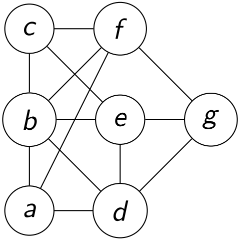
Twin-width

Édouard Bonnet

ENS Lyon, LIP

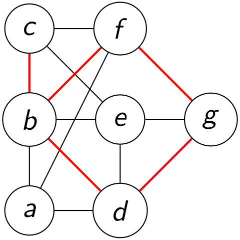
September 10th, 2021, IPEC Tutorial

Graphs



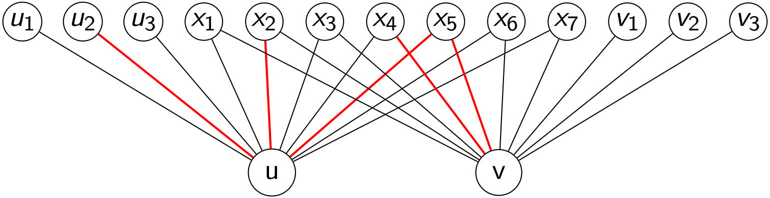
Two outcomes between a pair of vertices:
edge or non-edge

Trigraphs



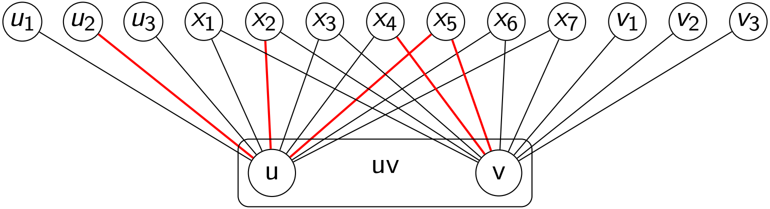
Three outcomes between a pair of vertices:
edge, or non-edge, or red edge (error edge)

Contractions in trigraphs



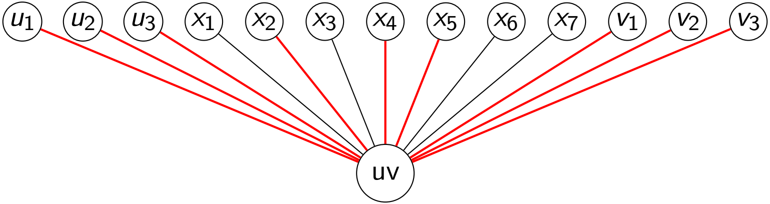
Identification of two non-necessarily adjacent vertices

Contractions in trigraphs



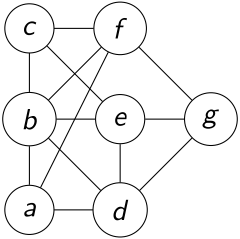
Identification of two non-necessarily adjacent vertices

Contractions in trigraphs



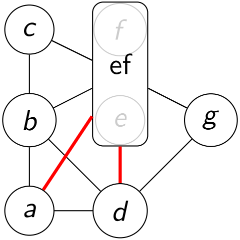
edges to $N(u) \triangle N(v)$ turn red, for $N(u) \cap N(v)$ red is absorbing

Contraction sequence



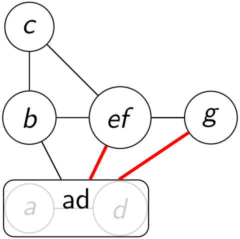
A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \ldots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.

Contraction sequence



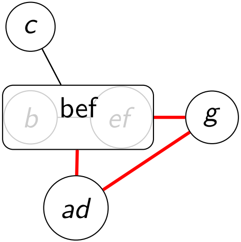
A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \ldots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.

Contraction sequence



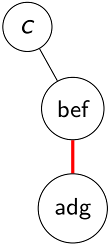
A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \ldots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.

Contraction sequence



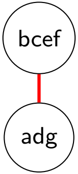
A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \ldots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.

Contraction sequence



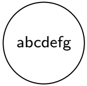
A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \ldots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.

Contraction sequence



A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \ldots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.

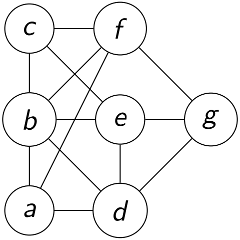
Contraction sequence



A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \ldots, G_2, G_1$ such that
G_i is obtained by performing one contraction in G_{i+1}.

Twin-width

tww(G): Least integer d such that G admits a contraction sequence where all trigraphs have *maximum red degree* at most d.

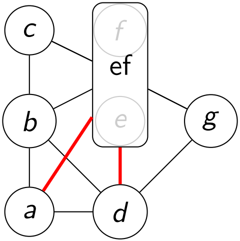


Maximum red degree $= 0$

overall maximum red degree = 0

Twin-width

tww(G): Least integer d such that G admits a contraction sequence where all trigraphs have *maximum red degree* at most d.

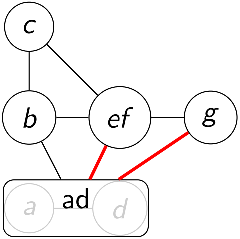


Maximum red degree = 2

overall maximum red degree = 2

Twin-width

tww(G): Least integer d such that G admits a contraction sequence where all trigraphs have *maximum red degree* at most d.

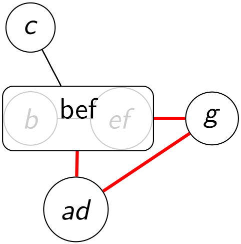


Maximum red degree = 2

overall maximum red degree = 2

Twin-width

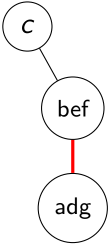
tww(G): Least integer d such that G admits a contraction sequence where all trigraphs have *maximum red degree* at most d.



Maximum red degree = 2
overall maximum red degree = 2

Twin-width

tww(G): Least integer d such that G admits a contraction sequence where all trigraphs have *maximum red degree* at most d.

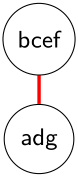


Maximum red degree $= 1$

overall maximum red degree = 2

Twin-width

tww(G): Least integer d such that G admits a contraction sequence where all trigraphs have *maximum red degree* at most d.

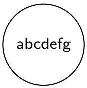


Maximum red degree $= 1$

overall maximum red degree = 2

Twin-width

tww(G): Least integer d such that G admits a contraction sequence where all trigraphs have *maximum red degree* at most d.



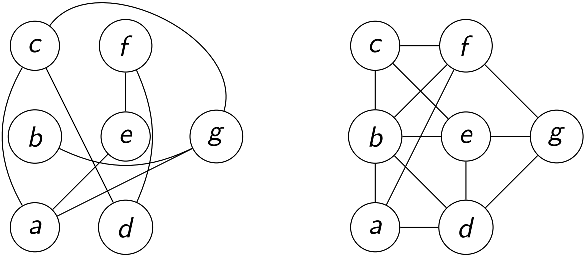
Maximum red degree $= 0$

overall maximum red degree = 2

Simple operations preserving small twin-width

- complementation: remains the same
- taking induced subgraphs: may only decrease
- adding one vertex linked arbitrarily: at most "doubles"
- substitution, lexicographic product: max of the twin-widths

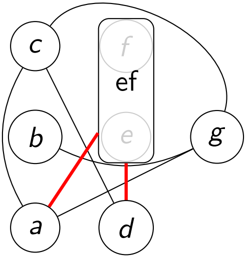
Complementation



$\overline{G}$ $\qquad$ G

$$\mathrm{tww}(\overline{G}) = \mathrm{tww}(G)$$

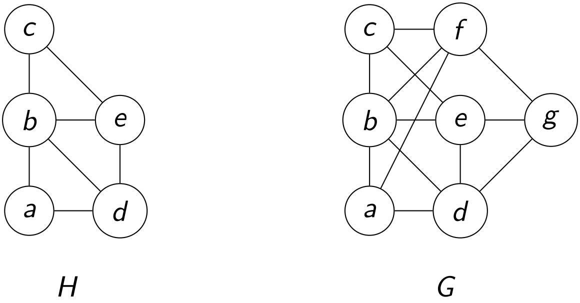
Complementation



$\overline{G_6}$ G_6

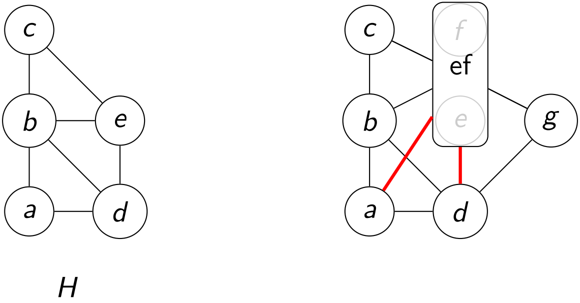
$$\text{tww}(\overline{G}) = \text{tww}(G)$$

Induced subgraph



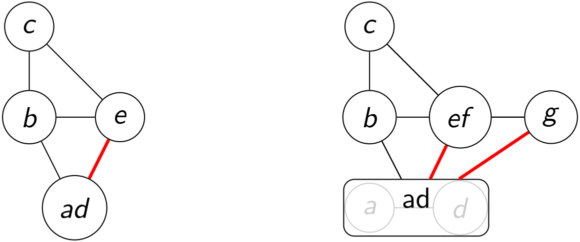
$\mathrm{tww}(H) \leqslant \mathrm{tww}(G)$

Induced subgraph



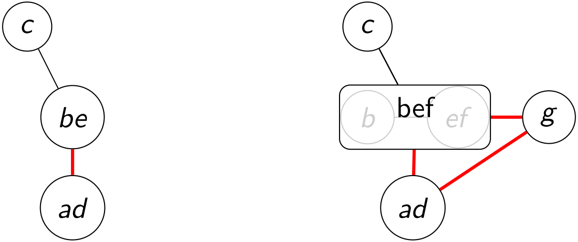
Ignore absent vertices

Induced subgraph



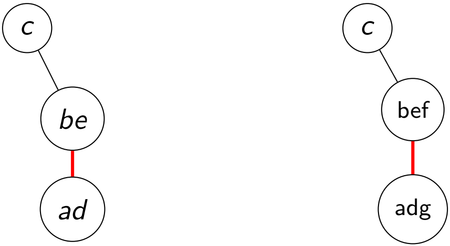
Mimic the contractions otherwise

Induced subgraph



Mimic the contractions otherwise

Induced subgraph



Mimic the contractions otherwise

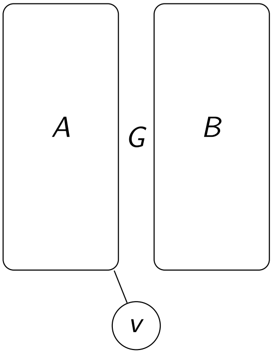
Induced subgraph

Mimic the contractions otherwise

Induced subgraph

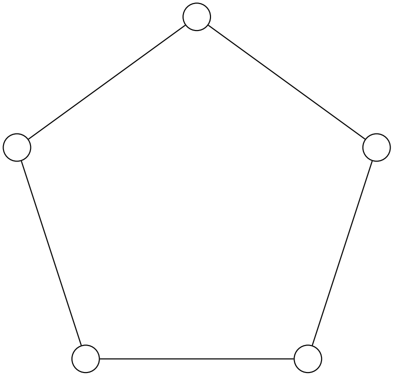
Mimic the contractions otherwise

Adding one apex v



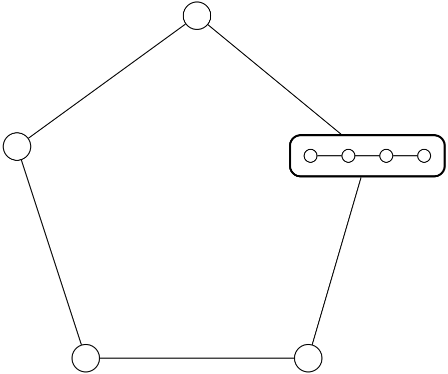
Ignore the contractions of $X \subseteq A$ with $Y \subseteq B$

