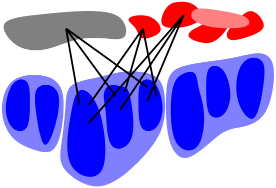
Lemma (For some choice of I, #clusters $= O(k)$)

If $(G_I, Y_0 \setminus I, k - |I|)$ is a Yes*-instance of* Disjoint Planar-$\mathcal{F}$-Deletion*, then the number ℓ of clusters of of $G_I - Y_0$ is at most $\boxed{(5t_{\mathcal{F}}\alpha_r\mu_r) \cdot k}$.*

Proposition (Fomin, Oum, Thilikos '10)

There exists a constant $\mu < 11.355$ such that for all $r > 2$, every n-vertex graph with no K_r-minor has at most $\mu_r \cdot n = 2^{\mu \cdot r \log \log r} \cdot n$ cliques.

Linear protrusion decomposition (2)



Lemma (For some choice of I, $\#\text{clusters} = O(k)$)

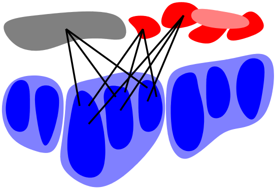
If $(G_I, Y_0 \setminus I, k - |I|)$ is a YES*-instance of* DISJOINT PLANAR-$\mathcal{F}$-DELETION*, then the number ℓ of clusters of of $G_I - Y_0$ is at most* $\boxed{(5t_{\mathcal{F}}\alpha_r\mu_r) \cdot k}$.

Proposition (Fomin, Oum, Thilikos '10)

There exists a constant $\mu < 11.355$ such that for all $r > 2$, every n-vertex graph with no K_r-minor has at most $\mu_r \cdot n = 2^{\mu \cdot r \log\log r} \cdot n$ cliques.

⋆ At most $\ell' = k - |I|$ clusters $C_1, \ldots, C_{\ell'}$ intersect the alternative solution $\tilde{X}$.

Linear protrusion decomposition (2)



Lemma (For some choice of I, #clusters $= O(k)$)

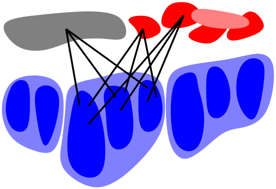
If $(G_I, Y_0 \setminus I, k - |I|)$ is a YES*-instance of* DISJOINT PLANAR-$\mathcal{F}$-DELETION, *then the number ℓ of clusters of of $G_I - Y_0$ is at most* $\boxed{(5t_{\mathcal{F}}\alpha_r\mu_r) \cdot k}$.

Proposition (Fomin, Oum, Thilikos '10)

There exists a constant $\mu < 11.355$ such that for all $r > 2$, every n-vertex graph with no K_r-minor has at most $\mu_r \cdot n = 2^{\mu \cdot r \log\log r} \cdot n$ cliques.

We have that $G' = G_I - \cup_{i=1}^{\ell'} C_i$ is $\mathcal{F}$-minor-free.

Linear protrusion decomposition (2)



Lemma (For some choice of I, #clusters $= O(k)$)

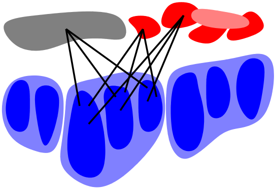
If $(G_I, Y_0 \setminus I, k - |I|)$ *is a* Yes*-instance of* Disjoint Planar-$\mathcal{F}$-Deletion, *then the number* ℓ *of clusters of of* $G_I - Y_0$ *is at most* $\boxed{(5t_{\mathcal{F}}\alpha_r\mu_r) \cdot k}$.

Proposition (Fomin, Oum, Thilikos '10)

There exists a constant $\mu < 11.355$ *such that for all* $r > 2$, *every* n*-vertex graph with no* K_r*-minor has at most* $\mu_r \cdot n = 2^{\mu \cdot r \log \log r} \cdot n$ *cliques.*

$\star$ Using edge simulation we construct a minor of G' on vertices of Y_0.

Linear protrusion decomposition (2)



Lemma (For some choice of I, #clusters $= O(k)$)

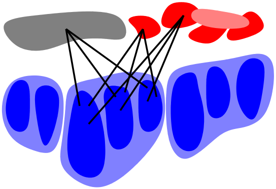
If $(G_I, Y_0 \setminus I, k - |I|)$ is a YES*-instance of* DISJOINT PLANAR-$\mathcal{F}$-DELETION, *then the number ℓ of clusters of of $G_I - Y_0$ is at most* $\boxed{(5t_{\mathcal{F}}\alpha_r\mu_r) \cdot k}$.

Proposition (Fomin, Oum, Thilikos '10)

There exists a constant $\mu < 11.355$ such that for all $r > 2$, every n-vertex graph with no K_r-minor has at most $\mu_r \cdot n = 2^{\mu \cdot r \log \log r} \cdot n$ cliques.

⋆ As before, the number of clusters used so far is at most $\alpha_r \cdot k$.

Linear protrusion decomposition (2)



Lemma (For some choice of I, $\#\text{clusters} = O(k)$)

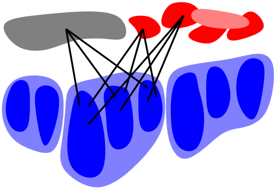
If $(G_I, Y_0 \setminus I, k - |I|)$ is a YES*-instance of* DISJOINT PLANAR-$\mathcal{F}$-DELETION, *then the number ℓ of clusters of of $G_I - Y_0$ is at most $\boxed{(5t_{\mathcal{F}}\alpha_r\mu_r) \cdot k}$.*

Proposition (Fomin, Oum, Thilikos '10)

There exists a constant $\mu < 11.355$ such that for all $r > 2$, every n-vertex graph with no K_r-minor has at most $\mu_r \cdot n = 2^{\mu \cdot r \log \log r} \cdot n$ cliques.

⋆ When we cannot add more edges, all neighborhoods of clusters are cliques!

Linear protrusion decomposition (2)



Lemma (For some choice of I, $\#\text{clusters} = O(k)$)

If $(G_I, Y_0 \setminus I, k - |I|)$ is a Yes-*instance of* Disjoint Planar-$\mathcal{F}$-Deletion, *then the number ℓ of clusters of of $G_I - Y_0$ is at most $\boxed{(5t_{\mathcal{F}}\alpha_r\mu_r) \cdot k}$.*

Proposition (Fomin, Oum, Thilikos '10)

There exists a constant $\mu < 11.355$ such that for all $r > 2$, every n-vertex graph with no K_r-minor has at most $\mu_r \cdot n = 2^{\mu \cdot r \log \log r} \cdot n$ cliques.

⋆ Now we use the Proposition: the number of remaining clusters is $\mu_r \cdot k$.

Back to the road map of the algorithm

Therefore, the partition $\mathcal{P} = Y_0 \uplus C_1 \uplus \cdots \uplus C_\ell$ is a

$$\boxed{(O(k), r + 2t_{\mathcal{F}})\text{-protrusion decomposition of } G_I = G - I}$$

Back to the road map of the algorithm

Therefore, the partition $\mathcal{P} = Y_0 \uplus C_1 \uplus \cdots \uplus C_\ell$ is a

$$\boxed{(O(k), r + 2t_{\mathcal{F}})\text{-protrusion decomposition of } G_I = G - I}$$

Recall the two main steps of our algorithm:

$\boxed{1}$ Guess the intersection $I = \tilde{X} \cap Y_0$ of the alt. solution $\tilde{X}$ with Y_0 s.t.:

- $G - I$ has a linear protrusion decomposition

$$\mathcal{P} = Y_0 \uplus C_1 \uplus \cdots \uplus C_\ell$$

- with $X \subseteq Y_0$ and $\tilde{X} \setminus I \subseteq V(G) \setminus Y_0$.

