Elimination distance to bounded degree on planar graphs

Alexander Lindermayr[1], Sebastian Siebertz[1] and **Alexandre Vigny**[1]

[1]Universität Bremen, Germany

MFCS, August 24-28, 2020

Outline

Trivial graphs classes?

Graph problems: Hamilton path, FO / MSO model checking, graph isomorphism,...

Graph classes: edgeless graphs, planar graphs, trees,...

Some hard problems are simple for some graph classes.

Running example

Graph isomorphism problem & Graphs with bounded degree

E.M. Luks (1982): XP algorithm for graph isomorphism
(parametrized by the degree d)

Input :	Graphs G, H.
Goal :	Are G and H isomorphic?
Running time :	$O(\lvert G\rvert^{f(d)})$
Parametrized by d:	the degree of G and H.

With a small twist

Two graphs with only one node of degree $> d$.

The algorithm can be adapted.

Step 1: Color the neighborhood of the high degree vertex.

Step 2: Remove the high degree vertex.

Step 3: Use the previous algorithm.

Deletion distance

J. Guo, F. Hüffner, R. Niedermeier (2004): Distance from Triviality.

G at *deletion distance k* from $\mathscr{C}$ iff $G - \{a_1, \ldots, a_k\} \in \mathscr{C}$

Deletion distance

J. Guo, F. Hüffner, R. Niedermeier (2004): Distance from Triviality.

G at *deletion distance* k from $\mathscr{C}$ iff $G - \{a_1, \ldots, a_k\} \in \mathscr{C}$

J. Bulian, A. Dawar (2016): FPT algorithm for graph isomorphism
(parametrized by deletion distance to degree d)

Input :	Graphs G, H, integer d.
Goal :	Are G and H isomorphic?
Running time :	$O(f(k,d) \cdot \lvert G \rvert^{g(d)})$
Parametrized by k:	the deletion distance (of G) to degree d.

---

k is computable in time $O(f(k,d) \cdot \lvert G \rvert)$.

Elimination distance

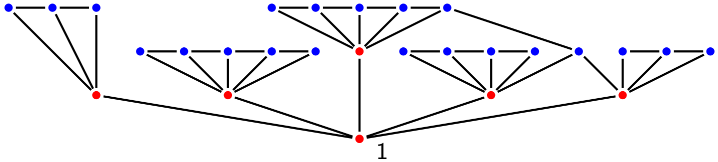
J. Bulian, A. Dawar (2016): Elimination distance to $\mathscr{C}$.

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

Example

$\mathscr{C}_d$: all graphs of degree at most d.

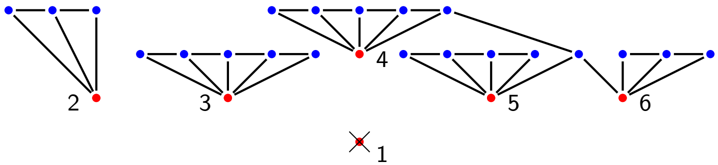
$$\mathrm{ed}_{\mathscr{C}_3}(G) = ??$$



Example

$\mathscr{C}_d$: all graphs of degree at most d.

$$\mathrm{ed}_{\mathscr{C}_3}(G) = ??$$

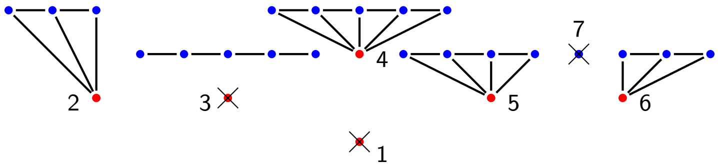


Round 1) : [1]

Example

$\mathscr{C}_d$: all graphs of degree at most d.

$$\mathrm{ed}_{\mathscr{C}_3}(G) = ??$$



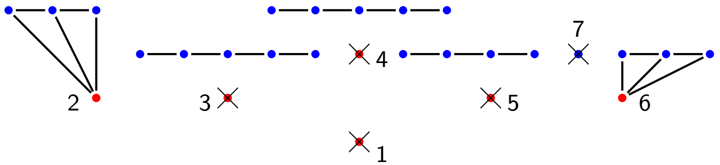
Round 1) : [1]