Substitution and lexicographic product



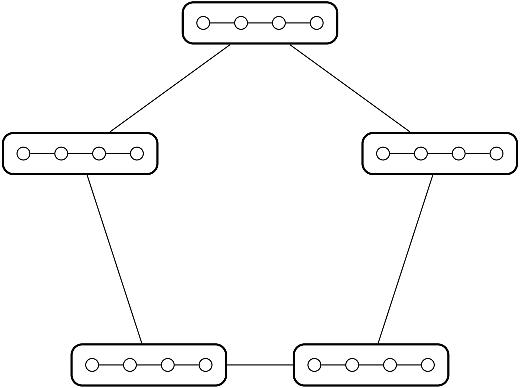
$G = C_5$

Substitution and lexicographic product



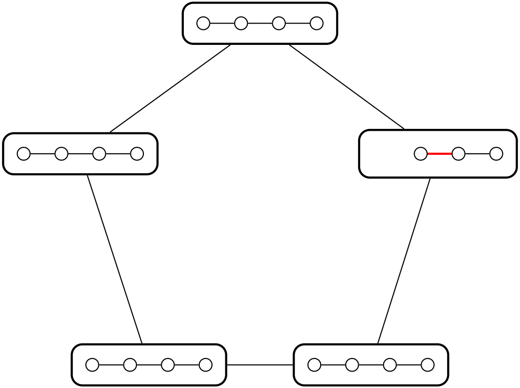
$G = C_5$, $H = P_4$, substitution $G[v \leftarrow H]$

Substitution and lexicographic product



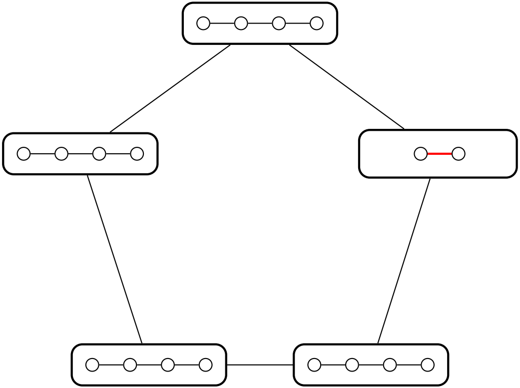
$G = C_5$, $H = P_4$, lexicographic product $G[H]$

Substitution and lexicographic product



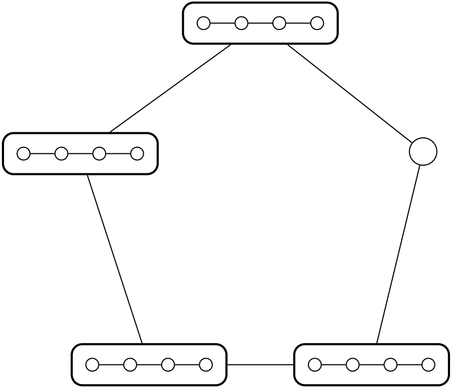
More generally any modular decomposition

Substitution and lexicographic product



More generally any modular decomposition

Substitution and lexicographic product

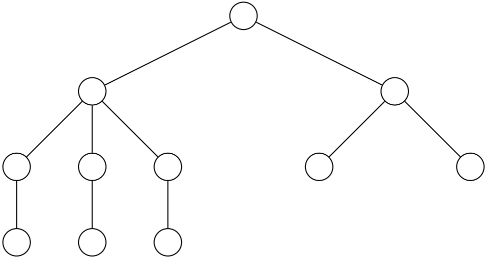


$$\text{tww}(G[H]) = \max(\text{tww}(G), \text{tww}(H))$$

Classes with bounded twin-width

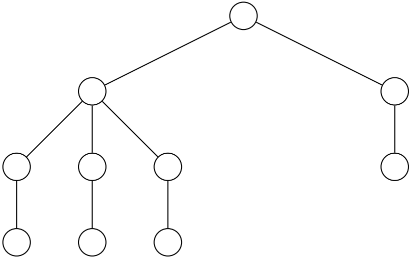
- cographs = twin-width 0
- trees, bounded treewidth, clique-width/rank-width
- grids
- ...

Trees



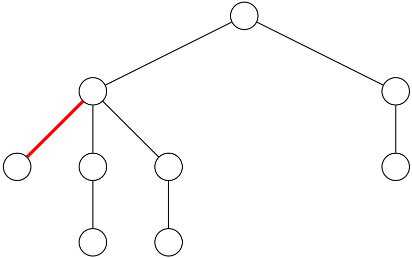
If possible, contract two twin leaves

Trees



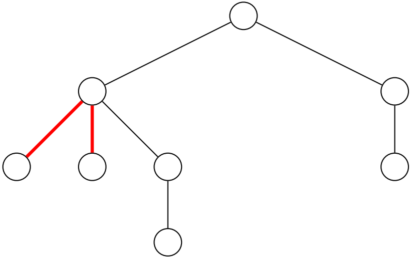
If not, contract a deepest leaf with its parent

Trees



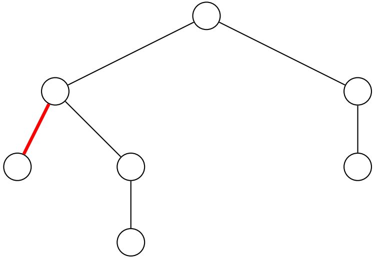
If not, contract a deepest leaf with its parent

Trees



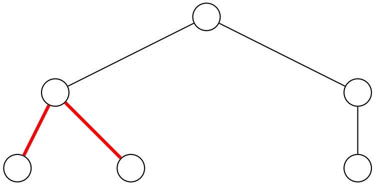
If possible, contract two twin leaves

Trees



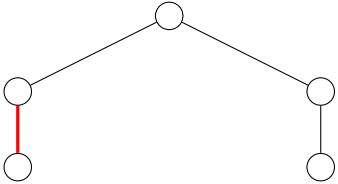
Cannot create a red degree-3 vertex

Trees



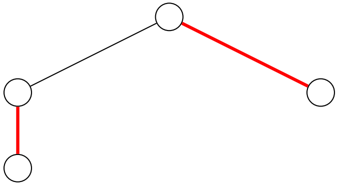
Cannot create a red degree-3 vertex

Trees



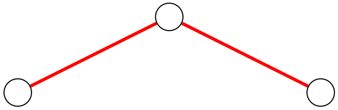
Cannot create a red degree-3 vertex

Trees



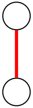
Cannot create a red degree-3 vertex

Trees



Cannot create a red degree-3 vertex

Trees

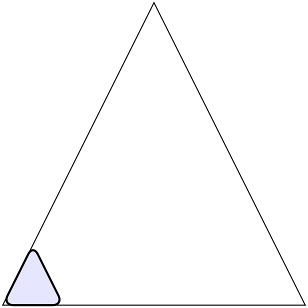


Cannot create a red degree-3 vertex

Trees

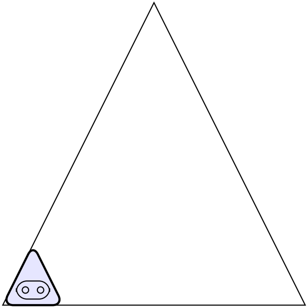
Cannot create a red degree-3 vertex

Bounded rank-width graphs



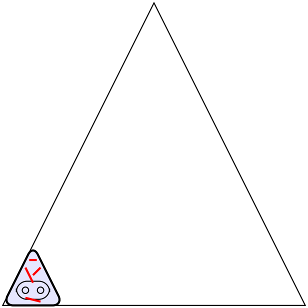
Generalization to bounded *rank-width*

Bounded rank-width graphs



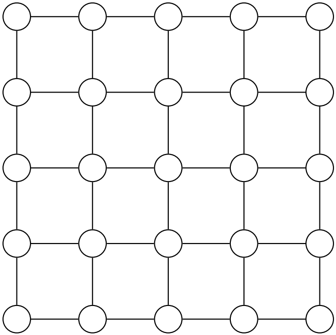
Two near-twins in a small subtree → contraction