Back to the road map of the algorithm

Therefore, the partition $\mathcal{P} = Y_0 \uplus C_1 \uplus \cdots \uplus C_\ell$ is a

$$\boxed{(O(k), r + 2t_{\mathcal{F}})\text{-protrusion decomposition of } G_I = G - I}$$

Recall the two main steps of our algorithm:

$\boxed{1}$ Guess the intersection $I = \tilde{X} \cap Y_0$ of the alt. solution $\tilde{X}$ with Y_0 s.t.:

- $G - I$ has a linear protrusion decomposition

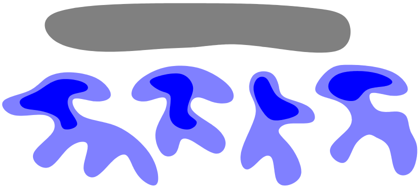
$$\mathcal{P} = Y_0 \uplus C_1 \uplus \cdots \uplus C_\ell$$

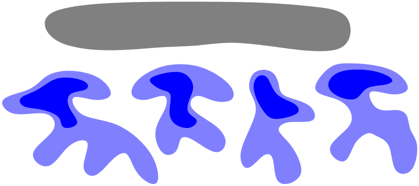
- with $X \subseteq Y_0$ and $\tilde{X} \setminus I \subseteq V(G) \setminus Y_0$.

$\boxed{2}$ Finally, compute $\tilde{X} \setminus I$, given a linear protrusion decomposition.

$\boxed{\text{Based on the finite index of MSO-definable properties (automaton theory)}}$

Solving the problem when given a linear protrusion decomposition

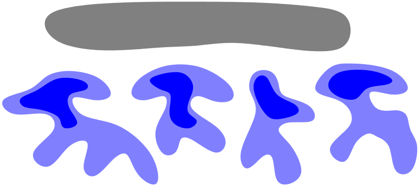




Main ingredients of our approach:

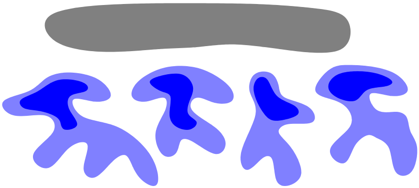
- ⋆ We define an equivalence relation on subsets of vertices of each restricted protrusion Y_i (roughly, same class if they behave in the same way).

Solving the problem when given a linear protrusion decomposition



Main ingredients of our approach:

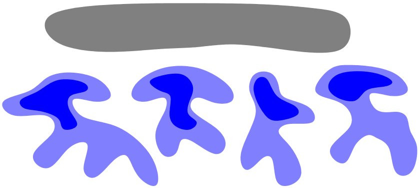
- ⋆ We define an equivalence relation on subsets of vertices of each restricted protrusion Y_i (roughly, same class if they behave in the same way).
- ⋆ Each of these equiv. relations defines finitely many equivalence classes s.t. any partial solution on Y_i can be replaced with one of the representatives.
  (by the finite index of MSO-definable properties) [Bodlaender, de Fluiter '01]



Main ingredients of our approach:

- ⋆ We define an equivalence relation on subsets of vertices of each restricted protrusion Y_i (roughly, same class if they behave in the same way).
- ⋆ Each of these equiv. relations defines finitely many equivalence classes s.t. any partial solution on Y_i can be replaced with one of the representatives.
  (by the finite index of MSO-definable properties) [Bodlaender, de Fluiter '01]
- ⋆ We use a decomposability property of the solution: there exists a solution which is formed by the union of one representative per restricted protrusion.

Solving the problem when given a linear protrusion decomposition



Main ingredients of our approach:

- ⋆ We define an equivalence relation on subsets of vertices of each restricted protrusion Y_i (roughly, same class if they behave in the same way).
- ⋆ Each of these equiv. relations defines finitely many equivalence classes s.t. any partial solution on Y_i can be replaced with one of the representatives.
  (by the finite index of MSO-definable properties) [Bodlaender, de Fluiter '01]
- ⋆ We use a decomposability property of the solution: there exists a solution which is formed by the union of one representative per restricted protrusion.
- ⋆ To make the algorithm constructive and uniform on the family $\mathcal{F}$, we use classic arguments from tree automaton theory (like method of test sets).

Next subsection is...

Conclusions and further research

Theorem

The PLANAR-$\mathcal{F}$-DELETION *problem can be solved in time* $2^{O(k)} \cdot n^2$.

Theorem

The PLANAR-$\mathcal{F}$-DELETION *problem can be solved in time* $2^{O(k)} \cdot n^2$.

★ Can a single-exponential algorithm exist when the family $\mathcal{F}$ does not contain any planar graph?

For $\mathcal{F} = \{K_5, K_{3,3}\}$, an explicit FPT algorithm is known.
It runs in time $2^{O(k \log k)} \cdot n$. [Jansen, Lokshtanov, Saurabh '14]

Conclusions and further research

Theorem

The PLANAR-$\mathcal{F}$-DELETION *problem can be solved in time* $2^{O(k)} \cdot n^2$.

★ Can a single-exponential algorithm exist when the family $\mathcal{F}$ does not contain any planar graph?

For $\mathcal{F} = \{K_5, K_{3,3}\}$, an explicit FPT algorithm is known.
It runs in time $2^{O(k \log k)} \cdot n$. [Jansen, Lokshtanov, Saurabh '14]

★ There exists a randomized constant-factor approximation algorithm for PLANAR-$\mathcal{F}$-DELETION. [Fomin, Lokshtanov, Misra, Saurabh '12]

Finding a deterministic constant-factor approximation remains open.

Conclusions and further research

Theorem

The $\textsc{Planar-}\mathcal{F}\textsc{-Deletion}$ *problem can be solved in time* $2^{O(k)} \cdot n^2$.

- ★ Can a single-exponential algorithm exist when the family $\mathcal{F}$ does not contain any planar graph?

  For $\mathcal{F} = \{K_5, K_{3,3}\}$, an explicit FPT algorithm is known.
  It runs in time $2^{O(k \log k)} \cdot n$. [Jansen, Lokshtanov, Saurabh '14]

- ★ There exists a randomized constant-factor approximation algorithm for $\textsc{Planar-}\mathcal{F}\textsc{-Deletion}$. [Fomin, Lokshtanov, Misra, Saurabh '12]

  Finding a deterministic constant-factor approximation remains open.

- ★ We could forbid the family of graphs $\mathcal{F}$ according to another containment relation, like topological minor.

Next section is...

Next subsection is...

Kernels

- A kernel for a parameterized problem Π is an algorithm that given (x, k) outputs, in time polynomial in $|x| + k$, an instance (x', k') s.t.:
  - ⋆ $(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$, and

Kernels

- A kernel for a parameterized problem Π is an algorithm that given (x, k) outputs, in time polynomial in $|x| + k$, an instance (x', k') s.t.:
  - ⋆ $(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$, and
  - ⋆ Both $\boxed{|x'|, k' \leqslant g(k)}$, where g is some computable function.

Kernels

- A kernel for a parameterized problem Π is an algorithm that given (x, k) outputs, in time polynomial in $|x| + k$, an instance (x', k') s.t.:
  - $(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$, and
  - Both $\boxed{|x'|, k' \leqslant g(k)}$, where g is some computable function.

