# Twin-width

Édouard Bonnet, Colin Geniet, Eun Jung Kim,
Stéphan Thomassé, and Rémi Watrigant

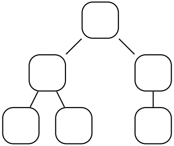
ENS Lyon, LIP

Séminaire Algorithmes et Complexité, IRIF, May 26th

# What is the most general tractable class?

# What are the most general tractable classes?

# What are the most general tractable classes?



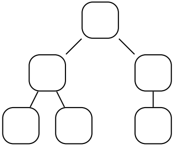
Bounded treewidth

Generalize

trees

tractable $MSO_2$

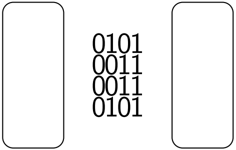
# What are the most general tractable classes?



Bounded treewidth

trees

tractable $MSO_2$



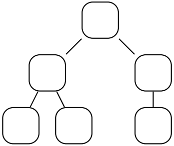

Bounded rank-width

cographs, treewidth

tractable $MSO_1$

Generalize

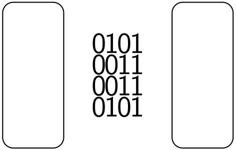
# What are the most general tractable classes?



Bounded treewidth

Generalize trees

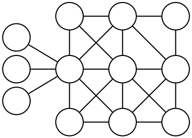
tractable $MSO_2$




Bounded rank-width

cographs, treewidth

tractable $MSO_1$

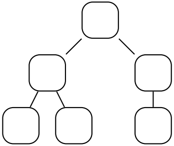


Nowhere dense

bounded degree, $H$-minor free

tractable FO

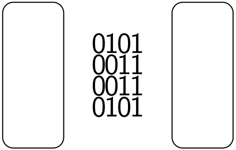
# What are the most general tractable classes?



Bounded treewidth

Generalize trees

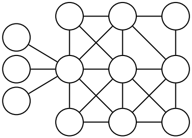
tractable $MSO_2$




Bounded rank-width

cographs, treewidth

tractable $MSO_1$



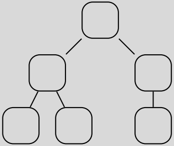
Nowhere dense

bounded degree, $H$-minor free

tractable FO

Also:
Bounded VC dimension
Perfect graphs
Bounded width posets
Pattern-avoiding permutations
⋮

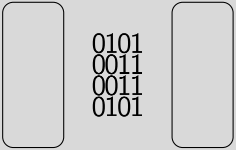
# What are the most general tractable classes?



Bounded treewidth

trees

tractable $MSO_2$

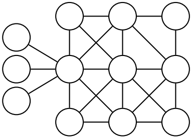



Bounded rank-width

cographs, treewidth

tractable $MSO_1$

Generalize



Nowhere dense

bounded degree, $H$-minor free

tractable FO

Also:

Bounded VC dimension

Perfect graphs

Bounded width posets

Pattern-avoiding permutations

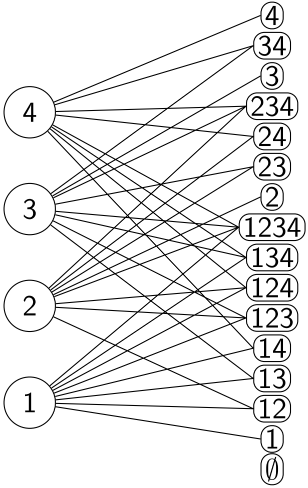
⋮

# Cograph generalization attempt

Iteratively identify **near** twins

# Cograph generalization attempt

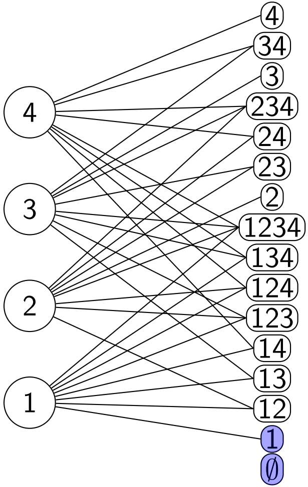
Iteratively identify **near** twins




This complicated graph passes the test

# Cograph generalization attempt

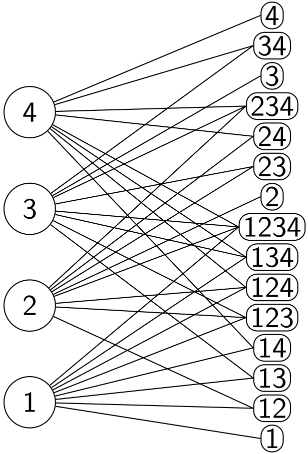
Iteratively identify **near** twins




This complicated graph passes the test

# Cograph generalization attempt

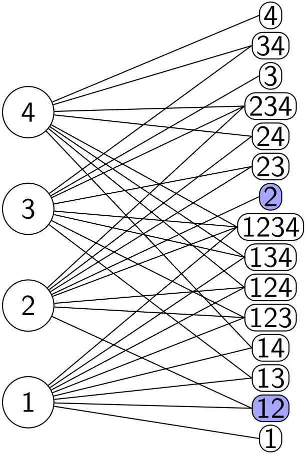
Iteratively identify **near** twins




This complicated graph passes the test

# Cograph generalization attempt

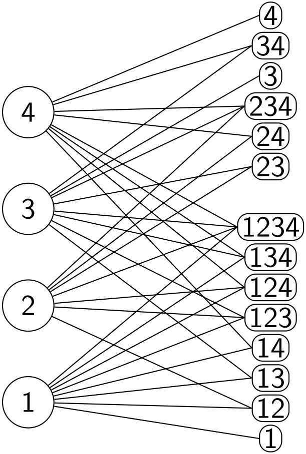
Iteratively identify **near** twins




This complicated graph passes the test

# Cograph generalization attempt

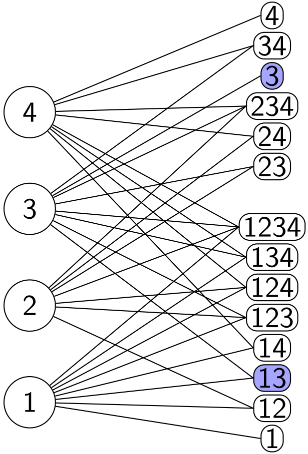
Iteratively identify **near** twins




This complicated graph passes the test

# Cograph generalization attempt

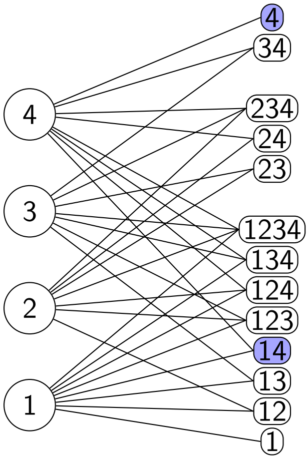
Iteratively identify **near** twins




This complicated graph passes the test

# Cograph generalization attempt

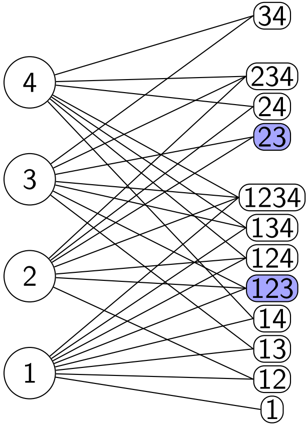
Iteratively identify **near** twins




This complicated graph passes the test

# Cograph generalization attempt

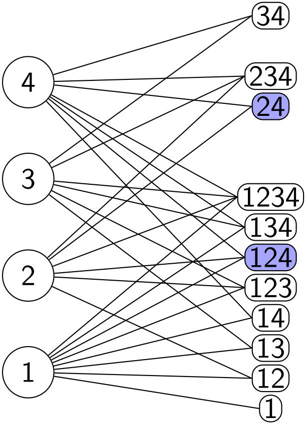
Iteratively identify **near** twins




This complicated graph passes the test

# Cograph generalization attempt

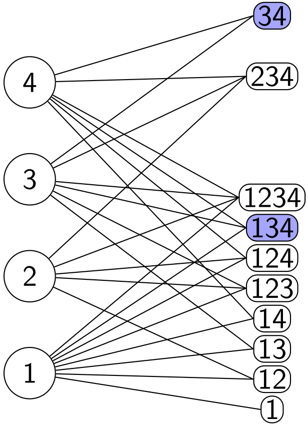
Iteratively identify **near** twins