Bounded rank-width graphs

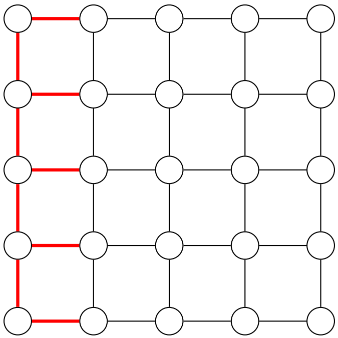


Red edges cluster in bounded size components

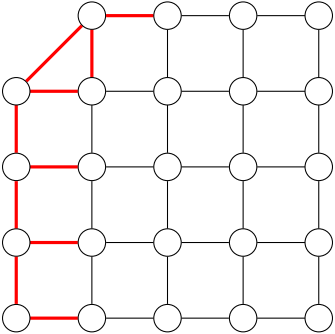
Grids



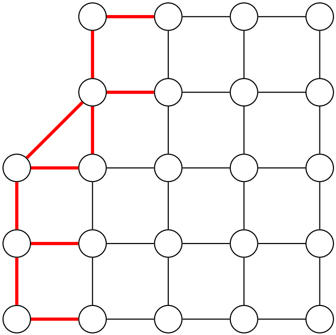
Grids



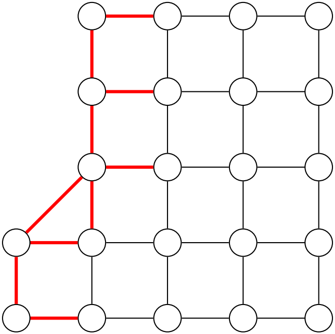
Grids



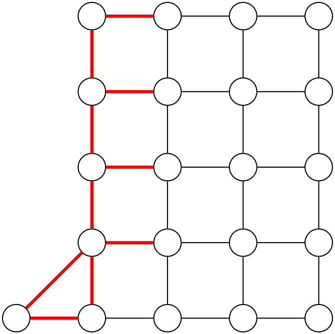
Grids



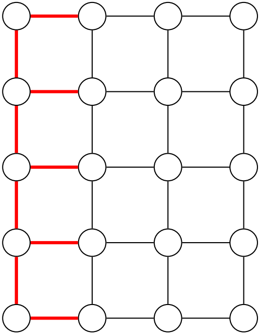
Grids



Grids

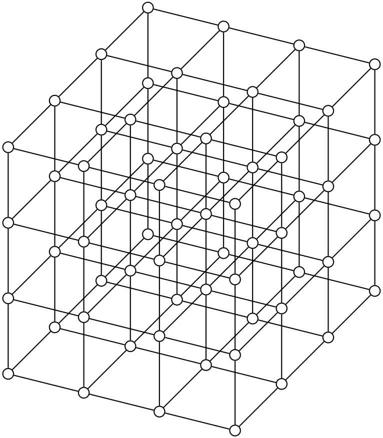


Grids



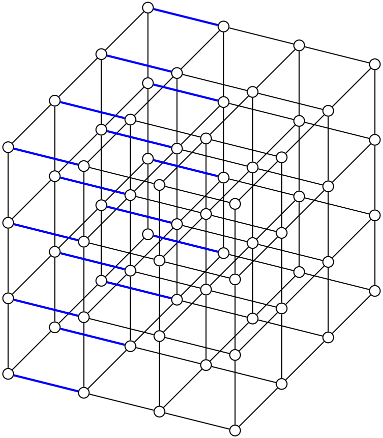
4-sequence for planar grids

3-dimensional grids



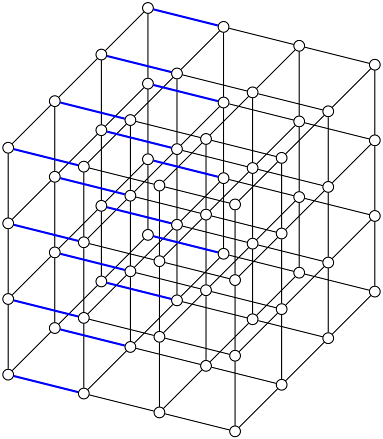
Contains arbitrary large clique minors

3-dimensional grids



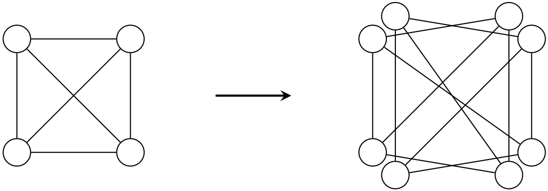
Contract the blue edges in any order → 12-sequence

3-dimensional grids



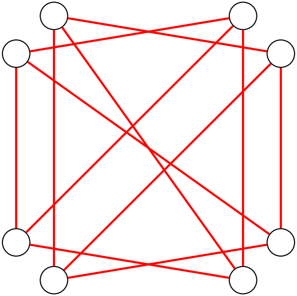
The d-dimensional grid has twin-width $\leqslant 4d$ (even $3d$)

2-lifts, expanders with bounded twin-width



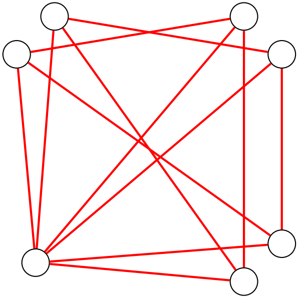
split each vertex in 2, replace each edge by 1 of the 2 matchings

2-lifts, expanders with bounded twin-width



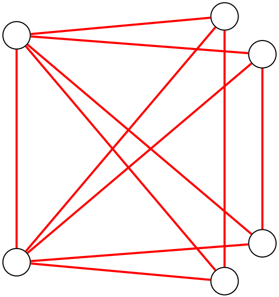
Iterated 2-lifts of K_4 have twin-width at most 6

2-lifts, expanders with bounded twin-width



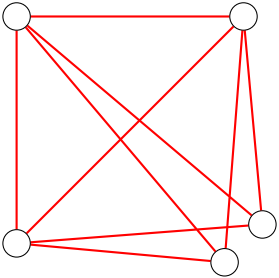
Iterated 2-lifts of K_4 have twin-width at most 6

2-lifts, expanders with bounded twin-width



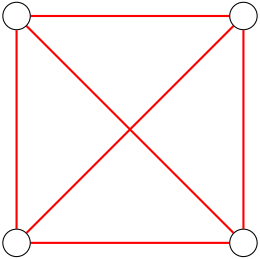
Iterated 2-lifts of K_4 have twin-width at most 6

2-lifts, expanders with bounded twin-width



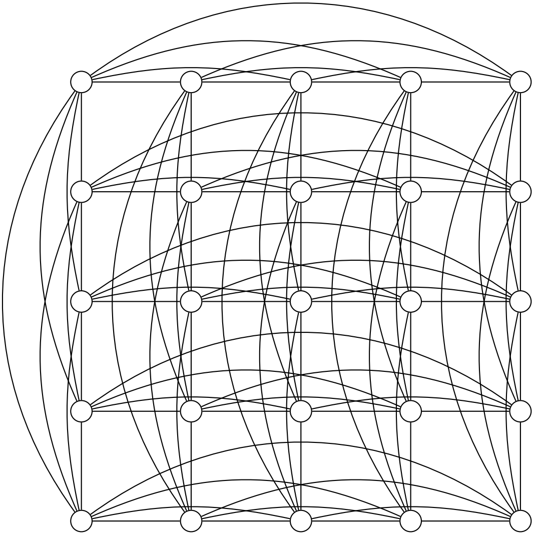
Iterated 2-lifts of K_4 have twin-width at most 6

2-lifts, expanders with bounded twin-width



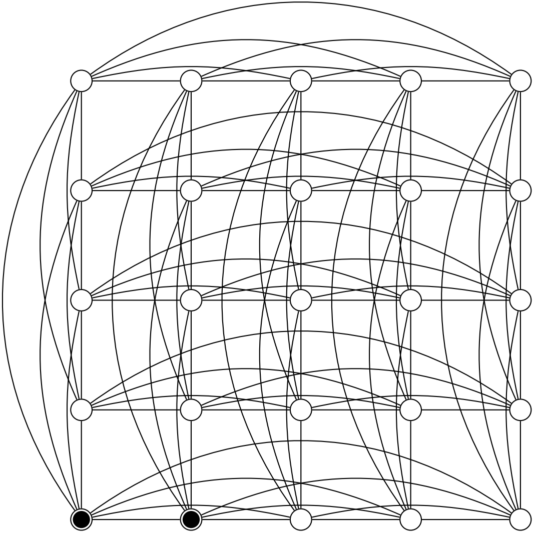
Iterated 2-lifts of K_4 have twin-width at most 6
but no balanced separators of size $o(n)$

First example of unbounded twin-width



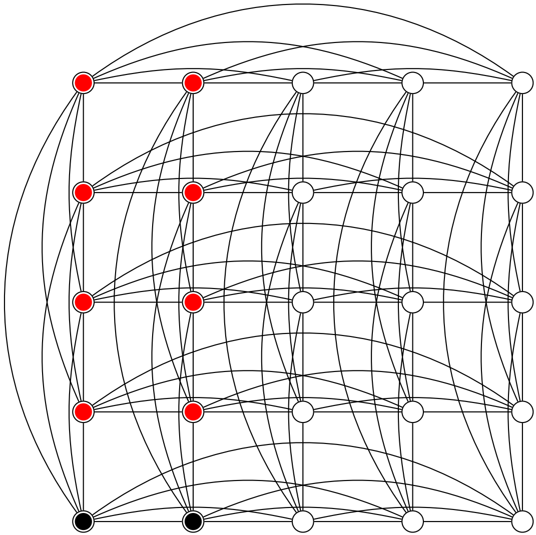
Line graph of a biclique a.k.a. rook graph

First example of unbounded twin-width



No pair of near twins

First example of unbounded twin-width



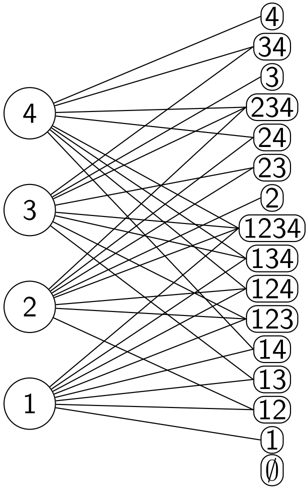
No pair of near twins

Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.

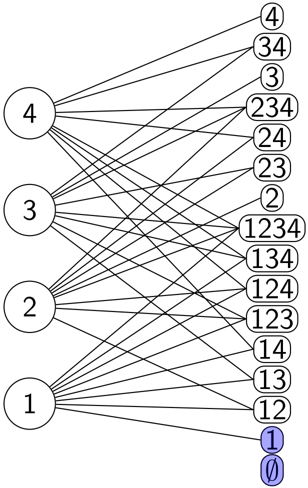
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



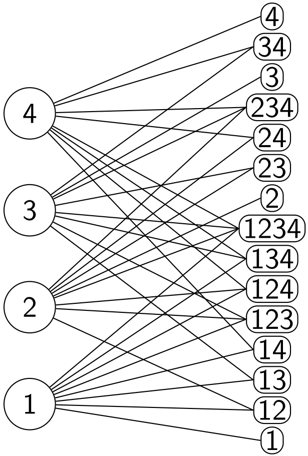
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



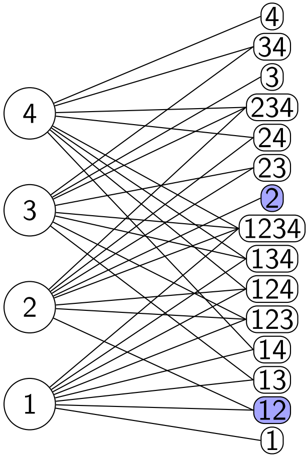
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



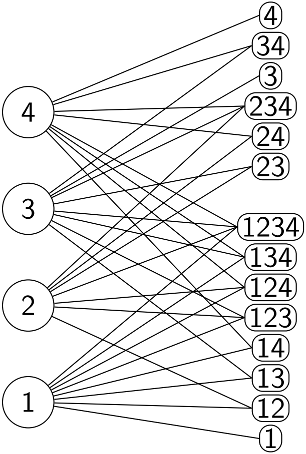
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



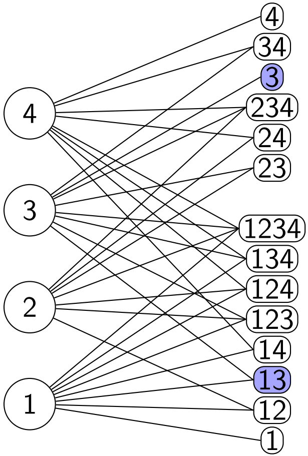
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



