Connectivity under vertex failure, logic and algorithm

Alexandre Vigny

With: Michał Pilipczuk, Nicole Schirrmacher, Sebastian Siebertz
and Szymon Toruńczyk

March 22, 2024

Outline

Introduction: Graphs, Algorithms, Logic

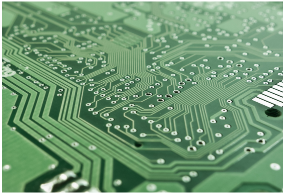
Connectivity under vertex failure

Logics between FO and MSO

Algorithmic graph theory

Given a graph G and a property P: "Does G satisfy P?"

$\rightarrow$ Is G planar?

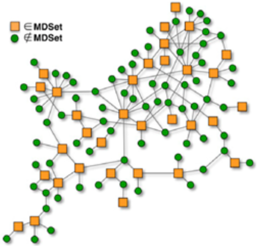


Algorithmic graph theory

Given a graph G and a property P: "Does G satisfy P?"

→ Is G planar?

→ Does G have a k-dominating set?



Algorithmic graph theory

Given a graph G and a property P: "Does G satisfy P?"

→ Is G planar?

→ Does G have a k-dominating set?

→ Is G connected?

Goal: Efficient algorithms ...

... at least for restricted graph classes and/or simple properties.

Logic

First-order (FO) logic

- $\rightarrow$ Can express k-independent set: *There are k vertices, that are not adjacent*
  $\exists x_1 \dots \exists x_k \ \bigwedge_{i<j} (x_i \neq x_j \wedge \neg E(x_i, x_j))$
- $\rightarrow$ Cannot express : connectivity, planarity, 2-colorability, ...

Logic

First-order (FO) logic

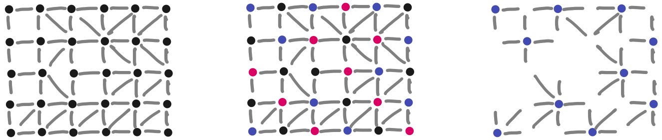
→ Can express k-independent set: *There are k vertices, that are not adjacent*
$\exists x_1 \ldots \exists x_k \bigwedge_{i<j} (x_i \neq x_j \wedge \neg E(x_i, x_j))$

→ Cannot express : connectivity, planarity, 2-colorability, ...

Monadic second-order (MSO) logic

→ More general than FO

→ Can express : 3-colorability:
$\exists X_1 \exists X_2 \exists X_3 \, (\forall x \bigvee_{i<3} x \in X_i) \wedge (\forall x \forall y \, E(x,y) \rightarrow \bigwedge_{i<3} (x \notin X_i \vee y \notin X_i))$



Logic & Meta theorems

Problems can be expressed in logic. (FO, MSO,...)

The $\mathcal{L}$, $\mathcal{C}$ model-checking problem:
Given $\varphi \in \mathcal{L}$ and $G \in \mathcal{C}$, does $G \models \varphi$?

Goal: fixed parameter tractable algorithms $\mathcal{O}(f(\varphi) \cdot |G|^c)$

Logic & Meta theorems

Problems can be expressed in logic. (FO, MSO,...)

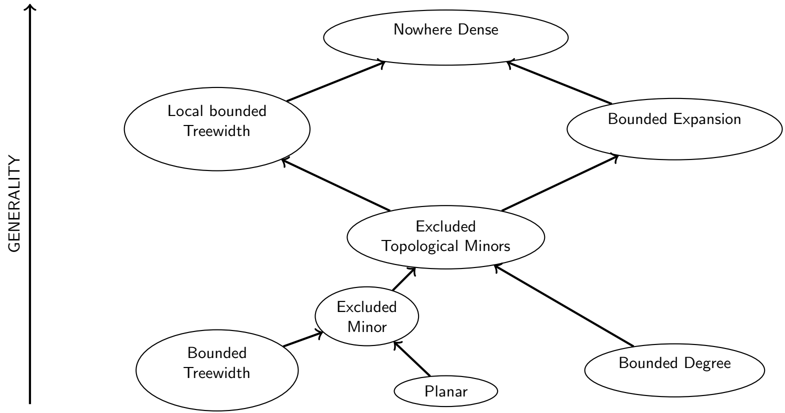
The $\mathcal{L}$, $\mathcal{C}$ model-checking problem:
Given $\varphi \in \mathcal{L}$ and $G \in \mathcal{C}$, does $G \models \varphi$?

Goal: fixed parameter tractable algorithms $\mathcal{O}(f(\varphi) \cdot |G|^c)$

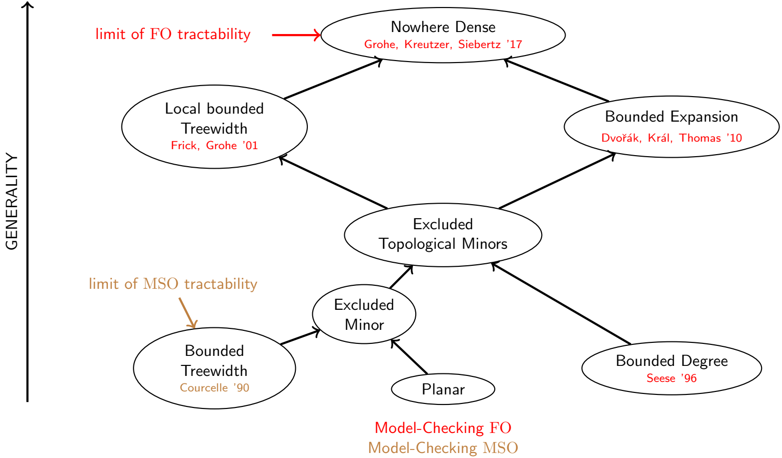
Courcelle's Theorem (1990):
for $\varphi \in \mathrm{MSO}$ and $\text{treewidth}(G) \leq k$, in time $\mathcal{O}(f(\varphi, k) \cdot |G|)$

$\rightarrow$ Generalize many known results, ex:
Arnborg, Proskurowski 1989:
independent sets, dominating sets, graph coloring, Hamiltonian, ...
are linear on partial k-tree.