- The function g is called the size of the kernel.
  - If $g(k) = k^{O(1)}$: Π admits a polynomial kernel.
  - If $g(k) = O(k)$: Π admits a linear kernel.

Kernels

- A kernel for a parameterized problem Π is an algorithm that given (x, k) outputs, in time polynomial in $|x| + k$, an instance (x', k') s.t.:
  - ⋆ $(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$, and
  - ⋆ Both $\boxed{|x'|, k' \leqslant g(k)}$, where g is some computable function.
- The function g is called the size of the kernel.
  - ⋆ If $g(k) = k^{O(1)}$: Π admits a polynomial kernel.
  - ⋆ If $g(k) = O(k)$: Π admits a linear kernel.
- Folklore result: for a parameterized problem Π,

$$\boxed{\Pi \text{ is FPT} \quad \Leftrightarrow \quad \Pi \text{ admits a kernel}}$$

Kernels

- A kernel for a parameterized problem Π is an algorithm that given (x, k) outputs, in time polynomial in $|x| + k$, an instance (x', k') s.t.:
  - ⋆ $(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$, and
  - ⋆ Both $\boxed{|x'|, k' \leqslant g(k)}$, where g is some computable function.

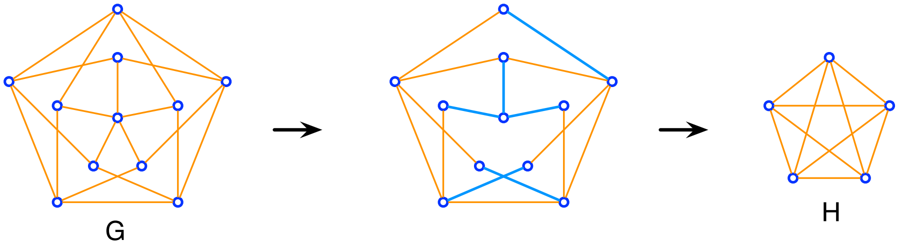
- The function g is called the size of the kernel.
  - ⋆ If $g(k) = k^{O(1)}$: Π admits a polynomial kernel.
  - ⋆ If $g(k) = O(k)$: Π admits a linear kernel.

- Folklore result: for a parameterized problem Π,

$$\boxed{\Pi \text{ is } \mathrm{FPT} \quad \Leftrightarrow \quad \Pi \text{ admits a kernel}}$$

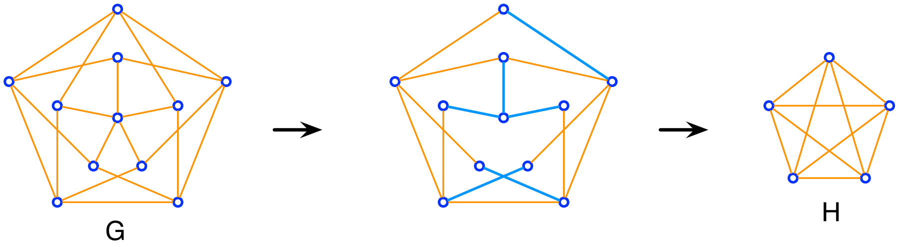
- Question: which FPT problems admit linear or polynomial kernels?

Minors and topological minors



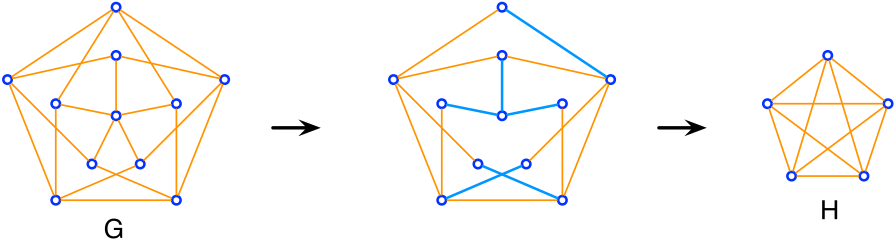
- H is a minor of a graph G if H can be obtained from a subgraph of G by contracting edges.

Minors and topological minors



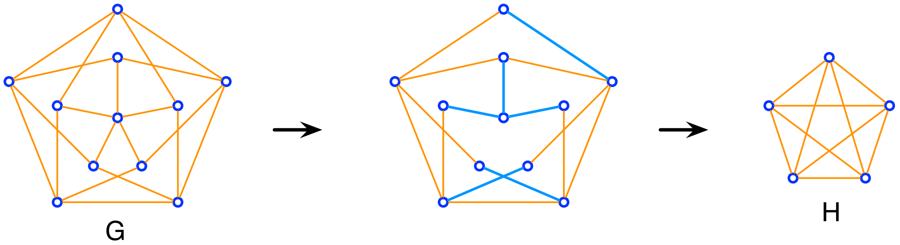
- H is a minor of a graph G if H can be obtained from a subgraph of G by contracting edges.
- H is a topological minor of G if H can be obtained from a subgraph of G by contracting edges with at least one endpoint of deg $\leqslant 2$.

Minors and topological minors



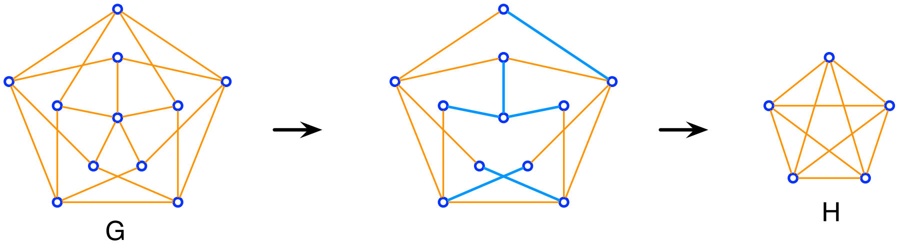
- H is a minor of a graph G if H can be obtained from a subgraph of G by contracting edges.
- H is a topological minor of G if H can be obtained from a subgraph of G by contracting edges with at least one endpoint of deg $\leqslant 2$.
- Therefore: $\boxed{H \text{ minor of } G \Rightarrow H \text{ topological minor of } G}$.

Minors and topological minors



- H is a minor of a graph G if H can be obtained from a subgraph of G by contracting edges.
- H is a topological minor of G if H can be obtained from a subgraph of G by contracting edges with at least one endpoint of deg $\leqslant 2$.
- Therefore: $\boxed{H \text{ minor of } G \not\Leftarrow H \text{ topological minor of } G}$.

Minors and topological minors



- H is a minor of a graph G if H can be obtained from a subgraph of G by contracting edges.
- H is a topological minor of G if H can be obtained from a subgraph of G by contracting edges with at least one endpoint of deg $\leqslant 2$.
- Therefore: $\boxed{H \text{ minor of } G \not\Leftarrow H \text{ topological minor of } G}$.
- Fixed H: $\boxed{H\text{-minor-free graphs} \subseteq H\text{-topological-minor-free graphs}}$.