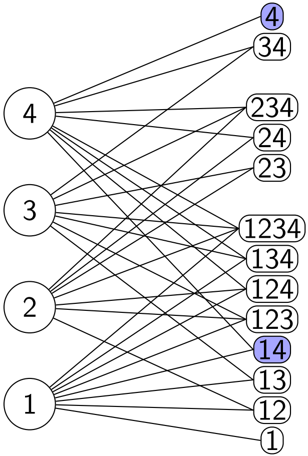
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



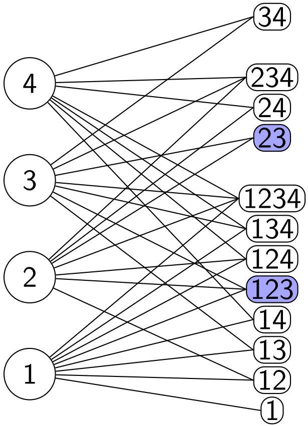
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



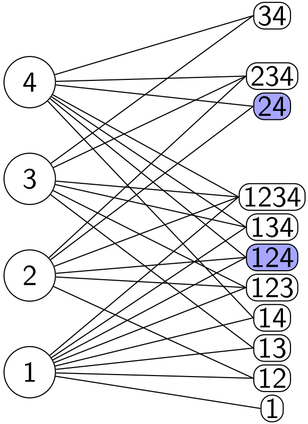
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



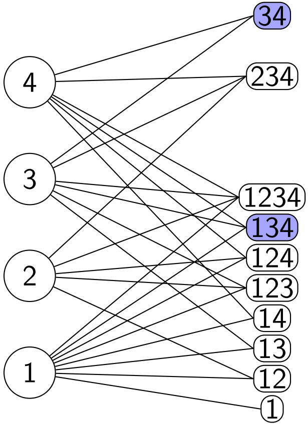
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



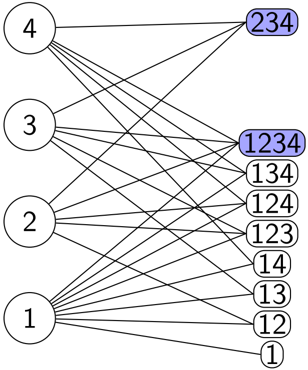
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



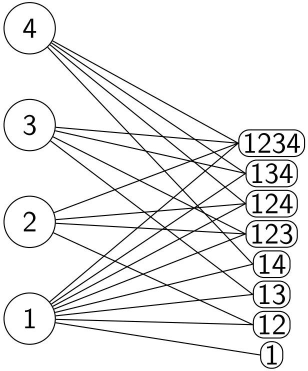
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



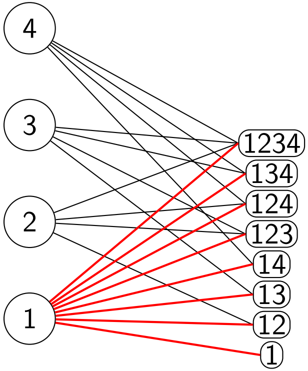
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



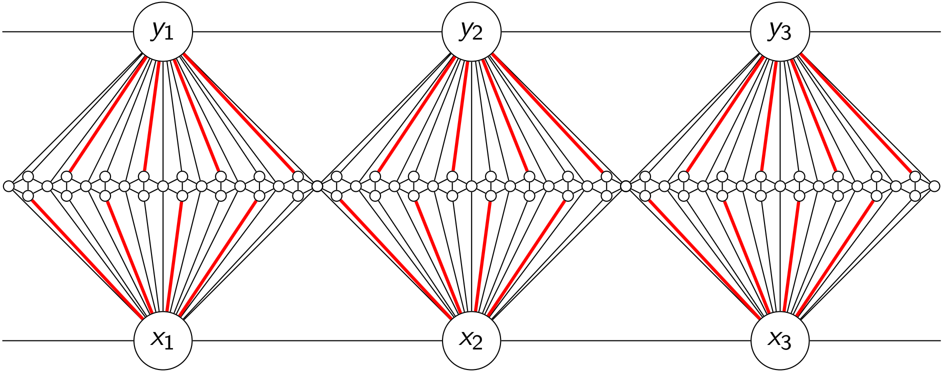
Universal bipartite graph

No $O(1)$-contraction sequence:
twin-width is *not* an iterated identification of near twins.



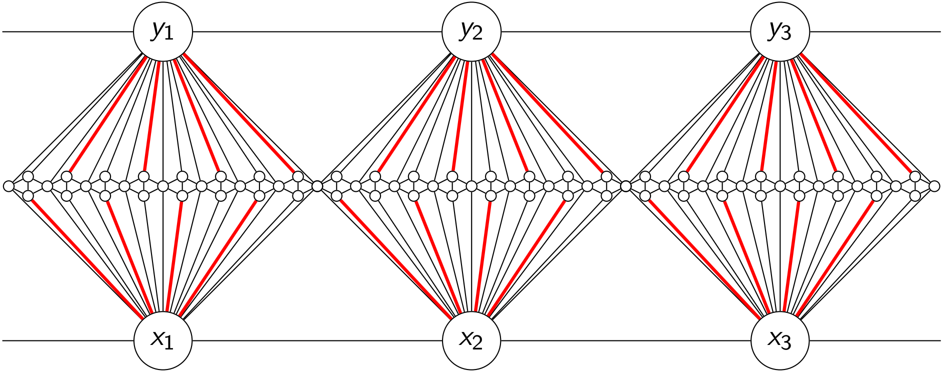
Planar graphs?

Planar graphs?



For every d, a planar trigraph without planar d-contraction

Planar graphs?



For every d, a planar trigraph without planar d-contraction

More powerfool tool needed

Twin-width in the language of matrices

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Encode a bipartite graph (or, if symmetric, any graph)

Twin-width in the language of matrices

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Contraction of two columns (similar with two rows)

Twin-width in the language of matrices

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & r & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & r & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & r & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

How is the twin-width (re)defined?

Twin-width in the language of matrices

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & r & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & r & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & r & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

How to tune it for non-bipartite graph?

Partition viewpoint

Matrix partition: partitions of the row set and of the column set
Matrix division: same but all the parts are *consecutive*

1	1	1	1	1	1	1	0
0	1	1	0	0	1	0	1
0	0	0	0	0	0	0	1
0	1	0	0	1	0	1	0
1	0	0	1	1	0	1	0
0	1	1	1	1	1	0	0
1	0	1	1	1	0	0	1

Partition viewpoint

Matrix partition: partitions of the row set and of the column set
Matrix division: same but all the parts are *consecutive*

$$\left[\begin{array}{c|c|cc|c|c|c|c}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ \hline
0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ \hline
1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ \hline
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ \hline
1 & 0 & 1 & 1 & 1 & 0 & 0 & 1
\end{array}\right]$$

Maximum number of non-constant zones per column or row part
$=$ **error value**

Partition viewpoint

Matrix partition: partitions of the row set and of the column set
Matrix division: same but all the parts are *consecutive*

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ \hline 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ \hline 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ \hline 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Maximum number of non-constant zones per column or row part
. . . until there are a single row part and column part

Partition viewpoint

Matrix partition: partitions of the row set and of the column set
Matrix division: same but all the parts are *consecutive*

1	1	1	1	1	1	1	0
0	1	1	0	0	1	0	1
0	0	0	0	0	0	0	1
0	1	0	0	1	0	1	0
1	0	0	1	1	0	1	0
0	1	1	1	1	1	0	0
1	0	1	1	1	0	0	1

Twin-width as maximum error value of a contraction sequence

Grid minor

t-grid minor: $t \times t$-division where every cell is non-empty
Non-empty cell: contains at least one 1 entry

$$
\left[\begin{array}{cc|cc|cc|cc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
\hline
0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
\hline
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 & 0 & 1
\end{array}\right]
$$

4-grid minor

Grid minor

t-grid minor: $t \times t$-division where every cell is non-empty
Non-empty cell: contains at least one 1 entry

$$\left[\begin{array}{cc|cc|cc|cc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
\hline
0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
\hline
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 & 0 & 1
\end{array}\right]$$

4-grid minor

A matrix is said t-**grid free** if it does not have a t-grid minor

Mixed minor

Mixed cell: not horizontal nor vertical

$$
\left[\begin{array}{cc|ccc|ccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
\hline
1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 & 0 & 1
\end{array}\right]
$$

3-mixed minor

Mixed minor

Mixed cell: not horizontal nor vertical

$$\left[\begin{array}{cc|ccc|ccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
\hline
1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 & 0 & 1
\end{array}\right]$$

3-mixed minor

Every mixed cell is witnessed by a 2×2 square = **corner**

Mixed minor

Mixed cell: not horizontal nor vertical

$$\left[\begin{array}{cc|ccc|ccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
\hline
1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 & 0 & 1
\end{array}\right]$$

3-mixed minor

A matrix is said t-**mixed free** if it does not have a t-mixed minor

Mixed value

$$
\begin{array}{c}
R_4 \\ \\ R_3 \\ \\ R_2 \\ \\ R_1
\end{array}
\left[\begin{array}{cc|ccc|c|cc}
1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ \hline
1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ \hline
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ \hline
0 & 1 & \color{red}{1} & \color{red}{1} & \color{red}{0} & 1 & 0 & 0 \\
1 & 0 & \color{red}{1} & \color{red}{0} & \color{red}{1} & 0 & 0 & 1
\end{array}\right]
$$

$$C_2$$

$\approx$ (maximum) number of cells with a corner per row/column part

Mixed value

R_4	1	1	1	0	0	1	1	0
R_3	1	0	1	0	0	1	0	1
	1	0	1	0	0	0	0	1
R_2	0	1	0	0	1	0	1	0
	1	1	0	0	1	0	1	0
R_1	0	1	1	1	0	1	0	0
	1	0	1	0	1	0	0	1

C_2

But we add the number of *boundaries* containing a corner

Mixed value

			C_2	C_2	C_2			
R_4	1	1	1	0	0	1	1	0
R_3	1	0	1	0	0	1	0	1
R_3	1	0	1	0	0	0	0	1
R_2	0	1	0	0	1	0	1	0
R_2	1	1	0	0	1	0	1	0
R_1	0	1	1	1	0	1	0	0
R_1	1	0	1	0	1	0	0	1

$R_3 \cup R_2$

$\therefore$ merging row parts do not increase mixed value of column part

Twin-width and mixed freeness

Theorem (B., Kim, Thomassé, Watrigant '20)

*If G admits **a** t-mixed free adjacency matrix, then* $tww(G) = 2^{2^{O(t)}}$.