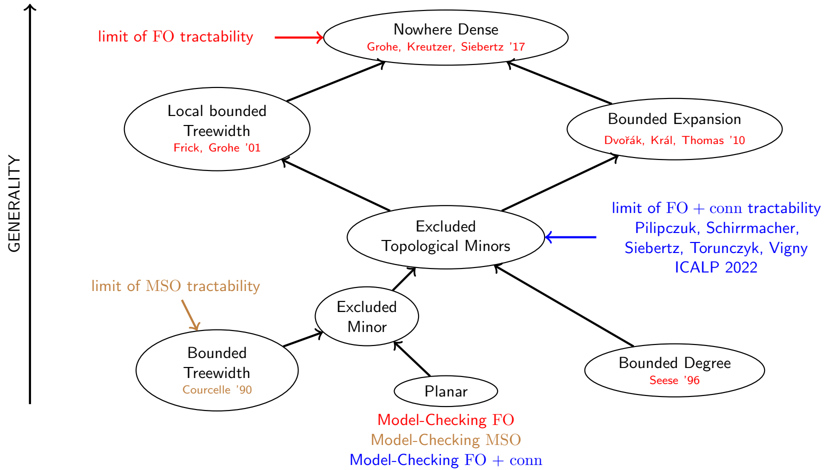
Monotone graph classes



Monotone graph classes



Monotone graph classes



Connectivity under vertex failure

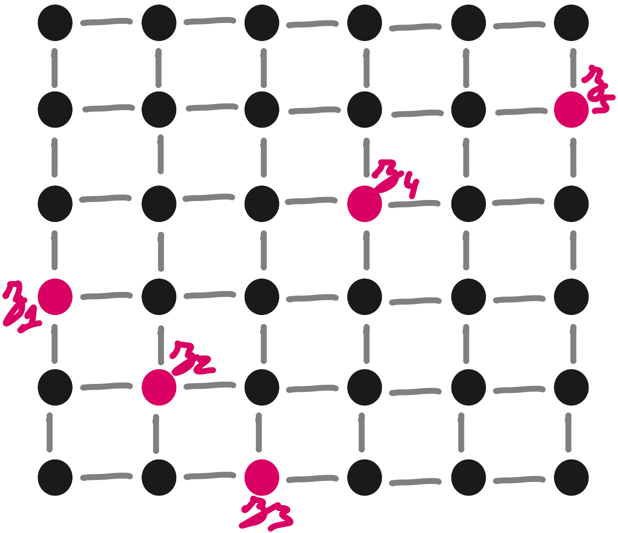
Input: A graph G, and k vertices $v_1, \ldots, v_k$

→ 1st precomputation.

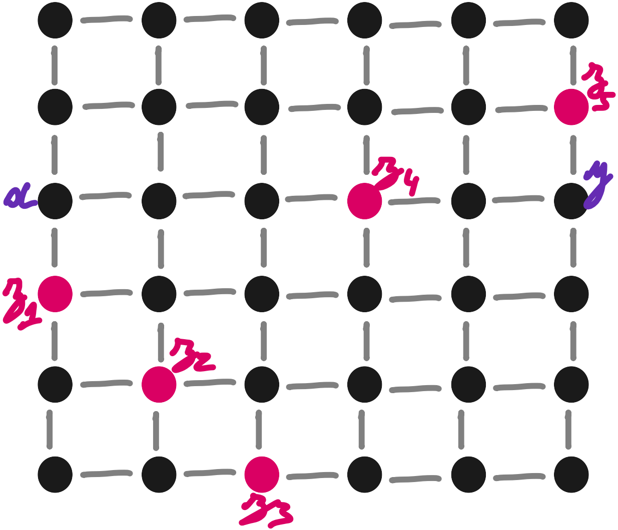
→ 2nd Given x, y: are they connected in $G \setminus \{z_1, \ldots, z_k\}$?

→ 3rd Update vertices $z_1, \ldots, z_k$
Be ready for 2nd as soon as possible.