Round 2) : [3,7]

Example

$\mathscr{C}_d$: all graphs of degree at most d.

$$\mathrm{ed}_{\mathscr{C}_3}(G) = ??$$



Round 1) : [1]

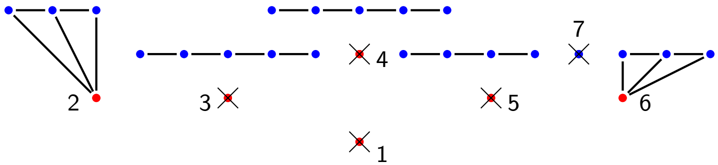
Round 2) : [3,7]

Round 3) : [4,5]

Example

$\mathscr{C}_d$: all graphs of degree at most d.

$$\mathrm{ed}_{\mathscr{C}_3}(G) = ??$$



Round 1) : [1]
Round 2) : [3,7]
Round 3) : [4,5]

Round 1) : [7]
Round 2) : [1]
Round 3) : [3,4,5]

$\cdots$

Bounded tree depth

Elimination distance is inspired from tree depth:

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

Tree depth k = Elimination distance k to the edgeless graph.

Graph isomorphism & Graphs with bounded degree

J. Bulian, A. Dawar (2016): FPT algorithm for graph isomorphism
(parametrized by elimination distance to degree d)

Input :	Graphs G, H, integer d.
Goal :	Are G and H isomorphic?
Running time :	$O(f(k,d) \cdot \lvert G\rvert^{g(d)})$
Parametrized by k :	the elimination distance (of G) to degree d.

Questions

When is $\mathrm{ed}_{\mathscr{C}}(G)$ easily computable ?

Restriction on $\mathscr{C}$: Edgeless graphs, Graph with bounded degree,...

Restriction on G : Parametrized by the tree width,

Parametrized by the size of an excluded minor,

Restricted to planar graph, ...

Our result

A. Lindermayr, S. Siebertz, A. Vigny:

Elimination distance to degree d is FPT over K_5-minor-free graphs.

Input :	Graph G, integers k, d.
Goal :	Is G at elimination distance k to degree d?
Restriction :	G exclude K_5 as a minor.
Running time :	$O(f(k,d) \cdot \|G\|^c)$

Tools

We use:

→ Simple combinatoric

→ MSO expressibility

→ Courcelle's Theorem

→ Grid Theorem

→ Irrelevant vertex technique

Expressible in MSO

"$\mathrm{ed}_{\mathscr{C}_d}(G) = k$" :

$$
\begin{aligned}
&\forall H_1 \preceq G, \quad \exists a_1 \\
&\quad \forall H_2 \preceq (H_1 - a_1), \quad \exists a_2 \\
&\qquad \forall H_3 \preceq (H_2 - \{a_1, a_2\}), \quad \exists a_3 \\
&\qquad\qquad \ddots \\
&\qquad\qquad\quad \forall H \preceq (H_k - \{a_1, \dots, a_k\}), \quad \deg(H) \leq d
\end{aligned}
$$

If $\mathscr{C}$ is MSO definable, then "$\mathrm{ed}_{\mathscr{C}}(G) = k$" is also MSO definable.

Courcelle's Theorem

B. Courcelle (1990):
Model checking of MSO formulas is FPT for bounded tree width graphs.

Input :	Graph G, formula ϕ.
Goal :	Does $G \models \phi$?
Running time :	$O(f(\|\phi\|, k) \cdot \|G\|)$.
Parametrized by k:	the tree width of G.

Grid Theorem

N. Robertson, P. D. Seymour (1986):
Every graph has either a "small enough" tree width,
or a "big enough" grid-minor.

Input :	Graph G, integer k.
Output :	Either a tree decomposition of width $O(g(k))$
	Or a $k \times k$ grid minor
Running time :	$O(f(k) \cdot \lvert G \rvert^c)$.