Twin-width and mixed freeness

Theorem (B., Kim, Thomassé, Watrigant '20)

If $\exists\sigma$ *s.t.* $Adj_\sigma(G)$ *is* t*-mixed free, then* $tww(G) = 2^{2^{O(t)}}$.

Twin-width and mixed freeness

Theorem (B., Kim, Thomassé, Watrigant '20)

If $\exists\sigma$ s.t. $Adj_\sigma(G)$ is t-mixed free, then $tww(G) = 2^{2^{O(t)}}$.

Step 1: find a division sequence $(\mathcal{D}_i)_i$ with mixed value $f(t)$

1	1	1	1	1	1	1	0
0	1	1	0	0	1	0	1
0	0	0	0	0	0	0	1
0	1	0	0	1	0	1	0
1	0	0	1	1	0	1	0
0	1	1	1	1	1	0	0
1	0	1	1	1	0	0	1

Merge consecutive parts greedily

Twin-width and mixed freeness

Theorem (B., Kim, Thomassé, Watrigant '20)

If $\exists \sigma$ *s.t.* $Adj_\sigma(G)$ *is* t*-mixed free, then* $tww(G) = 2^{2^{O(t)}}$.

Step 1: find a division sequence $(\mathcal{D}_i)_i$ **with mixed value** $f(t)$

1	1	1	1	1	1	1	0
0	1	1	0	0	1	0	1
0	0	0	0	0	0	0	1
0	1	0	0	1	0	1	0
1	0	0	1	1	0	1	0
0	1	1	1	1	1	0	0
1	0	1	1	1	0	0	1

Merge consecutive parts greedily

Twin-width and mixed freeness

Theorem (B., Kim, Thomassé, Watrigant '20)

If $\exists\sigma$ *s.t.* $Adj_\sigma(G)$ *is* t*-mixed free, then* $tww(G) = 2^{2^{O(t)}}$.

Step 1: find a division sequence $(\mathcal{D}_i)_i$ **with mixed value** $f(t)$

1	1	1	1	1	1	1	0
0	1	1	0	0	1	0	1
0	0	0	0	0	0	0	1
0	1	0	0	1	0	1	0
1	0	0	1	1	0	1	0
0	1	1	1	1	1	0	0
1	0	1	1	1	0	0	1

Merge consecutive parts greedily

Twin-width and mixed freeness

Theorem (B., Kim, Thomassé, Watrigant '20)

If $\exists\sigma$ *s.t.* $Adj_\sigma(G)$ *is t-mixed free, then* $tww(G) = 2^{2^{O(t)}}$.

Step 1: find a division sequence $(\mathcal{D}_i)_i$ **with mixed value** $f(t)$

1	1	1	1	1	1	1	0
0	1	1	0	0	1	0	1
0	0	0	0	0	0	0	1
0	1	0	0	1	0	1	0
1	0	0	1	1	0	1	0
0	1	1	1	1	1	0	0
1	0	1	1	1	0	0	1

Stuck, removing every other separation $\rightarrow \frac{f(t)}{2}$ mixed cells per part

Marcus-Tardos theorem

Theorem (Marcus and Tardos '04, Stanley-Wilf conjecture)

For every k, there is a c_k such that every $n \times m$ $0,1$-matrix with at least $c_k \max(n, m)$ 1 entries admits a k-grid minor.

Marcus-Tardos theorem

Theorem (Marcus and Tardos '04, Stanley-Wilf conjecture)

For every k, there is a c_k such that every $n \times m$ $0,1$-matrix with at least $c_k \max(n, m)$ 1 entries admits a k-grid minor.

Auxiliary 0,1-matrix with one entry per cell: a 1 iff the cell is mixed

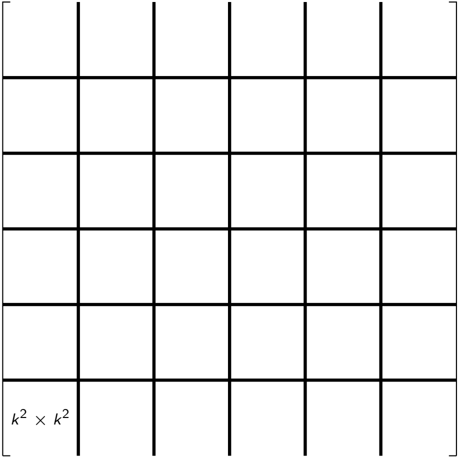
Marcus-Tardos one-page inductive proof

$$M = \begin{bmatrix} \\ \\ \\ \\ \end{bmatrix}$$

Let M be an $n \times n$ $0,1$-matrix without k-grid minor

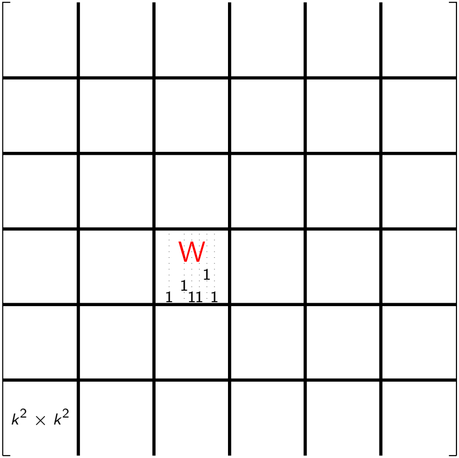
Marcus-Tardos one-page inductive proof

$M =$



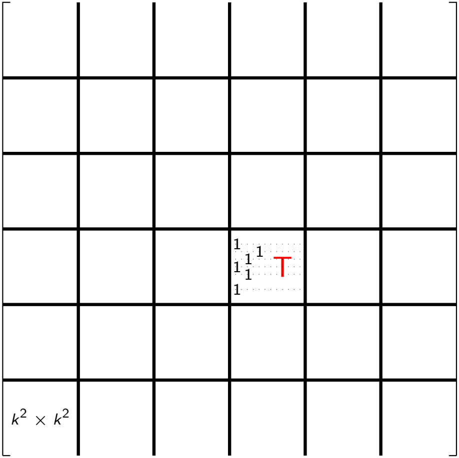
Draw a regular $\frac{n}{k^2} \times \frac{n}{k^2}$ division on top of M

Marcus-Tardos one-page inductive proof

$M =$


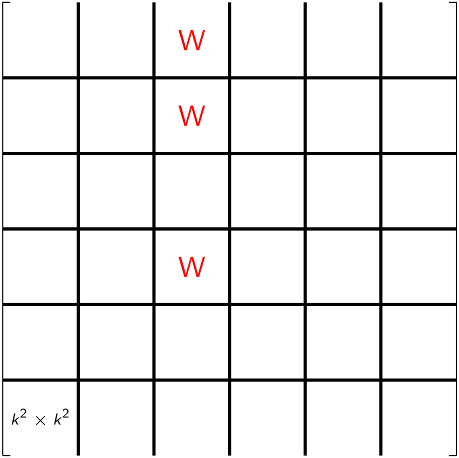
A cell is *wide* if it has at least k columns with a 1

Marcus-Tardos one-page inductive proof

$M =$ 

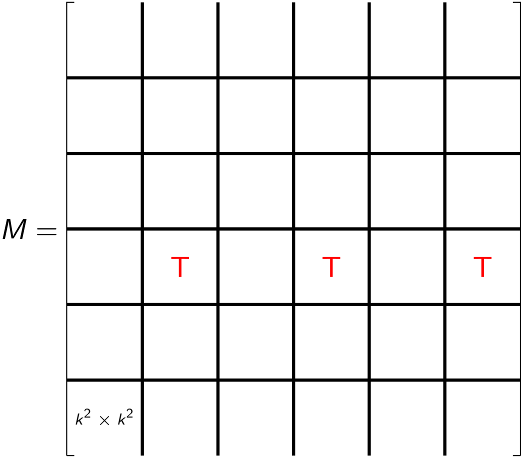
A cell is *tall* if it has at least k rows with a 1

Marcus-Tardos one-page inductive proof

$M =$ 

There are less than $k\binom{k^2}{k}$ wide cells per column part. Why?

Marcus-Tardos one-page inductive proof



There are less than $k\binom{k^2}{k}$ tall cells per row part

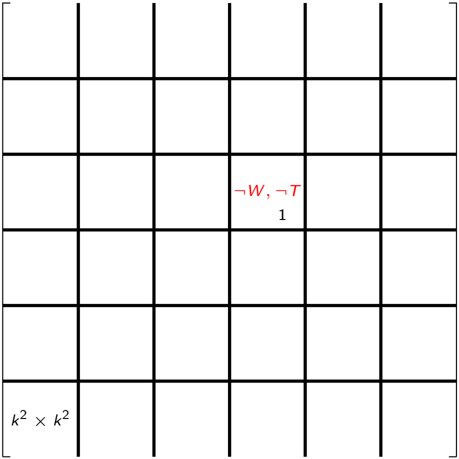
Marcus-Tardos one-page inductive proof

$$M = \begin{bmatrix} & & W & & & \\ & W & W & & & T \\ & & & & & \\ & T & W & T & & T \\ & & T & & & \\ k^2 \times k^2 & & & & & W \end{bmatrix}$$

In W and T, at most $2 \cdot \frac{n}{k^2} \cdot k\binom{k^2}{k} \cdot k^4 = 2k^3\binom{k^2}{k}n$ entries 1

Marcus-Tardos one-page inductive proof

$M =$



There are at most $(k-1)^2 c_k \frac{n}{k^2}$ remaining 1. Why?

Marcus-Tardos one-page inductive proof

$$M = \left[\begin{array}{c|c|c|c|c|c}
 & & W & & & \\ \hline
 & W & W & & & T \\ \hline
 & & & \neg W, \neg T \;\; 1 & & \\ \hline
 & T & W & T & & T \\ \hline
 & & T & & & \\ \hline
k^2 \times k^2 & & & & & W
\end{array}\right]$$

Choose $c_k = 2k^4\binom{k^2}{k}$ so that $(k-1)^2 c_k \frac{n}{k^2} + 2k^3\binom{k^2}{k} n \leqslant c_k n$

Twin-width and mixed freeness

Theorem (B., Kim, Thomassé, Watrigant '20)

If $\exists \sigma$ s.t. $Adj_\sigma(G)$ is t-mixed free, then $tww(G) = 2^{2^{O(t)}}$.

Step 1: find a division sequence $(\mathcal{D}_i)_i$ with mixed value $f(t)$

1	1	1	1	1	1	1	0
0	1	1	0	0	1	0	1
0	0	0	0	0	0	0	1
0	1	0	0	1	0	1	0
1	0	0	1	1	0	1	0
0	1	1	1	1	1	0	0
1	0	1	1	1	0	0	1

Stuck, removing every other separation $\rightarrow \frac{f(t)}{2}$ mixed cells per part

Twin-width and mixed freeness

Theorem (B., Kim, Thomassé, Watrigant '20)

If $\exists\sigma$ *s.t.* $Adj_\sigma(G)$ *is* t*-mixed free, then* $tww(G) = 2^{2^{O(t)}}$.