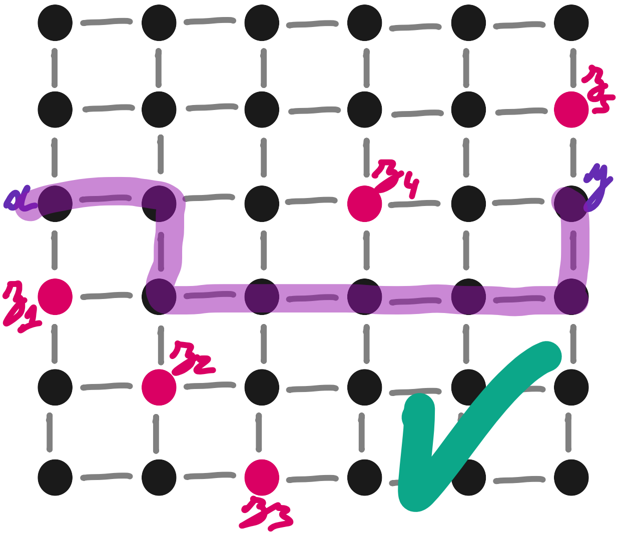
Examples



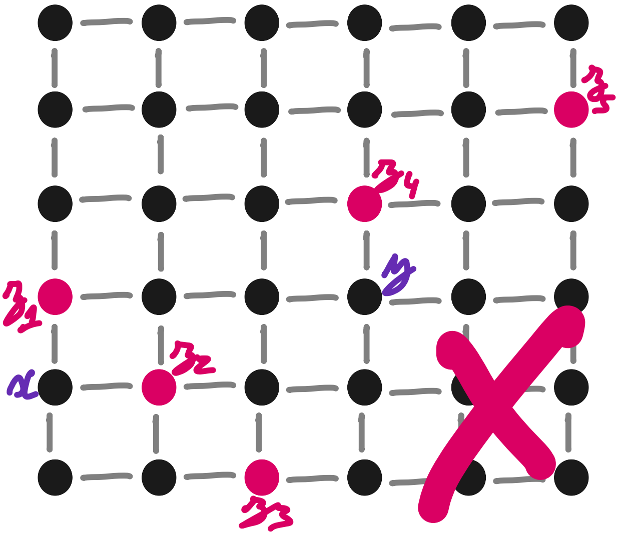
Examples



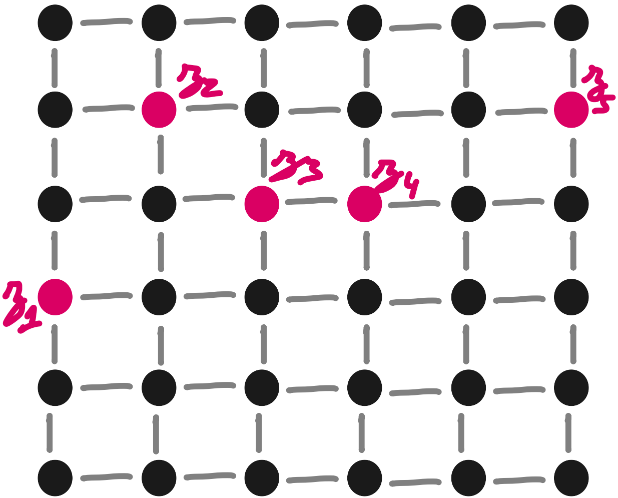
Examples



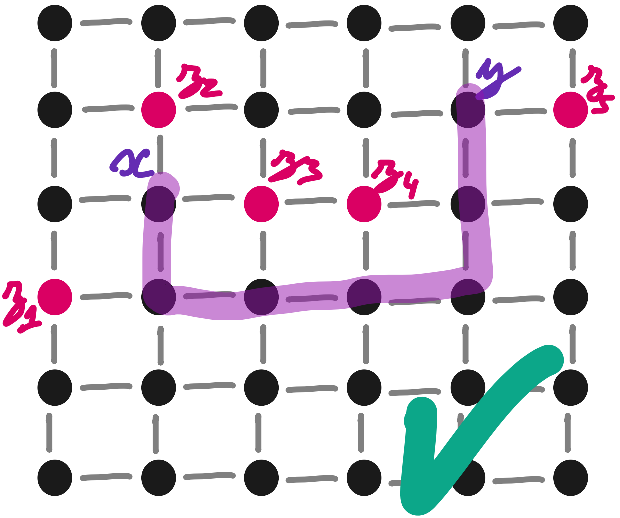
Examples



Examples



Examples



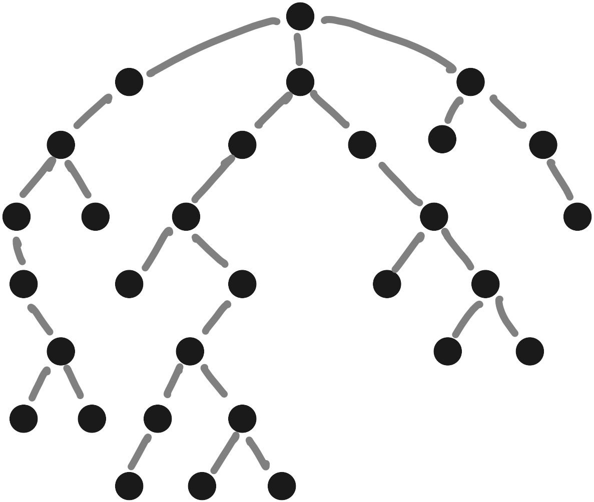
Computational complexity

	precomputation	query	update
Naive	$\mathcal{O}(1)$	$\mathcal{O}(\|G\|)$	$\mathcal{O}(1)$
Naive	$\mathcal{O}(\|G\|)$	$\mathcal{O}(1)$	$\mathcal{O}(\|G\|)$