Linear kernels on sparse graphs – an overview

- DOMINATING SET on planar graphs. [Alber, Fellows, Niedermeier '04]

Linear kernels on sparse graphs – an overview

- DOMINATING SET on planar graphs. [Alber, Fellows, Niedermeier '04]
- Framework for several problems on planar graphs. [Guo, Niedermeier '04]

Linear kernels on sparse graphs – an overview

- DOMINATING SET on planar graphs. [Alber, Fellows, Niedermeier '04]
- Framework for several problems on planar graphs. [Guo, Niedermeier '04]
- Meta-result for graphs of bounded genus.
  [Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh, Thilikos '09]

Linear kernels on sparse graphs – an overview

- DOMINATING SET on planar graphs. [Alber, Fellows, Niedermeier '04]
- Framework for several problems on planar graphs. [Guo, Niedermeier '04]
- Meta-result for graphs of bounded genus. [Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh, Thilikos '09]
- Meta-result for H-minor-free graphs. [Fomin, Lokshtanov, Saurabh, Thilikos '10]

Linear kernels on sparse graphs – an overview

- DOMINATING SET on planar graphs. [Alber, Fellows, Niedermeier '04]
- Framework for several problems on planar graphs. [Guo, Niedermeier '04]
- Meta-result for graphs of bounded genus. [Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh, Thilikos '09]
- Meta-result for H-minor-free graphs. [Fomin, Lokshtanov, Saurabh, Thilikos '10]
- Meta-result for H-topological-minor-free graphs. [Our result]

Our result

Theorem

Fix a graph H. Let Π be a parameterized graph problem on the class of H-topological-minor-free graphs that is treewidth-bounding and has finite integer index (FII). Then Π admits a linear kernel.

Our result

Theorem

Fix a graph H. Let Π be a parameterized graph problem on the class of H-topological-minor-free graphs that is treewidth-bounding and has finite integer index (FII). Then Π admits a linear kernel.

- A parameterized graph problem Π is treewidth-bounding if $\exists$ constants c, t such that if $(G, k) \in \Pi$ then

$$\exists X \subseteq V(G) \text{ s.t. } |X| \leqslant c \cdot k \text{ and } \mathrm{tw}(G - X) \leqslant t.$$

Our result

Theorem

Fix a graph H. Let Π be a parameterized graph problem on the class of H-topological-minor-free graphs that is treewidth-bounding and has finite integer index (FII). Then Π admits a linear kernel.

- A parameterized graph problem Π is treewidth-bounding if $\exists$ constants c, t such that if $(G, k) \in \Pi$ then

$$\exists X \subseteq V(G) \text{ s.t. } |X| \leqslant c \cdot k \text{ and } \mathrm{tw}(G - X) \leqslant t.$$

- FII allows us to replace large protrusions by smaller gadgets...

Our result

Theorem

Fix a graph H. Let Π be a parameterized graph problem on the class of H-topological-minor-free graphs that is treewidth-bounding and has finite integer index (FII). Then Π admits a linear kernel.

- A parameterized graph problem Π is treewidth-bounding if $\exists$ constants c, t such that if $(G, k) \in \Pi$ then

$$\exists X \subseteq V(G) \text{ s.t. } |X| \leqslant c \cdot k \text{ and } \mathrm{tw}(G - X) \leqslant t.$$

- FII allows us to replace large protrusions by smaller gadgets...

★ We assume that the $\boxed{\text{gadgets are given}}$. Our algorithm is non-uniform.

Our result

Theorem

Fix a graph H. Let Π be a parameterized graph problem on the class of H-topological-minor-free graphs that is treewidth-bounding and has finite integer index (FII). Then Π admits a linear kernel.

- A parameterized graph problem Π is treewidth-bounding if $\exists$ constants c, t such that if $(G, k) \in \Pi$ then

$$\exists X \subseteq V(G) \text{ s.t. } |X| \leqslant c \cdot k \text{ and } \mathrm{tw}(G - X) \leqslant t.$$

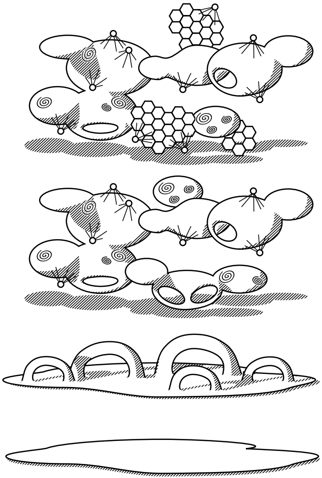
- FII allows us to replace large protrusions by smaller gadgets...

★ We assume that the $\boxed{\text{gadgets are given}}$. Our algorithm is non-uniform.

Problems affected by our result:

Treewidth-t Vertex Deletion, Chordal Vertex Deletion, Interval Vertex Deletion, Edge Dominating Set, Feedback Vertex Set, Connected Vertex Cover, ...

Linear kernels on sparse graphs – the conditions

H-topological-minor-free		treewidth-bounding
$\cup$		
H-minor-free		bidimensional, separation property
$\cup$		
bounded genus		quasi-compact
$\cup$		
planar		“distance-property”

(Figure by Felix Reidl)

Are our conditions very restrictive?

$\boxed{\text{We require FII + treewidth-bounding}}$

Are our conditions very restrictive?

$\boxed{\text{We require FII + treewidth-bounding}}$

- FII is necessary when using protrusion replacement rules.

Are our conditions very restrictive?

We require FII + treewidth-bounding

- FII is necessary when using protrusion replacement rules.
- What about requiring the problems to be treewidth-bounding?

Are our conditions very restrictive?

$\boxed{\text{We require FII + treewidth-bounding}}$

- FII is necessary when using protrusion replacement rules.

- What about requiring the problems to be treewidth-bounding?

  Conditions on H-minor-free graphs:
  bidimensional + separation property. [Fomin, Lokshtanov, Saurabh, Thilikos '10]

Are our conditions very restrictive?

$\boxed{\text{We require FII + treewidth-bounding}}$

- FII is necessary when using protrusion replacement rules.
- What about requiring the problems to be treewidth-bounding?

  Conditions on H-minor-free graphs:
  bidimensional + separation property. [Fomin, Lokshtanov, Saurabh, Thilikos '10]

  But it holds that

$$\boxed{\text{bidimensional + separation property}} \Rightarrow \boxed{\text{treewidth-bounding}}$$

Are our conditions very restrictive?

$\boxed{\text{We require FII + treewidth-bounding}}$

- FII is necessary when using protrusion replacement rules.

- What about requiring the problems to be treewidth-bounding?

  Conditions on H-minor-free graphs:
  bidimensional + separation property. [Fomin, Lokshtanov, Saurabh, Thilikos '10]

  But it holds that

  $$\boxed{\text{bidimensional + separation property}} \Rightarrow \boxed{\text{treewidth-bounding}}$$

- Thus, our results imply the linear kernels of [Fomin, Lokshtanov, Saurabh, Thilikos '10]

Next subsection is...

Finite Integer Index (FII)

[Bodlaender, de Fluiter '01]

Finite Integer Index (FII)

[Bodlaender, de Fluiter '01]

- Let Π be a parameterized graph problem restricted to a graph class $\mathcal{G}$ and let G_1, G_2 be two t-boundaried graphs in $\mathcal{G}_t$.

Finite Integer Index (FII)

[Bodlaender, de Fluiter '01]

- Let Π be a parameterized graph problem restricted to a graph class $\mathcal{G}$ and let G_1, G_2 be two t-boundaried graphs in $\mathcal{G}_t$.
- We say that $G_1 \equiv_{\Pi,t} G_2$ if there exists a constant $\Delta_{\Pi,t}(G_1, G_2)$ such that for all t-boundaried graphs H and for all k:
  1. $G_1 \oplus H \in \mathcal{G}$ iff $G_2 \oplus H \in \mathcal{G}$;
  2. $(G_1 \oplus H, k) \in \Pi$ iff $(G_2 \oplus H, k + \Delta_{\Pi,t}(G_1, G_2)) \in \Pi$.