Step 1: find a division sequence $(\mathcal{D}_i)_i$ **with mixed value** $f(t)$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Stuck, removing every other separation $\rightarrow \frac{f(t)}{2}$ mixed cells per part

Impossible!

Twin-width and mixed freeness

Theorem (B., Kim, Thomassé, Watrigant '20)

If $\exists \sigma$ *s.t.* $Adj_\sigma(G)$ *is* t*-mixed free, then* $tww(G) = 2^{2^{O(t)}}$.

Step 1: find a division sequence $(\mathcal{D}_i)_i$ **with mixed value** $f(t)$
Step 2: find a contraction sequence with error value $g(t)$

1	1	1	1	1	1	1	0
0	1	1	0	0	1	0	1
0	0	0	0	0	0	0	1
0	1	0	0	1	0	1	0
1	0	0	1	1	0	1	0
0	1	1	1	1	1	0	0
1	0	1	1	1	0	0	1

Refinement of $\mathcal{D}_i$ where each part coincides on the non-mixed cells

Twin-width and mixed freeness

Theorem (B., Kim, Thomassé, Watrigant '20)

If $\exists \sigma$ *s.t.* $Adj_\sigma(G)$ *is* t*-mixed free, then* $tww(G) = 2^{2^{O(t)}}$.

Twin-width and mixed freeness

Theorem (B., Kim, Thomassé, Watrigant '20)

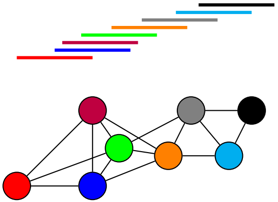
If $\exists\sigma$ *s.t.* $Adj_\sigma(G)$ *is t-mixed free, then* $tww(G) = 2^{2^{O(t)}}$.

Now to bound the twin-width of a class $\mathcal{C}$:

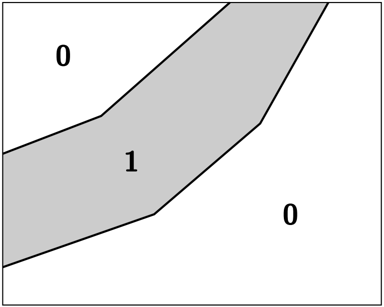
1) Find a *good* vertex-ordering procedure
2) Argue that, in this order, a t-mixed minor would conflict with $\mathcal{C}$

Unit interval graphs

Intersection graph of unit segments on the real line

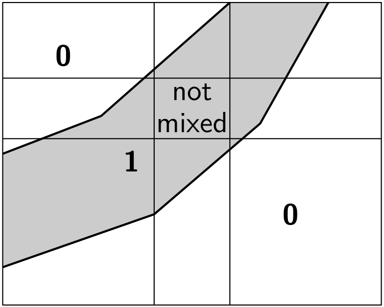


Unit interval graphs



order by left endpoints

Unit interval graphs



No 3-by-3 grid has all 9 cells crossed by two non-decreasing curves

Graph minors

Formed by **vertex deletion**, **edge deletion**, and **edge contraction**

A graph G is *H-minor free* if H is not a minor of G

A graph class is *H-minor free* if all its graphs are

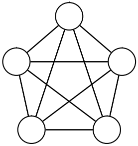
Graph minors

Formed by **vertex deletion**, **edge deletion**, and **edge contraction**

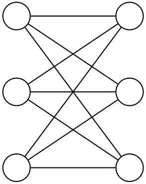
A graph G is *H-minor free* if H is not a minor of G

A graph class is *H-minor free* if all its graphs are

Planar graphs are exactly the graphs without K_5 or $K_{3,3}$ as a minor



K_5

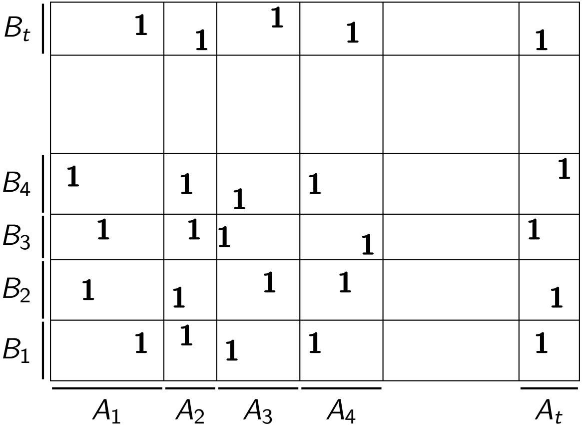


$K_{3,3}$

Bounded twin-width – K_t-minor free graphs

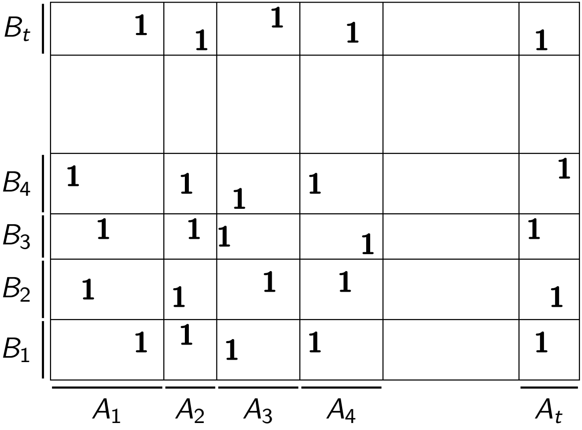
Given a hamiltonian path, we would just use this order

Bounded twin-width – K_t-minor free graphs



Contracting the $2t$ subpaths yields a $K_{t,t}$-minor, hence a K_t-minor

Bounded twin-width – K_t-minor free graphs

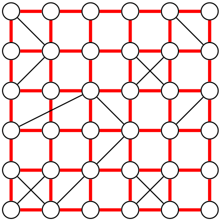


Instead we use a specially crafted lex-DFS discovery order

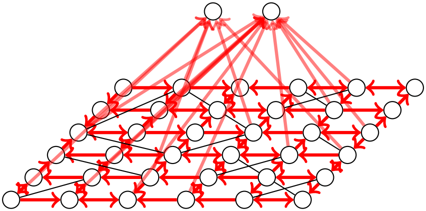
A surprising and convenient equivalent

Theorem (B., Kim, Reinald, Thomassé '21+)

Twin-width and oriented twin-width are functionally equivalent.



red degree

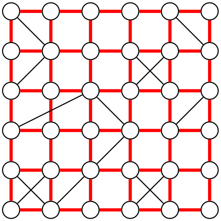


red out-degree
(red arcs oriented from the contraction)

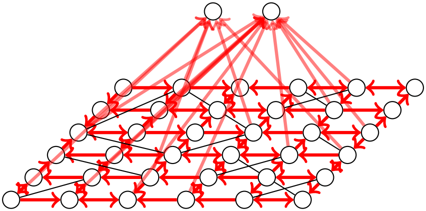
A surprising and convenient equivalent

Theorem (B., Kim, Reinald, Thomassé '21+)

Twin-width and oriented twin-width are functionally equivalent.



red degree



red out-degree
(red arcs oriented from the contraction)

Theorem (Kotzig's theorem '55)

Planar graphs have oriented twin-width at most 9.

Theorem (B., Geniet, Kim, Thomassé, Watrigant '20 & '21)

The following classes have bounded twin-width, and $O(1)$-sequences can be computed in polynomial time.

- *Bounded rank-width, and even, boolean-width graphs,*
- *every hereditary proper subclass of permutation graphs,*
- *posets of bounded antichain size (seen as digraphs),*
- *unit interval graphs,*
- *K_t-minor free graphs,*
- *map graphs,*
- *subgraphs of d-dimensional grids,*
- *K_t-free unit d-dimensional ball graphs,*
- *$\Omega(\log n)$-subdivisions of all the n-vertex graphs,*
- *cubic expanders defined by iterative random 2-lifts from K_4,*
- *strong products of two bounded twin-width classes, one with bounded degree, etc.*

Theorem (B., Geniet, Kim, Thomassé, Watrigant '20 & '21)

The following classes have bounded twin-width, and $O(1)$-sequences can be computed in polynomial time.

- *Bounded rank-width, and even, boolean-width graphs,*
- *every hereditary proper subclass of permutation graphs,*
- *posets of bounded antichain size (seen as digraphs),*
- *unit interval graphs,*
- K_t*-minor free graphs,*
- *map graphs,*
- *subgraphs of* d*-dimensional grids,*
- K_t*-free unit* d*-dimensional ball graphs,*
- $\Omega(\log n)$*-subdivisions of all the* n*-vertex graphs,*
- *cubic expanders defined by iterative random 2-lifts from* K_4*,*
- *strong products of two bounded twin-width classes, one with bounded degree, etc.*

Can we solve problems faster, given an $O(1)$-sequence?

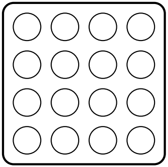
One cograph definition

Cographs form the unique *maximal hereditary* class in which every[1] graph has two *twins*

[1] provided it has at least two vertices

One cograph definition

Cographs form the unique *maximal hereditary* class in which every[1] graph has two *twins*

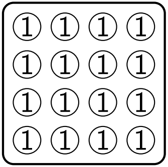


Is there another algorithmic scheme based on this definition?

[1] provided it has at least two vertices

One cograph definition

Cographs form the unique *maximal hereditary* class in which every[1] graph has two *twins*

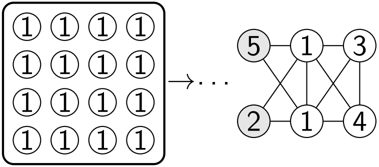


Let's try with $\alpha(G)$, and store in a vertex its inner max solution

[1] provided it has at least two vertices

One cograph definition

Cographs form the unique *maximal hereditary* class in which every[1] graph has two *twins*

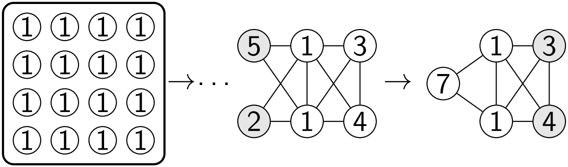


We can find a pair of false/true twins

[1] provided it has at least two vertices

One cograph definition

Cographs form the unique *maximal hereditary* class in which every[1] graph has two *twins*

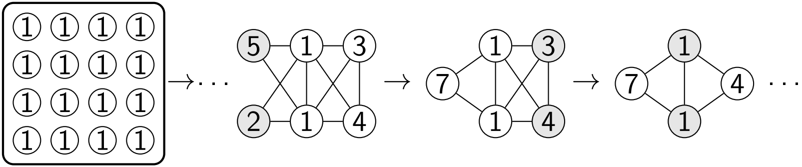


Sum them if they are false twins

[1] provided it has at least two vertices

One cograph definition

Cographs form the unique *maximal hereditary* class in which every[1] graph has two *twins*



Max them if they are true twins

[1] provided it has at least two vertices

Example of k-Independent Set

d-sequence: $G = G_n, G_{n-1}, \ldots, G_2, G_1 = K_1$

Algorithm: **Compute by dynamic programming a best partial solution in each red connected subgraph of size at most k.**

Example of k-Independent Set

d-sequence: $G = G_n, G_{n-1}, \ldots, G_2, G_1 = K_1$

Algorithm: **Compute by dynamic programming a best partial solution in each red connected subgraph of size at most k.**

$d^{2k}n^2$ red connected subgraphs, actually only $d^{2k}n = 2^{O_d(k)}n$

Example of k-Independent Set

d-sequence: $G = G_n, G_{n-1}, \ldots, G_2, G_1 = K_1$

Algorithm: **Compute by dynamic programming a best partial solution in each red connected subgraph of size at most k.**

$d^{2k}n^2$ red connected subgraphs, actually only $d^{2k}n = 2^{O_d(k)}n$

In G_n: red connected subgraphs are singletons, so are the solutions.
In G_1: If solution of size at least k, global solution.