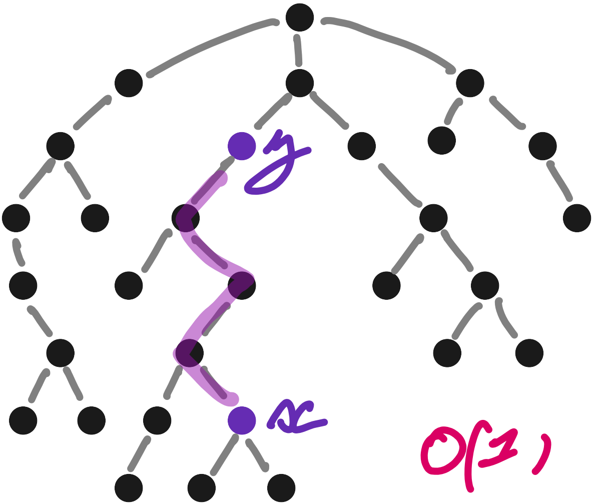
Computational complexity

	precomputation	query	update
Naive	$\mathcal{O}(1)$	$\mathcal{O}(\|G\|)$	$\mathcal{O}(1)$
Naive	$\mathcal{O}(\|G\|)$	$\mathcal{O}(1)$	$\mathcal{O}(\|G\|)$
Duan, Pettie '20	$\mathcal{O}(\lvert G\rvert \|G\| \cdot \log \lvert G\rvert)$	$\mathcal{O}(k)$	$\mathcal{O}(k^3 \log^3 \lvert G\rvert)$
Brand Saranurak '19 (randomized)	$\mathcal{O}(\lvert G\rvert^\omega)$	$\mathcal{O}(k^2)$	$\mathcal{O}(k^\omega)$
contribution 1	$2^{2^{\mathcal{O}(k)}} \cdot \lvert G\rvert^2 \cdot \|G\|$	$2^{2^{\mathcal{O}(k)}}$	$2^{2^{\mathcal{O}(k)}}$
contribution 2	$2^{\mathcal{O}(k \log k)} \cdot \lvert G\rvert^{\mathcal{O}(1)}$	$k^{\mathcal{O}(1)}$	$k^{\mathcal{O}(1)}$

Simple case: Trees

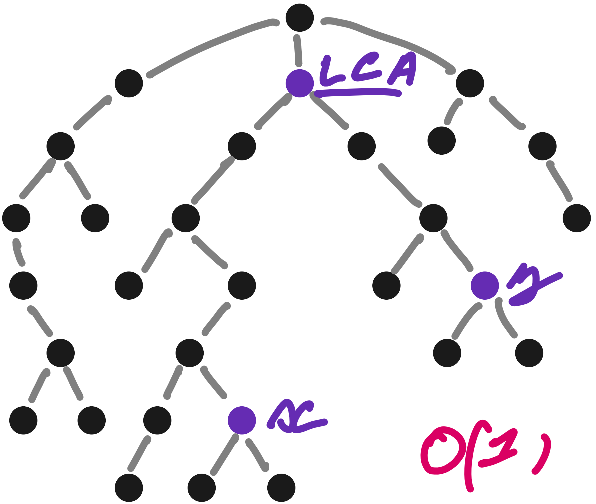


Simple case: Trees



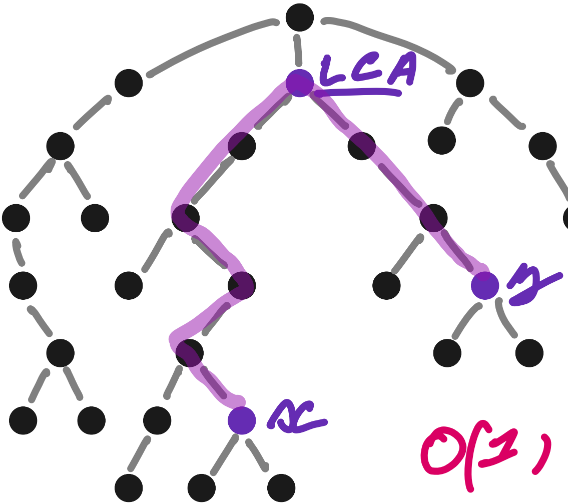
$O(1)$

Simple case: Trees



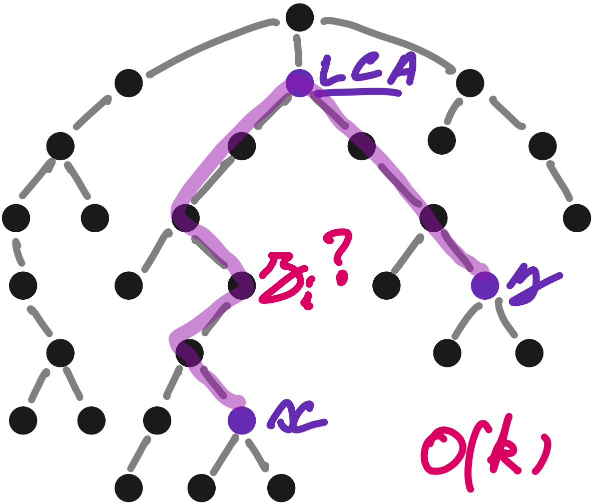
$O(1)$

Simple case: Trees



$O(1)$

Simple case: Trees



Well connected graphs

Def: A graph G is $k+1$ connected if for any $z_1, \ldots, z_k$

$$G \setminus \{z_1, \ldots, z_k\} \text{ is connected}$$

$\rightarrow$ Answering is trivial !