Proof in a special case

Special case: Graph with degree $k + d$.

In the full proof:

- → $k + d < \deg(a)$
- → $d < \deg(a) \leq k + d$
- → $\deg(a) \leq d$

Here:

- → $d < \deg(a) \leq k + d$ (red nodes)
- → $\deg(a) \leq d$ (blue nodes)

A little bit of combinatorics

How many red nodes can there be?

1 round of elimination $\rightarrow$ $k+d$ connected components.

k rounds of elimination $\rightarrow$ affect $(k+d)^{2(k+d)}$ nodes.

There are at most $r=(k+d)^{2(k+d)}$ nodes of degree $>d$.

More than r red nodes $\rightarrow$ we have $\mathrm{ed}_{\mathscr{C}_d}(G)>k$.
Otherwise we continue.

Using the grid theorem

We have two cases:

→ Tree decomposition of width $g(r)$.

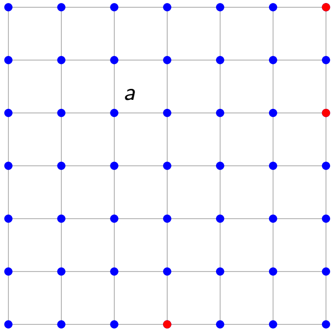
Courcelle's Theorem : $O(f(\phi, g(r)) \cdot |G|)$

$O(f(k,d) \cdot |G|)$

→ Grid minor of size $r \times r$.

Find an irrelevant vertex.

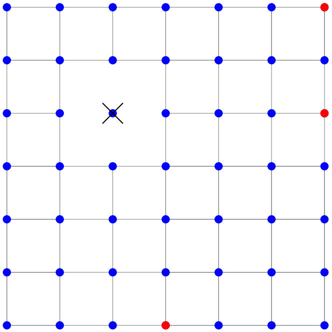
Large grid minor?



If G has a large grid minor,
we can find an irrelevant vertex.

Vertex a is solution irrelevant:
$\mathrm{ed}_{\mathscr{C}_d}(G) \leq k \Leftrightarrow \mathrm{ed}_{\mathscr{C}_d}(G - \{a\}) \leq k$

Large grid minor?

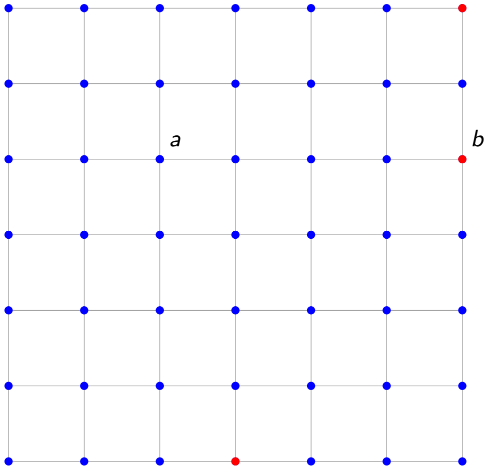


If G has a large grid minor,
we can find an irrelevant vertex.

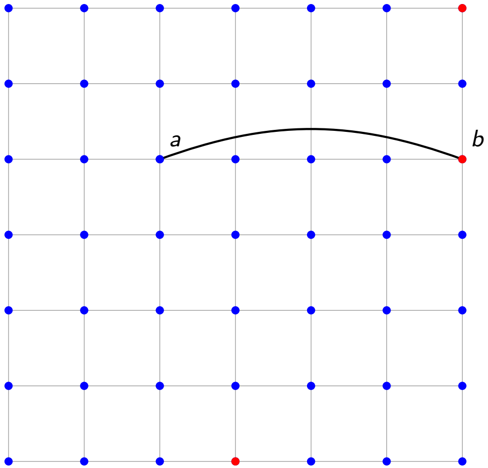
Vertex a is solution irrelevant:
$\mathrm{ed}_{\mathscr{C}_d}(G) \leq k \Leftrightarrow \mathrm{ed}_{\mathscr{C}_d}(G - \{a\}) \leq k$

In our case a vertex
is irrelevant if it is "far enough"
from nodes with high degree.