Example of k-Independent Set

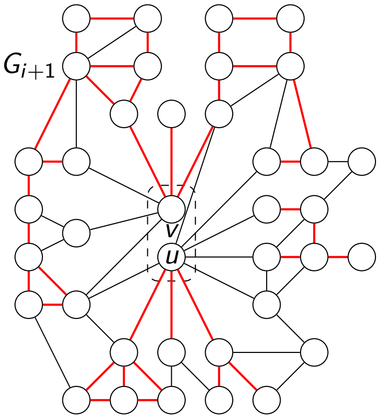
d-sequence: $G = G_n, G_{n-1}, \ldots, G_2, G_1 = K_1$

Algorithm: **Compute by dynamic programming a best partial solution in each red connected subgraph of size at most k.**

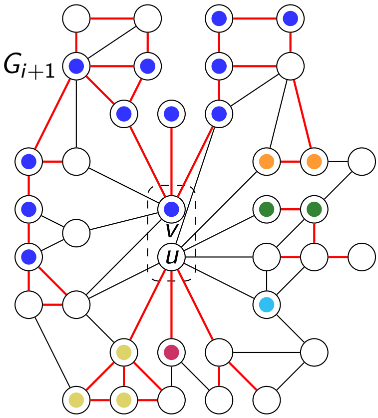
$d^{2k} n^2$ red connected subgraphs, actually only $d^{2k} n = 2^{O_d(k)} n$

In G_n: red connected subgraphs are singletons, so are the solutions.
In G_1: If solution of size at least k, global solution.

How to go from the partial solutions of G_{i+1} to those of G_i?



Best partial solution inhabiting •?



3 unions of $\leqslant d+2$ red connected subgraphs to consider in G_{i+1} with u, or v, or both

Other (almost) single-exponential parameterized algorithms

Theorem (B., Geniet, Kim, Thomassé, Watrigant '21)

Given a d-sequence $G = G_n, \ldots, G_1 = K_1$,

- k-Independent Set,
- k-Clique,
- (r, k)-Scattered Set,
- k-Dominating Set, *and*
- (r, k)-Dominating Set

can be solved in time $2^{O(k)}n$,
whereas Subgraph Isomorphism *and* Induced Subgraph Isomorphism *can be solved in time $2^{O(k \log k)}n$.*

Other (almost) single-exponential parameterized algorithms

Theorem (B., Geniet, Kim, Thomassé, Watrigant '21)

Given a d-sequence $G = G_n, \ldots, G_1 = K_1$,

- k-Independent Set,
- k-Clique,
- (r, k)-Scattered Set,
- k-Dominating Set, *and*
- (r, k)-Dominating Set

can be solved in time $2^{O(k)}n$,
whereas Subgraph Isomorphism *and* Induced Subgraph Isomorphism *can be solved in time $2^{O(k \log k)}n$.*

A more general FPT algorithm?

First-order model checking on graphs

GRAPH FO MODEL CHECKING	**Parameter:** $\|\varphi\|$
Input: A graph G and a first-order sentence $\varphi \in FO(\{E_2, =_2\})$	
Question: $G \models \varphi$?	

First-order model checking on graphs

GRAPH FO MODEL CHECKING	**Parameter:** $\lvert\varphi\rvert$
Input: A graph G and a first-order sentence $\varphi \in FO(\{E_2, =_2\})$	
Question: $G \models \varphi$?	

Example:

$$\varphi = \exists x_1 \exists x_2 \cdots \exists x_k \forall x \bigvee_{1 \leqslant i \leqslant k} x = x_i \vee \bigvee_{1 \leqslant i \leqslant k} E(x, x_i) \vee E(x_i, x)$$

$G \models \varphi$? $\Leftrightarrow$

First-order model checking on graphs

GRAPH FO MODEL CHECKING	**Parameter:** $\lvert\varphi\rvert$

Input: A graph G and a first-order sentence $\varphi \in FO(\{E_2, =_2\})$
Question: $G \models \varphi$?

Example:

$$\varphi = \exists x_1 \exists x_2 \cdots \exists x_k \forall x \bigvee_{1 \leqslant i \leqslant k} x = x_i \vee \bigvee_{1 \leqslant i \leqslant k} E(x, x_i) \vee E(x_i, x)$$

$G \models \varphi$? $\Leftrightarrow$ k-DOMINATING SET

First-order model checking on graphs

GRAPH FO MODEL CHECKING	**Parameter:** $\lvert\varphi\rvert$
Input: A graph G and a first-order sentence $\varphi \in FO(\{E_2, =_2\})$	
Question: $G \models \varphi$?	

Example:

$$\varphi = \exists x_1 \exists x_2 \cdots \exists x_k \bigwedge_{1 \leqslant i < j \leqslant k} \neg(x_i = x_j) \wedge \neg E(x_i, x_j) \wedge \neg E(x_j, x_i)$$

$G \models \varphi$? $\Leftrightarrow$

First-order model checking on graphs

GRAPH FO MODEL CHECKING	**Parameter:** $\lvert\varphi\rvert$

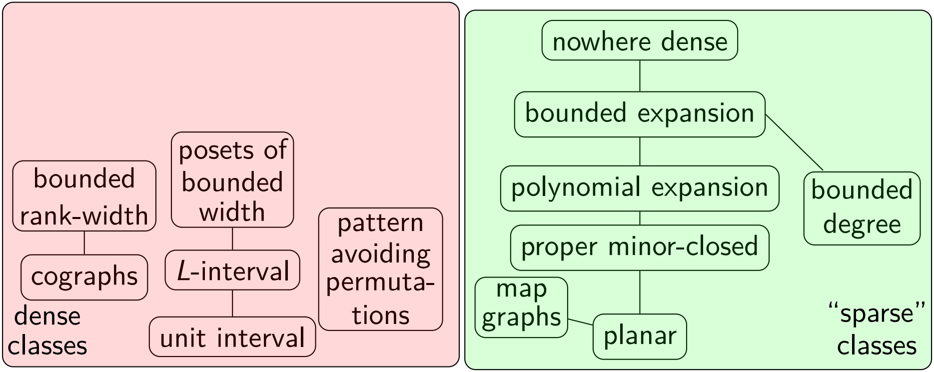
Input: A graph G and a first-order sentence $\varphi \in FO(\{E_2, =_2\})$
Question: $G \models \varphi$?

Example:

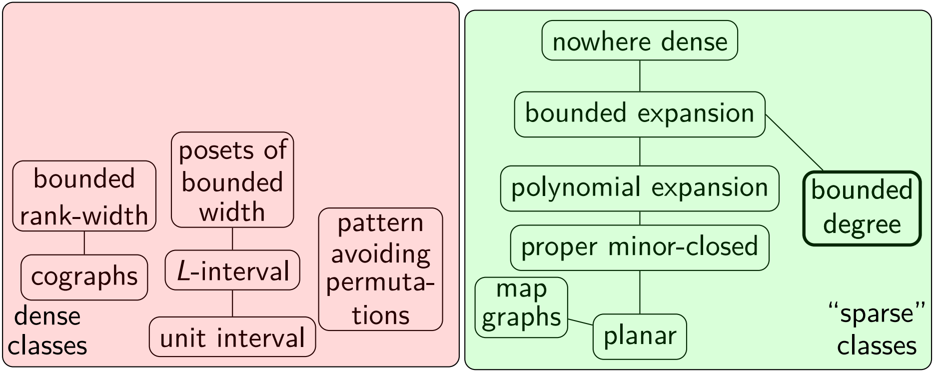
$$\varphi = \exists x_1 \exists x_2 \cdots \exists x_k \bigwedge_{1 \leqslant i < j \leqslant k} \neg(x_i = x_j) \wedge \neg E(x_i, x_j) \wedge \neg E(x_j, x_i)$$

$G \models \varphi$? $\Leftrightarrow$ k-INDEPENDENT SET

Classes with known tractable FO model checking

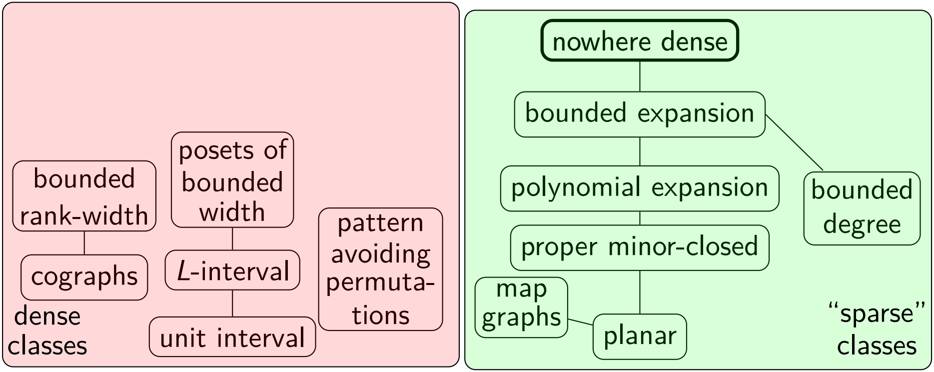


Classes with known tractable FO model checking



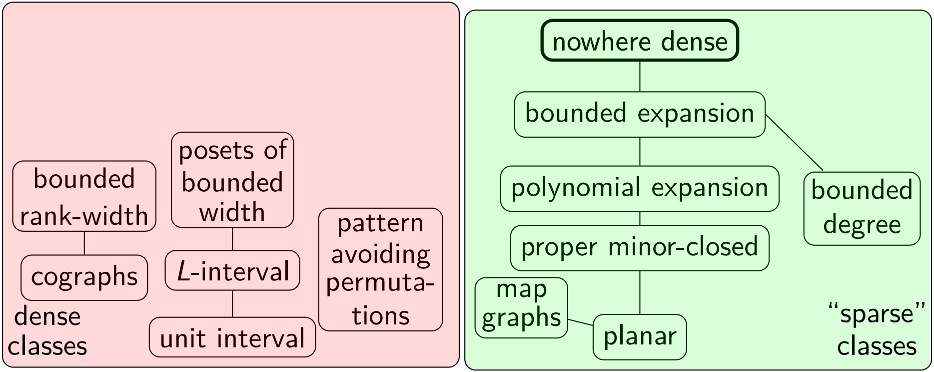
FO MODEL CHECKING solvable in $f(|\varphi|)n$ on bounded-degree graphs [Seese '96]