Well connected graphs

Def: A graph G is $k+1$ connected if for any $z_1, \ldots, z_k$

$$G \setminus \{z_1, \ldots, z_k\} \text{ is connected}$$

$\rightarrow$ Answering is trivial !

$\rightarrow$ Most graphs are not trees or k-connected

Unbreakable graphs

Def: A graph G is (q, k)-unbreakable if for all sets S with $|S| \leq k$:

for all separations A, B of $G \setminus S$ either $|A| \leq q$ or $|B| \leq q$

Intuition: If C is the biggest connected component (in $G \setminus S$)

Then $|G \setminus C| \leq q$

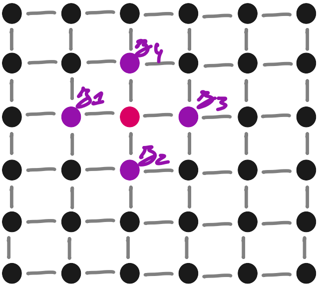
Unbreakable graphs

Def: A graph G is (q, k)-unbreakable if for all sets S with $|S| \leq k$:

for all separations A, B of $G \setminus S$ either $|A| \leq q$ or $|B| \leq q$

Intuition: If C is the biggest connected component (in $G \setminus S$)

Then $|G \setminus C| \leq q$



if $k = 4$ then $q \geq 5$

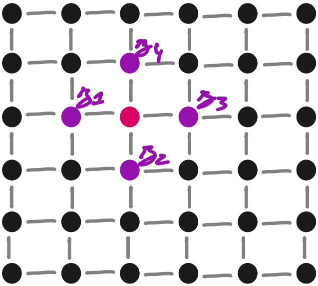
Unbreakable graphs

Def: A graph G is (q,k)-unbreakable if for all sets S with $|S| \leq k$:

for all separations A, B of $G \setminus S$ either $|A| \leq q$ or $|B| \leq q$

Intuition: If C is the biggest connected component (in $G \setminus S$)

Then $|G \setminus C| \leq q$

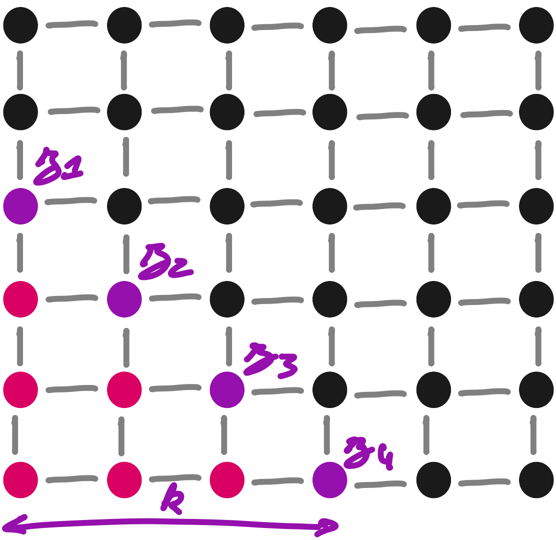


if $k = 4$ then $q \geq 5$

In general, if $k < n$ then $q =$ **?**

tinyurl.com/short-polls

n-n grids are (k^2, k)-unbreakable



What we do with conn_k and unbreakability

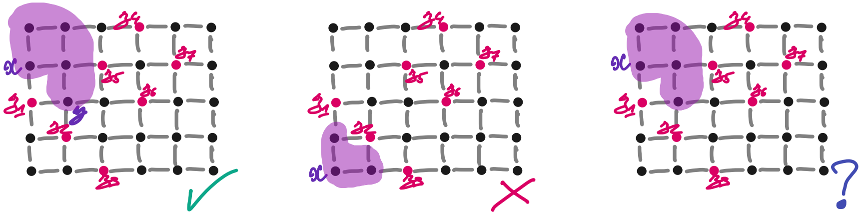
If G is (q,k)-unbreakable, then $\mathrm{conn}_k()$ solvable in time $\mathcal{O}(q+k)$.

Take $x, y, z_1, \ldots, z_k$:

→ Depth-first search in $G \setminus \{z_1, \ldots, z_k\}$ around x:

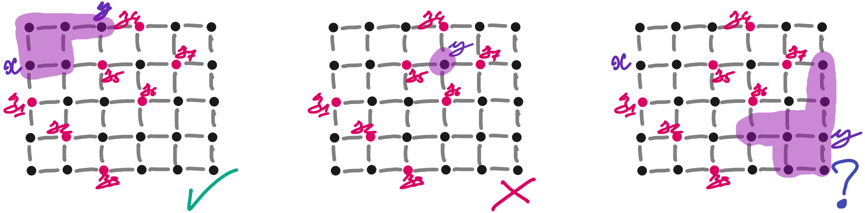
→ Stop after $q+1$ new vertices.

1. Found y?
2. Explored the whole component of x?
3. Then x is in "The big component".



What we do with conn_k and unbreakability

If G is (q, k)-unbreakable, then $\mathrm{conn}_k()$ solvable in time $\mathcal{O}(q + k)$.