Finite Integer Index (FII)

[Bodlaender, de Fluiter '01]

- Let Π be a parameterized graph problem restricted to a graph class $\mathcal{G}$ and let G_1, G_2 be two t-boundaried graphs in $\mathcal{G}_t$.
- We say that $G_1 \equiv_{\Pi,t} G_2$ if there exists a constant $\Delta_{\Pi,t}(G_1, G_2)$ such that for all t-boundaried graphs H and for all k:
  1. $G_1 \oplus H \in \mathcal{G}$ iff $G_2 \oplus H \in \mathcal{G}$;
  2. $(G_1 \oplus H, k) \in \Pi$ iff $(G_2 \oplus H, k + \Delta_{\Pi,t}(G_1, G_2)) \in \Pi$.
- Problem Π has FII in the class $\mathcal{G}$ if for every integer t, the equivalence relation $\equiv_{\Pi,t}$ has a finite number of equivalence classes.

Finite Integer Index (FII)

[Bodlaender, de Fluiter '01]

- Let Π be a parameterized graph problem restricted to a graph class $\mathcal{G}$ and let G_1, G_2 be two t-boundaried graphs in $\mathcal{G}_t$.
- We say that $G_1 \equiv_{\Pi,t} G_2$ if there exists a constant $\Delta_{\Pi,t}(G_1, G_2)$ such that for all t-boundaried graphs H and for all k:
  1. $G_1 \oplus H \in \mathcal{G}$ iff $G_2 \oplus H \in \mathcal{G}$;
  2. $(G_1 \oplus H, k) \in \Pi$ iff $(G_2 \oplus H, k + \Delta_{\Pi,t}(G_1, G_2)) \in \Pi$.
- Problem Π has FII in the class $\mathcal{G}$ if for every integer t, the equivalence relation $\equiv_{\Pi,t}$ has a finite number of equivalence classes.
- Main idea: If a parameterized problem has FII then its instances can be reduced by replacing any "large" protrusion by a "small" gadget (representative in a set $\mathcal{R}_t$) from the same equivalence class.

Finite Integer Index (FII)

[Bodlaender, de Fluiter '01]

- Let Π be a parameterized graph problem restricted to a graph class $\mathcal{G}$ and let G_1, G_2 be two t-boundaried graphs in $\mathcal{G}_t$.
- We say that $G_1 \equiv_{\Pi,t} G_2$ if there exists a constant $\Delta_{\Pi,t}(G_1, G_2)$ such that for all t-boundaried graphs H and for all k:
  1. $G_1 \oplus H \in \mathcal{G}$ iff $G_2 \oplus H \in \mathcal{G}$;
  2. $(G_1 \oplus H, k) \in \Pi$ iff $(G_2 \oplus H, k + \Delta_{\Pi,t}(G_1, G_2)) \in \Pi$.
- Problem Π has FII in the class $\mathcal{G}$ if for every integer t, the equivalence relation $\equiv_{\Pi,t}$ has a finite number of equivalence classes.

- Main idea If a parameterized problem has FII then its instances can be reduced by replacing any "large" protrusion by a "small" gadget (representative in a set $\mathcal{R}_t$) from the same equivalence class.
- The protrusion limit of Π is a function $\rho_\Pi : \mathbb{N} \to \mathbb{N}$ defined as $\rho_\Pi(t) = \max_{G \in \mathcal{R}_t} |V(G)|$.

Finite Integer Index (FII)

[Bodlaender, de Fluiter '01]

- Let Π be a parameterized graph problem restricted to a graph class $\mathcal{G}$ and let G_1, G_2 be two t-boundaried graphs in $\mathcal{G}_t$.
- We say that $G_1 \equiv_{\Pi,t} G_2$ if there exists a constant $\Delta_{\Pi,t}(G_1, G_2)$ such that for all t-boundaried graphs H and for all k:
  1. $G_1 \oplus H \in \mathcal{G}$ iff $G_2 \oplus H \in \mathcal{G}$;
  2. $(G_1 \oplus H, k) \in \Pi$ iff $(G_2 \oplus H, k + \Delta_{\Pi,t}(G_1, G_2)) \in \Pi$.
- Problem Π has FII in the class $\mathcal{G}$ if for every integer t, the equivalence relation $\equiv_{\Pi,t}$ has a finite number of equivalence classes.
- Main idea If a parameterized problem has FII then its instances can be reduced by replacing any "large" protrusion by a "small" gadget (representative in a set $\mathcal{R}_t$) from the same equivalence class.
- The protrusion limit of Π is a function $\rho_\Pi : \mathbb{N} \to \mathbb{N}$ defined as $\rho_\Pi(t) = \max_{G \in \mathcal{R}_t} |V(G)|$. We also define $\rho'_\Pi(t) = \rho_\Pi(2t)$.

Disconnected PLANAR-$\mathcal{F}$-DELETION has not FII

- We prove: if $\mathcal{F}$ is a family of graphs containing some disconnected graph H, then PLANAR-$\mathcal{F}$-DELETION has not FII (in general).

Disconnected Planar-$\mathcal{F}$-Deletion has not FII

- Let o-Π be the non-parameterized version of Planar-$\mathcal{F}$-Deletion. Let G_1 and G_2 be two t-boundaried graphs.

Disconnected PLANAR-$\mathcal{F}$-DELETION has not FII

- Let o-Π be the non-parameterized version of PLANAR-$\mathcal{F}$-DELETION. Let G_1 and G_2 be two t-boundaried graphs. We define $G_1 \sim_{\Pi,t} G_2$ iff $\exists$ integer i such that $\forall$ t-boundaried graph H, it holds

$$\pi(G_1 \oplus H) = \pi(G_2 \oplus H) + i,$$

where $\pi(G)$ denotes the optimal value of problem o-Π on graph G.

Disconnected PLANAR-$\mathcal{F}$-DELETION has not FII

- Let o-Π be the non-parameterized version of PLANAR-$\mathcal{F}$-DELETION. Let G_1 and G_2 be two t-boundaried graphs. We define $G_1 \sim_{\Pi,t} G_2$ iff $\exists$ integer i such that $\forall$ t-boundaried graph H, it holds

$$\pi(G_1 \oplus H) = \pi(G_2 \oplus H) + i,$$

  where $\pi(G)$ denotes the optimal value of problem o-Π on graph G.
- We let $F_1 = K_4$, $F_2 = K_{2,3}$, $F := F_1 \uplus F_2$, and $\mathcal{F} = \{F\}$.

Disconnected Planar-$\mathcal{F}$-Deletion has not FII

- Let o-Π be the non-parameterized version of Planar-$\mathcal{F}$-Deletion. Let G_1 and G_2 be two t-boundaried graphs. We define $G_1 \sim_{\Pi,t} G_2$ iff $\exists$ integer i such that $\forall$ t-boundaried graph H, it holds

$$\pi(G_1 \oplus H) = \pi(G_2 \oplus H) + i,$$

  where $\pi(G)$ denotes the optimal value of problem o-Π on graph G.
- We let $F_1 = K_4$, $F_2 = K_{2,3}$, $F := F_1 \uplus F_2$, and $\mathcal{F} = \{F\}$.
- For $i \geqslant 1$, let G_i (resp. H_i) be the 1-boundaried graph consisting of a boundary vertex v (resp. u) together with i disjoint copies of F_1 (resp. F_2) joined to v (resp. u) by an edge.

Disconnected PLANAR-$\mathcal{F}$-DELETION has not FII

- Let o-Π be the non-parameterized version of PLANAR-$\mathcal{F}$-DELETION. Let G_1 and G_2 be two t-boundaried graphs. We define $G_1 \sim_{\Pi,t} G_2$ iff $\exists$ integer i such that $\forall$ t-boundaried graph H, it holds

$$\pi(G_1 \oplus H) = \pi(G_2 \oplus H) + i,$$

  where $\pi(G)$ denotes the optimal value of problem o-Π on graph G.