Classes with known tractable FO model checking



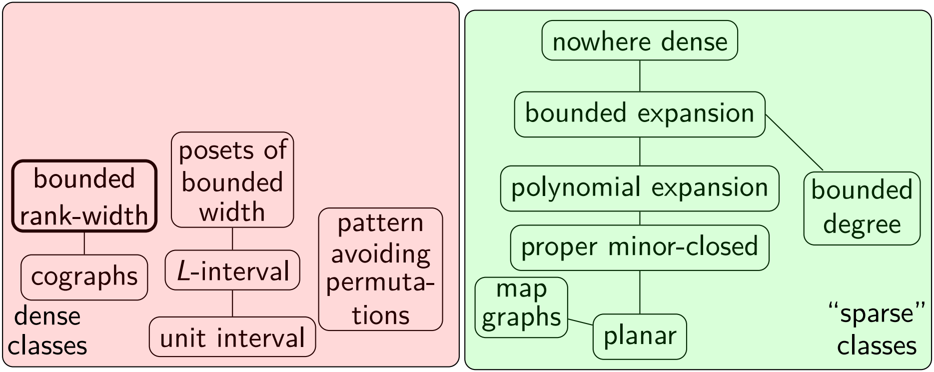
FO Model Checking solvable in $f(|\varphi|)n^{1+\varepsilon}$ on any nowhere dense class [Grohe, Kreutzer, Siebertz '14]

Classes with known tractable FO model checking



End of the story for the subgraph-closed classes
tractable FO MODEL CHECKING ⇔ nowhere dense

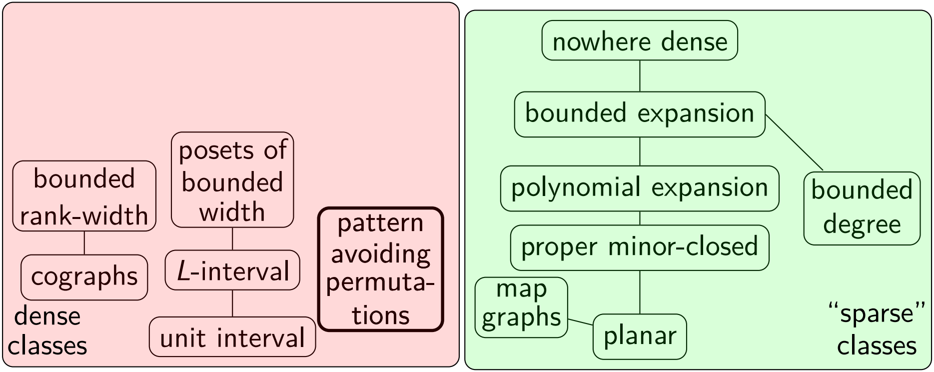
Classes with known tractable FO model checking



MSO_1 Model Checking solvable in $f(|\varphi|, w)n$ on graphs of rank-width w

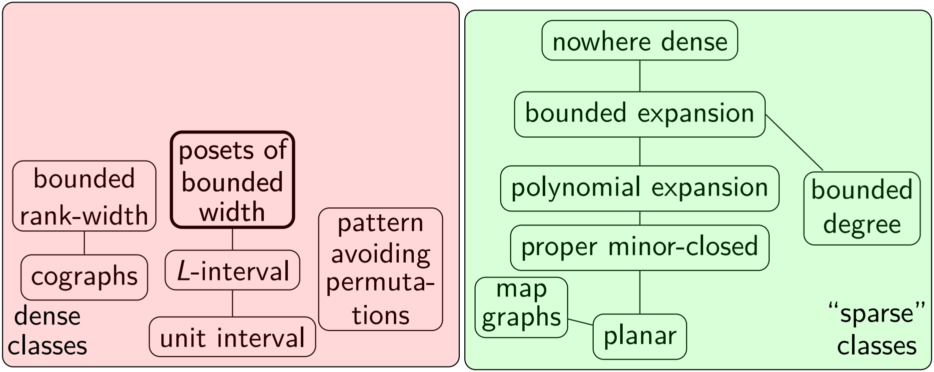
[Courcelle, Makowsky, Rotics '00]

Classes with known tractable FO model checking



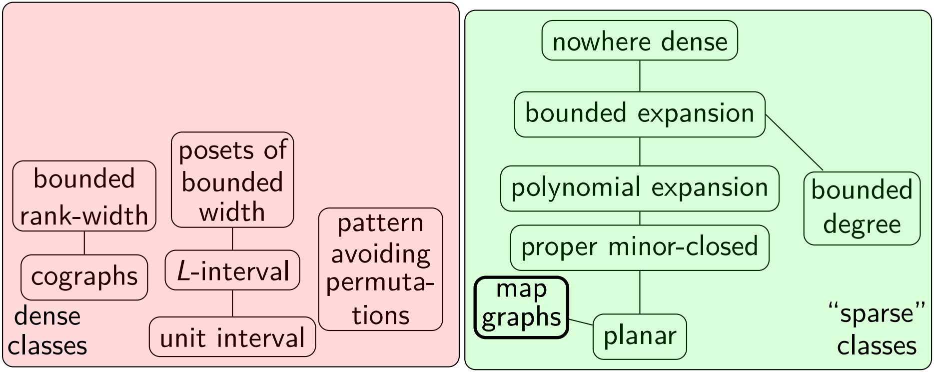
Is σ a subpermutation of τ? solvable in $f(|\sigma|)|\tau|$
[Guillemot, Marx '14]

Classes with known tractable FO model checking



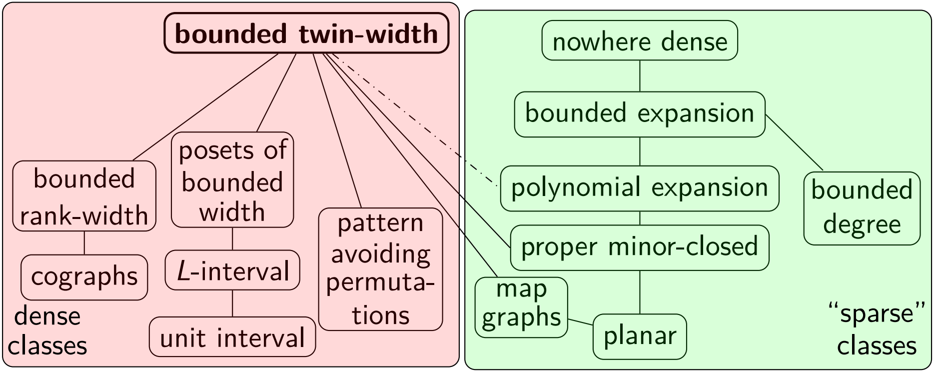
FO Model Checking solvable in $f(|\varphi|, w)n^2$ on posets of width w
[GHLOORS '15]

Classes with known tractable FO model checking



FO MODEL CHECKING solvable in $f(|\varphi|)n^{O(1)}$ on map graphs
[Eickmeyer, Kawarabayashi '17]

Classes with known tractable FO model checking

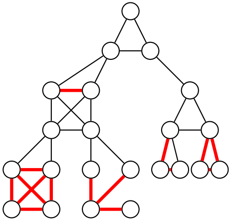


FO MODEL CHECKING solvable in $f(|\varphi|, d)n$ on graphs with a d-sequence [B., Kim, Thomassé, Watrigant '20]

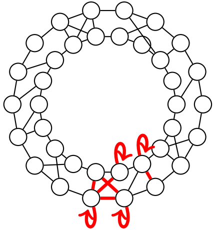
Classic width-measures via contraction sequences

Theorem (B., Kim, Reinald, Thomassé '21+)

Component twin-width is functionally equivalent to rank-width.
Total twin-width is functionally equivalent to linear rank-width.



Component twin-width:
max red component size



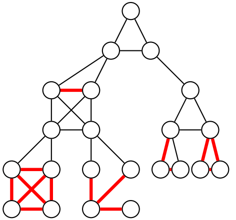
Total twin-width:
max number of red edges

The sparse regime captures treewidth and pathwidth

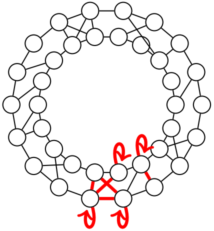
Classic width-measures via contraction sequences

Theorem (B., Kim, Reinald, Thomassé '21+)

Component twin-width is functionally equivalent to rank-width.
Total twin-width is functionally equivalent to linear rank-width.



Component twin-width:
max red component size



Total twin-width:
max number of red edges

Alternative proof of Courcelle, Makowsky, Rotics's theorem:
FO model checking approach using Feferman-Vaught instead of Gaifman's theorem

Small classes

Small: class with at most $n!c^n$ labeled graphs on $[n]$.

Theorem (B., Geniet, Kim, Thomassé, Watrigant '21)

Bounded twin-width classes are small.

Unifies and extends the same result for:
σ-free permutations [Marcus, Tardos '04]
K_t-minor free graphs [Norine, Seymour, Thomas, Wollan '06]

Small classes

Small: class with at most $n!c^n$ labeled graphs on $[n]$.

Theorem (B., Geniet, Kim, Thomassé, Watrigant '21)

Bounded twin-width classes are small.

Subcubic graphs, interval graphs, triangle-free unit segment graphs have **unbounded** twin-width

Small classes

Small: class with at most $n!c^n$ labeled graphs on $[n]$.

Theorem (B., Geniet, Kim, Thomassé, Watrigant '21)

Bounded twin-width classes are small.

The converse for hereditary classes does not hold

Theorem (B., Geniet, Tessera, Thomassé '21+)

There is a randomized construction of a finitely-generated group whose hereditary class of finite restrictions of the Cayley graph has unbounded twin-width (and yet is small).

Open questions

Algorithm to compute/approximate twin-width in general

Explicit examples of bounded-degree graphs of unbounded twin-width

Fully classify classes with tractable FO model checking

Some more classes could have bounded twin-width: polynomial expansion, $K_{t,t}$-free string graphs, etc.

Could smallness alone be algorithmically exploitable?

What about kernels? (Amadeus's talk)