→ Depth-first search in $G \setminus \{z_1, \ldots, z_k\}$ around y:

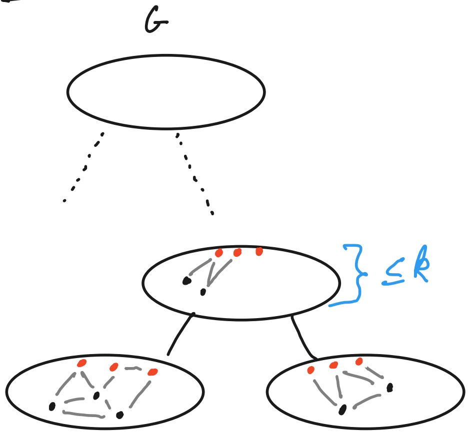
  → Stop after $q + 1$ new vertices.

  1. Found x?
  2. Explored the whole component of y?
  3. Then y is in "The big component".

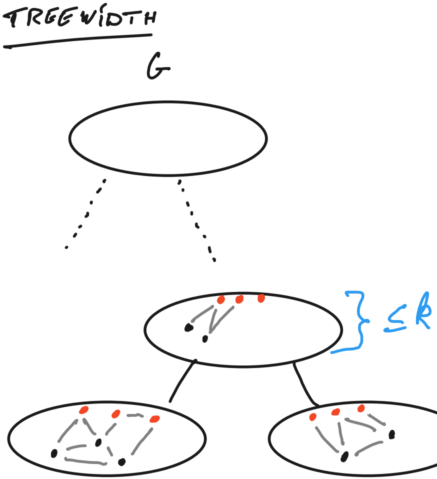
→ We can conclude $\mathrm{conn}_k(x, y, z_1, \ldots, z_k)$