- We let $F_1 = K_4$, $F_2 = K_{2,3}$, $F := F_1 \uplus F_2$, and $\mathcal{F} = \{F\}$.
- For $i \geqslant 1$, let G_i (resp. H_i) be the 1-boundaried graph consisting of a boundary vertex v (resp. u) together with i disjoint copies of F_1 (resp. F_2) joined to v (resp. u) by an edge.
- By construction, if $i, j \geqslant 1$, it holds $\pi(G_i \oplus H_j) = \min\{i, j\}$.

Disconnected PLANAR-$\mathcal{F}$-DELETION has not FII

- Let o-Π be the non-parameterized version of PLANAR-$\mathcal{F}$-DELETION. Let G_1 and G_2 be two t-boundaried graphs. We define $G_1 \sim_{\Pi,t} G_2$ iff $\exists$ integer i such that $\forall$ t-boundaried graph H, it holds

$$\pi(G_1 \oplus H) = \pi(G_2 \oplus H) + i,$$

where $\pi(G)$ denotes the optimal value of problem o-Π on graph G.

- We let $F_1 = K_4$, $F_2 = K_{2,3}$, $F := F_1 \uplus F_2$, and $\mathcal{F} = \{F\}$.
- For $i \geqslant 1$, let G_i (resp. H_i) be the 1-boundaried graph consisting of a boundary vertex v (resp. u) together with i disjoint copies of F_1 (resp. F_2) joined to v (resp. u) by an edge.
- By construction, if $i, j \geqslant 1$, it holds $\pi(G_i \oplus H_j) = \min\{i, j\}$.
- Then, if we take $1 \leqslant n < m$,

$$\begin{aligned}
\pi(G_n \oplus H_{n-1}) - \pi(G_m \oplus H_{n-1}) &= (n-1) - (n-1) = 0,\\
\pi(G_n \oplus H_m) - \pi(G_m \oplus H_m) &= n - m < 0.
\end{aligned}$$

Disconnected Planar-$\mathcal{F}$-Deletion has not FII

- Let o-Π be the non-parameterized version of Planar-$\mathcal{F}$-Deletion. Let G_1 and G_2 be two t-boundaried graphs. We define $G_1 \sim_{\Pi,t} G_2$ iff $\exists$ integer i such that $\forall$ t-boundaried graph H, it holds

$$\pi(G_1 \oplus H) = \pi(G_2 \oplus H) + i,$$

where $\pi(G)$ denotes the optimal value of problem o-Π on graph G.

- We let $F_1 = K_4$, $F_2 = K_{2,3}$, $F := F_1 \uplus F_2$, and $\mathcal{F} = \{F\}$.
- For $i \geqslant 1$, let G_i (resp. H_i) be the 1-boundaried graph consisting of a boundary vertex v (resp. u) together with i disjoint copies of F_1 (resp. F_2) joined to v (resp. u) by an edge.
- By construction, if $i, j \geqslant 1$, it holds $\pi(G_i \oplus H_j) = \min\{i, j\}$.
- Then, if we take $1 \leqslant n < m$,

$$\begin{aligned} \pi(G_n \oplus H_{n-1}) - \pi(G_m \oplus H_{n-1}) &= (n-1) - (n-1) = 0, \\ \pi(G_n \oplus H_m) - \pi(G_m \oplus H_m) &= n - m < 0. \end{aligned}$$

- Thus, $G_n, G_m \notin$ same equiv. class of $\sim_{\Pi,1}$ whenever $1 \leqslant n < m$.

Some important ingredients (suppose problem Π has FII)

Some important ingredients (suppose problem Π has FII)

Lemma (The parameter does not increase)

$\forall$ *fixed* t, $\exists$ *finite set* $\mathcal{R}_t$ *of* t*-boundaried graphs s.t. for each* t*-boundaried graph* $G \in \mathcal{G}_t$ $\exists$ $G' \in \mathcal{R}_t$ *s.t.* $G \equiv_{\Pi,t} G'$ *and* $\Delta_{\Pi,t}(G, G') \geqslant 0$.

Some important ingredients (suppose problem Π has FII)

Lemma (The parameter does not increase)

$\forall$ fixed t, $\exists$ finite set $\mathcal{R}_t$ of t-boundaried graphs s.t. for each t-boundaried graph $G \in \mathcal{G}_t$ $\exists$ $G' \in \mathcal{R}_t$ s.t. $G \equiv_{\Pi,t} G'$ and $\Delta_{\Pi,t}(G, G') \geqslant 0$.

Lemma (Finding maximum sized protrusions)

Let t be a constant. Given an n-vertex graph G, a t-protrusion of G with the maximum number of vertices can be found in time $O(n^{t+1})$.

Some important ingredients (suppose problem Π has FII)

Lemma (The parameter does not increase)

$\forall$ fixed t, $\exists$ finite set $\mathcal{R}_t$ of t-boundaried graphs s.t. for each t-boundaried graph $G \in \mathcal{G}_t$ $\exists$ $G' \in \mathcal{R}_t$ s.t. $G \equiv_{\Pi,t} G'$ and $\Delta_{\Pi,t}(G, G') \geqslant 0$.

Lemma (Finding maximum sized protrusions)

Let t be a constant. Given an n-vertex graph G, a t-protrusion of G with the maximum number of vertices can be found in time $O(n^{t+1})$.

Lemma (Big... but not too big!)

If one is given a t-protrusion $X \subseteq V(G)$ s.t. $\rho'_{\Pi}(t) < |X|$, then one can, in time $O(|X|)$, find an equiv. $2t$-protrusion W s.t. $\rho'_{\Pi}(t) < |W| \leqslant 2 \cdot \rho'_{\Pi}(t)$.

Some important ingredients (suppose problem Π has FII)

Lemma (The parameter does not increase)

$\forall$ fixed t, $\exists$ finite set $\mathcal{R}_t$ of t-boundaried graphs s.t. for each t-boundaried graph $G \in \mathcal{G}_t$ $\exists$ $G' \in \mathcal{R}_t$ s.t. $G \equiv_{\Pi,t} G'$ and $\Delta_{\Pi,t}(G, G') \geqslant 0$.

Lemma (Finding maximum sized protrusions)

Let t be a constant. Given an n-vertex graph G, a t-protrusion of G with the maximum number of vertices can be found in time $O(n^{t+1})$.

Lemma (Big... but not too big!)

If one is given a t-protrusion $X \subseteq V(G)$ s.t. $\rho'_\Pi(t) < |X|$, then one can, in time $O(|X|)$, find an equiv. $2t$-protrusion W s.t. $\rho'_\Pi(t) < |W| \leqslant 2 \cdot \rho'_\Pi(t)$.

Lemma (Replacing protrusions of constant size)

For $t \in \mathbb{N}$, suppose that the set $\mathcal{R}_t$ of representatives of $\equiv_{\Pi,t}$ is given. If W is a t-protrusion of size at most a fixed constant c, then one can decide in constant time which $G' \in \mathcal{R}_t$ satisfies $G' \equiv_{\Pi,t} G[W]$.

Protrusion replacement

Protrusion reduction rule

- Let $(G, k) \in \Pi$ and let $t \in \mathbb{N}$ be a constant (to be fixed later).