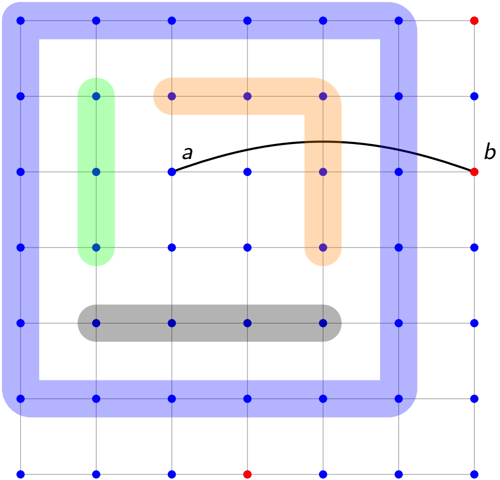
Excluding K_5



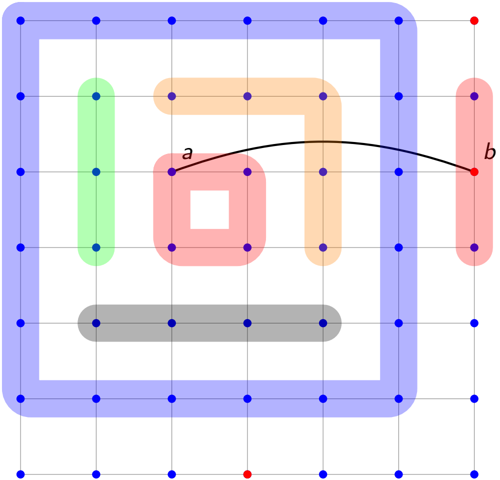
Excluding K_5



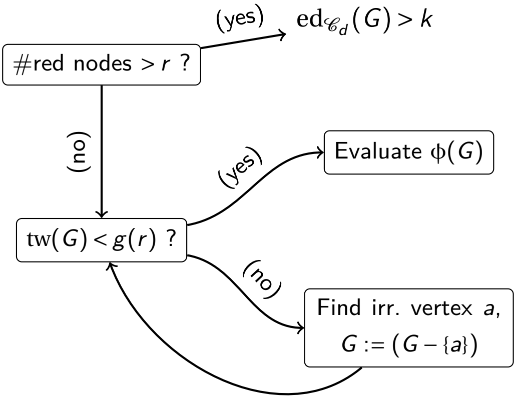
Excluding K_5



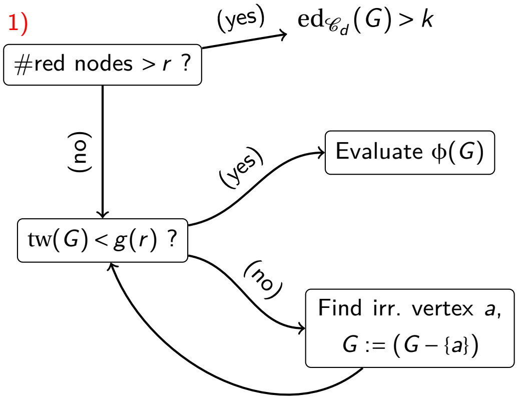
Excluding K_5



Overall picture / The irrelevant vertex technique

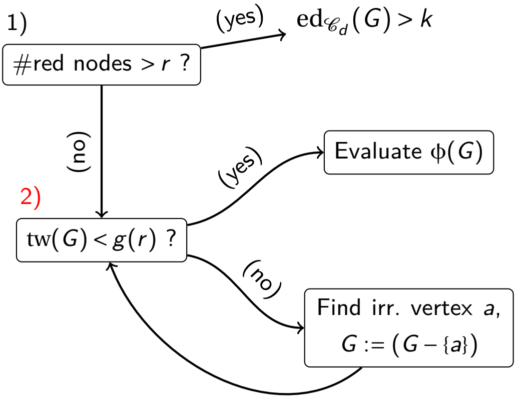


Overall picture / The irrelevant vertex technique



1) Simple combinatorics

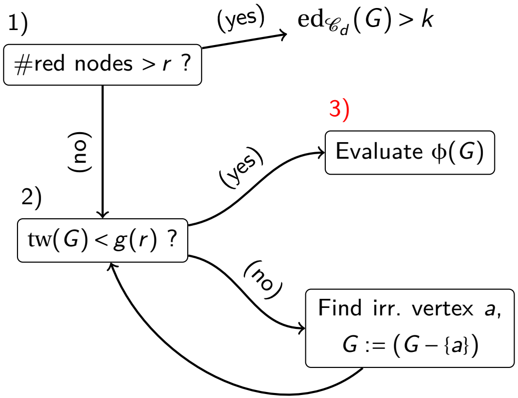
Overall picture / The irrelevant vertex technique



1) Simple combinatorics

2) Grid Theorem

Overall picture / The irrelevant vertex technique

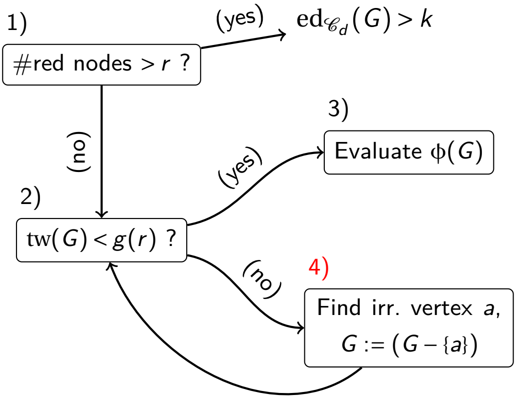


1) Simple combinatorics

2) Grid Theorem

3) Courcelle's Theorem

Overall picture / The irrelevant vertex technique



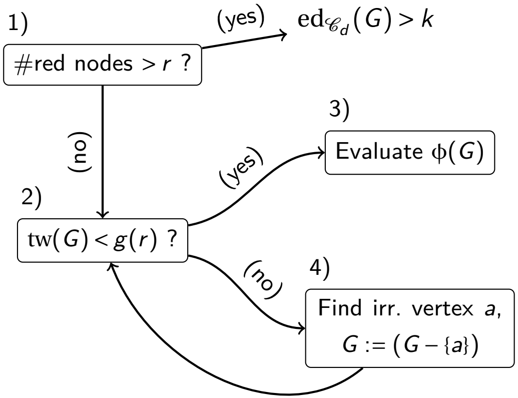
1) Simple combinatorics

2) Grid Theorem

3) Courcelle's Theorem

4) Find and remove $\{a\}$

Overall picture / The irrelevant vertex technique



1) Simple combinatorics

2) Grid Theorem

3) Courcelle's Theorem

4) Find and remove $\{a\}$

Total: $O(f(k,d) \cdot |G|^c)$

Future work

Open problem 1

FPT algorithm for "$\mathrm{ed}_{\mathscr{C}_d}(G) = k$" for any graph G.

Open problem 2

FPT algorithm for "$\mathrm{ed}_{\mathscr{C}_d}(G) = k$" for any graph G of degree $k + d$.

Conjecture

Algorithm for open problem 2) → algorithm for open problem 1).

Future work

Open problem 1

FPT algorithm for "$\mathrm{ed}_{\mathscr{C}_d}(G) = k$" for any graph G.

Open problem 2

FPT algorithm for "$\mathrm{ed}_{\mathscr{C}_d}(G) = k$" for any graph G of degree $k + d$.

Conjecture

Algorithm for open problem 2) → algorithm for open problem 1).

Thank you!