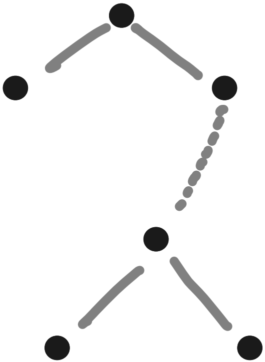
Tree-width

TREEWIDTH

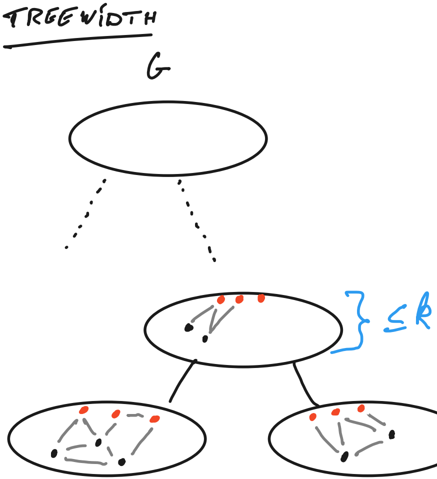


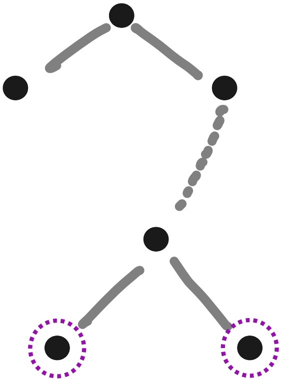
Tree-width



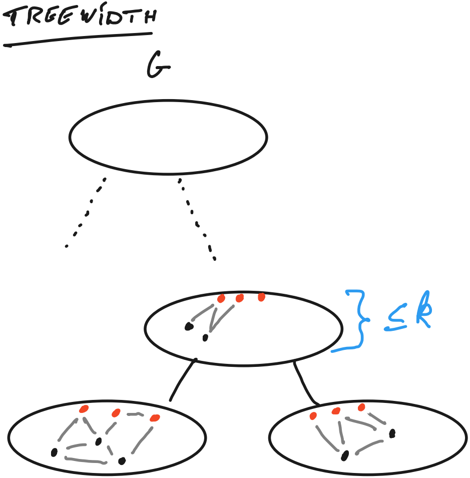


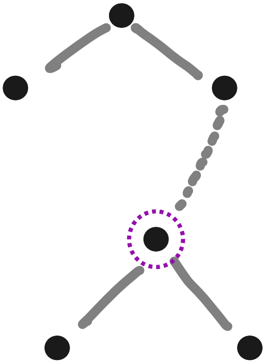
Tree-width



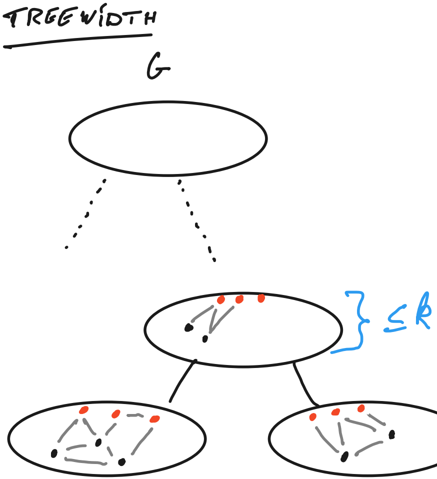


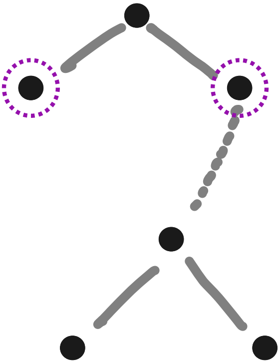
Tree-width



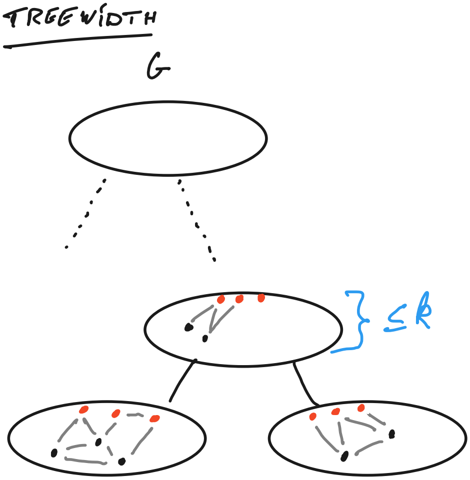


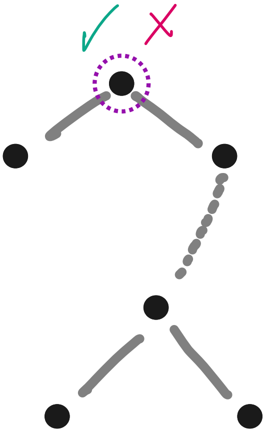
Tree-width





Tree-width





Unbreakable graphs

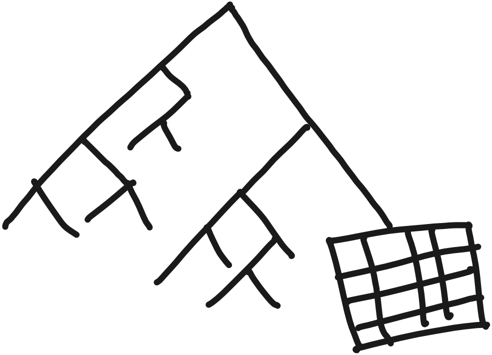
Cygan, Lokshtanov, Pilipczuk, Pilipczuk, Saurabh '19:

For every k, G there is a tree decomposition of G such that:

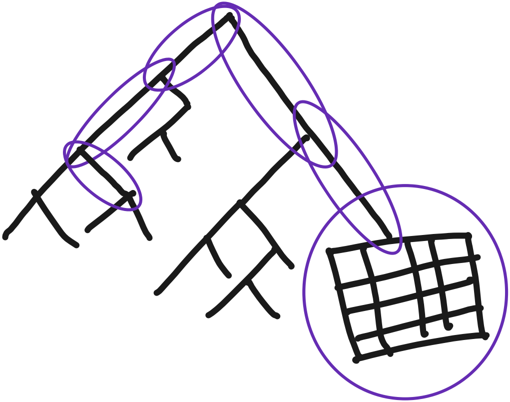
- $\rightarrow$ Every bag is (k, q)-unbreakable.
- $\rightarrow$ $q \in 2^{\mathcal{O}(k^2)}$.
- $\rightarrow$ Adjacent bags have intersection of size q.
- $\rightarrow$ Computable in time $2^{\mathcal{O}(k^2)}|G|\|G\|$.

Then perform **dynamic programming** on the tree decomposition.