Protrusion replacement

Protrusion reduction rule

- Let $(G, k) \in \Pi$ and let $t \in \mathbb{N}$ be a constant (to be fixed later).
- Suppose that G has a t-protrusion $W' \subseteq V(G)$ s.t. $|W'| > \rho'_{\Pi}(t)$.

Protrusion replacement

Protrusion reduction rule

- Let $(G, k) \in \Pi$ and let $t \in \mathbb{N}$ be a constant (to be fixed later).
- Suppose that G has a t-protrusion $W' \subseteq V(G)$ s.t. $|W'| > \rho'_{\Pi}(t)$.
- Let $W \subseteq V(G)$ be a $2t$-protrusion of G s.t. $\rho'_{\Pi}(t) < |W| \leqslant 2 \cdot \rho'_{\Pi}(t)$.

Protrusion replacement

Protrusion reduction rule

- Let $(G, k) \in \Pi$ and let $t \in \mathbb{N}$ be a constant (to be fixed later).
- Suppose that G has a t-protrusion $W' \subseteq V(G)$ s.t. $|W'| > \rho'_{\Pi}(t)$.
- Let $W \subseteq V(G)$ be a $2t$-protrusion of G s.t. $\rho'_{\Pi}(t) < |W| \leqslant 2 \cdot \rho'_{\Pi}(t)$.
- We let G_W denote the $2t$-boundaried graph $G[W]$ with boundary $\mathbf{bd}(G_W) = \partial_G(W)$.

Protrusion replacement

Protrusion reduction rule

- Let $(G, k) \in \Pi$ and let $t \in \mathbb{N}$ be a constant (to be fixed later).
- Suppose that G has a t-protrusion $W' \subseteq V(G)$ s.t. $|W'| > \rho'_\Pi(t)$.
- Let $W \subseteq V(G)$ be a $2t$-protrusion of G s.t. $\rho'_\Pi(t) < |W| \leqslant 2 \cdot \rho'_\Pi(t)$.
- We let G_W denote the $2t$-boundaried graph $G[W]$ with boundary $\mathbf{bd}(G_W) = \partial_G(W)$.
- Let further $G_1 \in \mathcal{R}_{2t}$ be the representative of G_W for the equivalence relation $\equiv_{\Pi,|\partial(W)|}$.

Protrusion replacement

Protrusion reduction rule

- Let $(G, k) \in \Pi$ and let $t \in \mathbb{N}$ be a constant (to be fixed later).
- Suppose that G has a t-protrusion $W' \subseteq V(G)$ s.t. $|W'| > \rho'_\Pi(t)$.
- Let $W \subseteq V(G)$ be a $2t$-protrusion of G s.t. $\rho'_\Pi(t) < |W| \leqslant 2 \cdot \rho'_\Pi(t)$.
- We let G_W denote the $2t$-boundaried graph $G[W]$ with boundary $\mathbf{bd}(G_W) = \partial_G(W)$.
- Let further $G_1 \in \mathcal{R}_{2t}$ be the representative of G_W for the equivalence relation $\equiv_{\Pi,|\partial(W)|}$.
- The protrusion reduction rule (for boundary size t) is the following:

Reduce (G, k)
to $(G', k') = (G[V \setminus W] \oplus G_1, k - \Delta_{\Pi,2t}(G_1, G_W))$.

Protrusion replacement

Protrusion reduction rule

- Let $(G, k) \in \Pi$ and let $t \in \mathbb{N}$ be a constant (to be fixed later).
- Suppose that G has a t-protrusion $W' \subseteq V(G)$ s.t. $|W'| > \rho'_\Pi(t)$.
- Let $W \subseteq V(G)$ be a $2t$-protrusion of G s.t. $\rho'_\Pi(t) < |W| \leqslant 2 \cdot \rho'_\Pi(t)$.
- We let G_W denote the $2t$-boundaried graph $G[W]$ with boundary $\mathbf{bd}(G_W) = \partial_G(W)$.
- Let further $G_1 \in \mathcal{R}_{2t}$ be the representative of G_W for the equivalence relation $\equiv_{\Pi,|\partial(W)|}$.
- The protrusion reduction rule (for boundary size t) is the following:

Reduce (G, k)
to $(G', k') = (G[V \setminus W] \oplus G_1, k - \Delta_{\Pi,2t}(G_1, G_W))$.

It runs in polynomial time...

Protrusion replacement

Protrusion reduction rule

- Let $(G, k) \in \Pi$ and let $t \in \mathbb{N}$ be a constant (to be fixed later).
- Suppose that G has a t-protrusion $W' \subseteq V(G)$ s.t. $|W'| > \rho'_\Pi(t)$.
- Let $W \subseteq V(G)$ be a $2t$-protrusion of G s.t. $\rho'_\Pi(t) < |W| \leqslant 2 \cdot \rho'_\Pi(t)$.
- We let G_W denote the $2t$-boundaried graph $G[W]$ with boundary $\mathbf{bd}(G_W) = \partial_G(W)$.
- Let further $G_1 \in \mathcal{R}_{2t}$ be the representative of G_W for the equivalence relation $\equiv_{\Pi,|\partial(W)|}$.
- The protrusion reduction rule (for boundary size t) is the following:

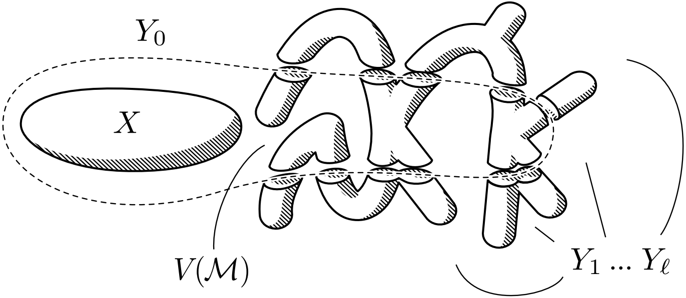
Reduce (G, k)
to $(G', k') = (G[V \setminus W] \oplus G_1, k - \Delta_{\Pi,2t}(G_1, G_W))$.

It runs in $\boxed{\text{polynomial time}}$... given the sets of representatives!

Protrusion decompositions (in case someone forgot!)

An (α, t)-protrusion decomposition of a graph G is a partition $\mathcal{P} = Y_0 \uplus Y_1 \uplus \cdots \uplus Y_\ell$ of $V(G)$ such that:

- for every $1 \leqslant i \leqslant \ell$, $N(Y_i) \subseteq Y_0$;
- for every $1 \leqslant i \leqslant \ell$, $Y_i \cup N_{Y_0}(Y_i)$ is a t-protrusion of G;
- $\max\{\ell, |Y_0|\} \leqslant \alpha$.



(Figure by Felix Reidl)

Kernelization algorithm

1. $\boxed{\text{We apply exhaustively the protrusion replacement rule.}}$

Kernelization algorithm

1. $\boxed{\text{We apply exhaustively the protrusion replacement rule.}}$

   If (G, k) is reduced w.r.t. the protrusion reduction rule with boundary size β (this can be done in polynomial time), $\forall t \leqslant \beta$, every t-protrusion W of G has size $\leqslant \rho'_{\Pi}(t)$.

Kernelization algorithm

1. $\boxed{\text{We apply exhaustively the protrusion replacement rule.}}$

   If (G, k) is reduced w.r.t. the protrusion reduction rule with boundary size β (this can be done in polynomial time), $\forall t \leqslant \beta$, every t-protrusion W of G has size $\leqslant \rho'_{\Pi}(t)$.

   We can choose $\beta := 2t + \omega(H)$, where t comes from the treewidth-bounding property of Π.