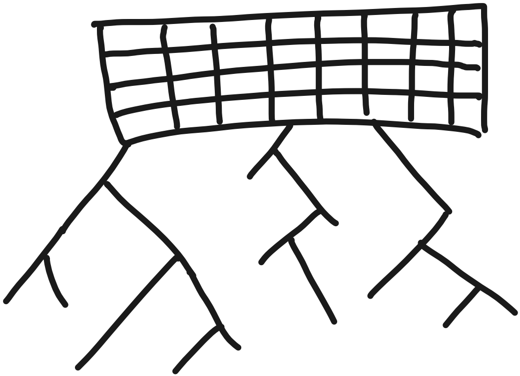
Examples of decompositions



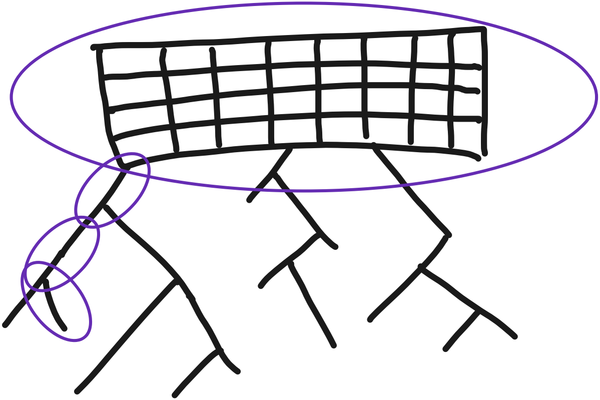
Examples of decompositions



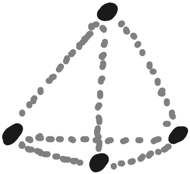
Examples of decompositions



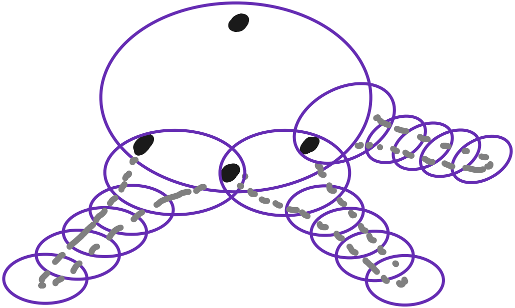
Examples of decompositions



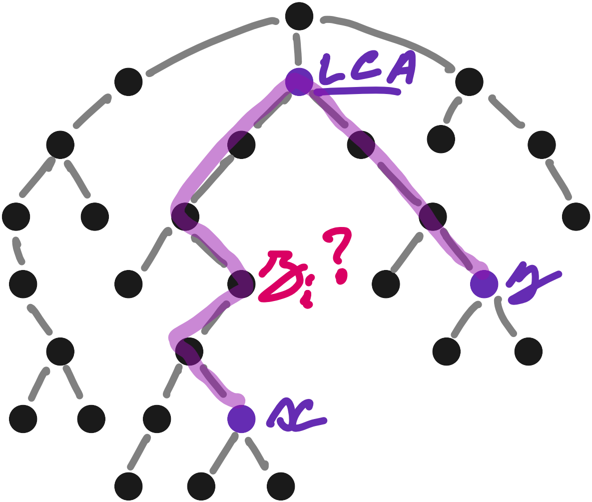
Examples of decompositions



Examples of decompositions



Back to the case of trees!



Adding Connectivity to FO

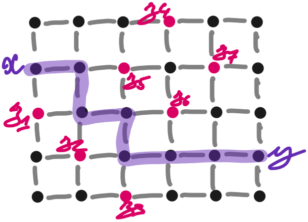
Schirrmacher, Siebertz, Vigny '21 and Bojanczyk '21

Syntax

→ Uses : FO and $\mathbf{conn}_k(x, y, z_1, \ldots, z_k)$

Meaning

→ x and y are connected after the deletion of $z_1, \ldots, z_k$.



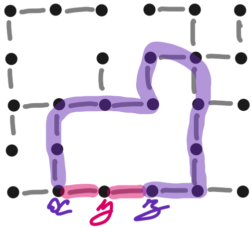
Expressive power of $\mathrm{FO} + \mathrm{conn}$

→ connectivity

$$\forall x \forall y \ \mathrm{conn}_0(x, y)$$

→ cycle

$$\varphi_{cycle} := \exists x \exists y \exists z \big(E(x, y) \wedge E(y, z) \wedge z \neq x \wedge \mathrm{conn}_1(z, x, y) \big)$$



→ Not expressible

planarity, bipartiteness, Hamiltonicity, . . .

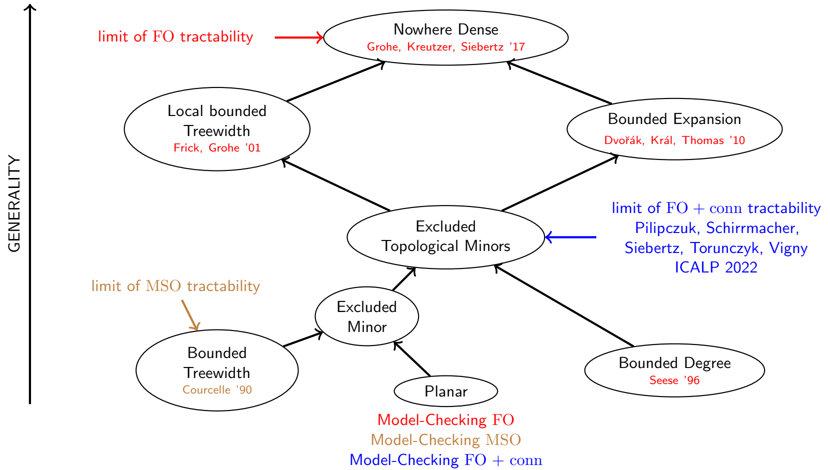
Main result

Theorem: Pilipczuk, Schirrmacher, Siebertz, Torunczyk, Vigny

→ Model-checking for properties in $\mathrm{FO} + \mathrm{conn}$ over graph classes excluding a topological minor is solvable in time FPT.

→ Model-checking is not FPT for more general graph classes.
Under complexity assumptions

Monotone graph classes



More logics: disjoint-paths logic $(\mathrm{FO} + \mathrm{DP})$

Syntax

$\rightarrow$ Uses : FO and **disjoint-paths**$_{k}(x_1, y_1, \ldots, x_k, y_k)$

Meaning

$\rightarrow$ x_i and y_i are connected by internally vertex-disjoint paths.

Expressive power of disjoint-paths logic (FO + DP)

→ connectivity

$$\forall x \forall y \ \mathrm{disjoint\text{-}paths}_1(x, y)$$

→ cycle

$$\exists x \exists y \exists z \ \mathrm{disjoint\text{-}paths}_3((x, y), (y, z), (z, x))$$

→ connectivity operators

$$\mathrm{conn}_k(x, y, z_1, \dots, z_k) := \mathrm{disjoint\text{-}paths}_{k+1}[(x, y), (z_1, z_1), \dots, (z_k, z_k)]$$

Disjoint-paths logic (FO + DP)

Expressible

- → connectivity
- → (topological) minors,
- → planarity, ...

Not expressible

- → bipartiteness

strict hierarchy

- → $\mathrm{FO} + \mathrm{DP}_1 \subsetneq \mathrm{FO} + \mathrm{DP}_2 \subsetneq \dots$

Model checking?

- → For planar graphs?
- → For excluded minor?

New results

Theorem: Schirrmacher, Siebertz, Stamoulis, Thilikos, Vigny

$\rightarrow$ Model-checking for properties in $\mathrm{FO} + \mathrm{DP}$ over graph classes excluding a topological minor is solvable in time FPT.

$\rightarrow$ Model-checking is not FPT for more general graph classes.
Under complexity assumptions

Thank you very much for your attention!