Kernelization algorithm

1. $\boxed{\text{We apply exhaustively the protrusion replacement rule.}}$

   If (G, k) is reduced w.r.t. the protrusion reduction rule with boundary size β (this can be done in polynomial time), $\forall t \leqslant \beta$, every t-protrusion W of G has size $\leqslant \rho'_{\Pi}(t)$.

   We can choose $\beta := 2t + \omega(H)$, where t comes from the treewidth-bounding property of Π.

2. $\boxed{\text{We use protrusion decompositions to analyze the kernel size.}}$

Kernelization algorithm

1. We apply exhaustively the protrusion replacement rule.

   If (G, k) is reduced w.r.t. the protrusion reduction rule with boundary size β (this can be done in polynomial time), $\forall t \leqslant \beta$, every t-protrusion W of G has size $\leqslant \rho'_\Pi(t)$.

   We can choose $\beta := 2t + \omega(H)$, where t comes from the treewidth-bounding property of Π.

2. We use protrusion decompositions to analyze the kernel size.

   Using what we explained before, we can easily prove that:

   Let Π be a parameterized graph problem that has FII and is t-treewidth-bounding, both on the class of H-topological-minor-free graphs.

Kernelization algorithm

1. We apply exhaustively the protrusion replacement rule.

   If (G, k) is reduced w.r.t. the protrusion reduction rule with boundary size β (this can be done in polynomial time), $\forall t \leqslant \beta$, every t-protrusion W of G has size $\leqslant \rho'_\Pi(t)$.

   We can choose $\beta := 2t + \omega(H)$, where t comes from the treewidth-bounding property of Π.

2. We use protrusion decompositions to analyze the kernel size.

   Using what we explained before, we can easily prove that:

   Let Π be a parameterized graph problem that has FII and is t-treewidth-bounding, both on the class of H-topological-minor-free graphs. Then any reduced YES-instance (G, k) has a protrusion decomposition $V(G) = Y_0 \uplus Y_1 \uplus \cdots \uplus Y_\ell$ s.t.:

   1. $|Y_0| = O(k)$;
   2. $|Y_i| \leqslant \rho'_\Pi(2t + \omega_\mathcal{H})$ for $1 \leqslant i \leqslant \ell$; and
   3. $\ell = O(k)$.

Next subsection is...

Limits of our approach and further research

- For which notions of sparseness (beyond H-topological-minor-free graphs) can we use our technique to obtain polynomial kernels?

Limits of our approach and further research

- For which notions of sparseness (beyond H-topological-minor-free graphs) can we use our technique to obtain polynomial kernels?

  1. A class $\mathcal{G}$ of graphs $\boxed{\text{locally excludes a minor}}$ if $\forall r \in \mathbb{N}$, $\exists H_r$ s.t. the r-neighborhood of a vertex of any graph of $\mathcal{G}$ excludes H_r as a minor.

     (includes H-minor-free but incomparable with H-topological-minor-free)

Limits of our approach and further research

- For which notions of sparseness (beyond H-topological-minor-free graphs) can we use our technique to obtain polynomial kernels?

  1. A class $\mathcal{G}$ of graphs $\boxed{\text{locally excludes a minor}}$ if $\forall r \in \mathbb{N}$, $\exists H_r$ s.t. the r-neighborhood of a vertex of any graph of $\mathcal{G}$ excludes H_r as a minor.

     (includes H-minor-free but incomparable with H-topological-minor-free)

     Except for a very restricted case, our technique fails.

Limits of our approach and further research

- For which notions of sparseness (beyond H-topological-minor-free graphs) can we use our technique to obtain polynomial kernels?

1. A class $\mathcal{G}$ of graphs locally excludes a minor if $\forall r \in \mathbb{N}$, $\exists H_r$ s.t. the r-neighborhood of a vertex of any graph of $\mathcal{G}$ excludes H_r as a minor.

   (includes H-minor-free but incomparable with H-topological-minor-free)

   Except for a very restricted case, our technique fails.

2. Graphs of bounded expansion (contains H-topological-minor-free)?

Limits of our approach and further research

- For which notions of sparseness (beyond H-topological-minor-free graphs) can we use our technique to obtain polynomial kernels?

  1. A class $\mathcal{G}$ of graphs $\boxed{\text{locally excludes a minor}}$ if $\forall r \in \mathbb{N}$, $\exists H_r$ s.t. the r-neighborhood of a vertex of any graph of $\mathcal{G}$ excludes H_r as a minor.

     (includes H-minor-free but incomparable with H-topological-minor-free)

     Except for a very restricted case, our technique fails.

  2. Graphs of $\boxed{\text{bounded expansion}}$ (contains H-topological-minor-free)?

     Obtaining a kernel for Treewidth-t Vertex Deletion on graphs of bounded expansion is as hard as on general graphs.

Limits of our approach and further research

- For which notions of sparseness (beyond H-topological-minor-free graphs) can we use our technique to obtain polynomial kernels?

  1. A class $\mathcal{G}$ of graphs $\boxed{\text{locally excludes a minor}}$ if $\forall r \in \mathbb{N}$, $\exists H_r$ s.t. the r-neighborhood of a vertex of any graph of $\mathcal{G}$ excludes H_r as a minor.

     (includes H-minor-free but incomparable with H-topological-minor-free)

     Except for a very restricted case, our technique fails.

  2. Graphs of $\boxed{\text{bounded expansion}}$ (contains H-topological-minor-free)?

     Obtaining a kernel for Treewidth-t Vertex Deletion on graphs of bounded expansion is as hard as on general graphs.

     Best known kernel: $k^{O(t)}$. [Fomin, Lokshtanov, Misra, Saurabh '12]

Limits of our approach and further research

- For which notions of sparseness (beyond H-topological-minor-free graphs) can we use our technique to obtain polynomial kernels?

  1. A class $\mathcal{G}$ of graphs $\boxed{\text{locally excludes a minor}}$ if $\forall r \in \mathbb{N}$, $\exists H_r$ s.t. the r-neighborhood of a vertex of any graph of $\mathcal{G}$ excludes H_r as a minor.

     (includes H-minor-free but incomparable with H-topological-minor-free)

     Except for a very restricted case, our technique fails.

  2. Graphs of $\boxed{\text{bounded expansion}}$ (contains H-topological-minor-free)?

     Obtaining a kernel for Treewidth-t Vertex Deletion on graphs of bounded expansion is as hard as on general graphs.

     Best known kernel: $k^{O(t)}$. [Fomin, Lokshtanov, Misra, Saurabh '12]

- Constructing the kernels? Finding the sets of representatives!!

Limits of our approach and further research

- For which notions of sparseness (beyond H-topological-minor-free graphs) can we use our technique to obtain polynomial kernels?

  1. A class $\mathcal{G}$ of graphs $\boxed{\text{locally excludes a minor}}$ if $\forall r \in \mathbb{N}$, $\exists H_r$ s.t. the r-neighborhood of a vertex of any graph of $\mathcal{G}$ excludes H_r as a minor.

     (includes H-minor-free but incomparable with H-topological-minor-free)

     Except for a very restricted case, our technique fails.

  2. Graphs of $\boxed{\text{bounded expansion}}$ (contains H-topological-minor-free)?

     Obtaining a kernel for Treewidth-t Vertex Deletion on graphs of bounded expansion is as hard as on general graphs.

     Best known kernel: $k^{O(t)}$. [Fomin, Lokshtanov, Misra, Saurabh '12]

- Constructing the kernels? Finding the sets of representatives!!

- Explicit constants? Lower bounds on their size?

Gràcies!