This complicated graph passes the test

# Cograph generalization attempt

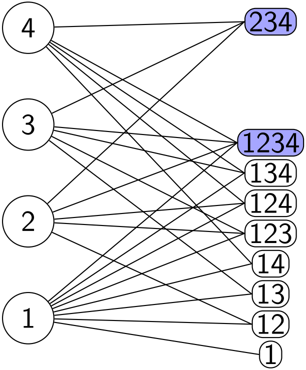
Iteratively identify **near** twins




This complicated graph passes the test

# Cograph generalization attempt

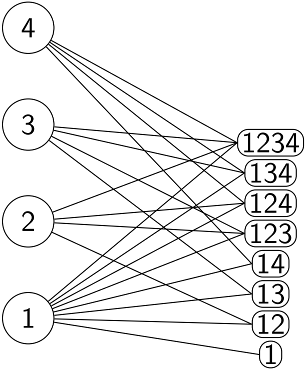
Iteratively identify **near** twins




This complicated graph passes the test

# Cograph generalization attempt

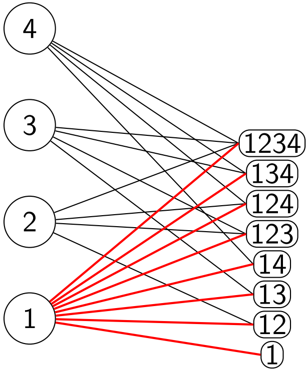
Iteratively identify **near** twins




This complicated graph passes the test

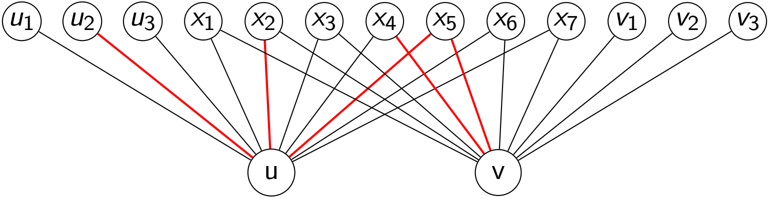
# Cograph generalization

Iteratively identify **near** twins and **keep the error degree small**



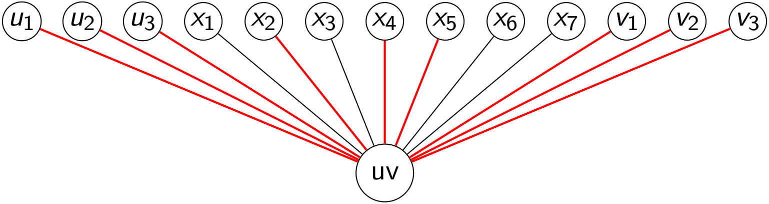

It would not with that further restriction

# Contraction and trigraph



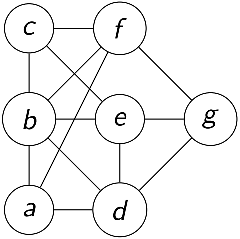

Trigraph: non-edges, edges, and red edges (error)

# Contraction and trigraph




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

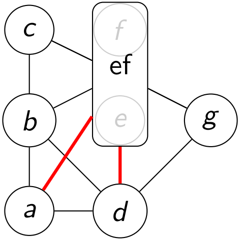
# Contraction sequence and twin-width




Maximum red degree $= 0$

**overall maximum red degree = 0**

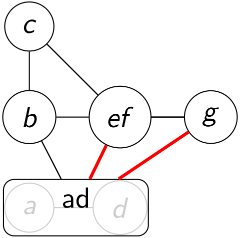
# Contraction sequence and twin-width




Maximum red degree = 2

**overall maximum red degree = 2**

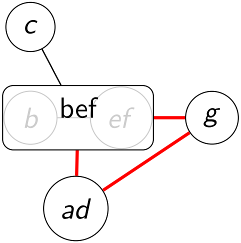
# Contraction sequence and twin-width




Maximum red degree = 2

**overall maximum red degree = 2**

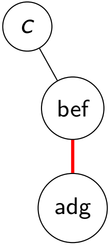
# Contraction sequence and twin-width




Maximum red degree = 2

**overall maximum red degree = 2**

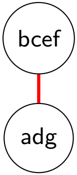
# Contraction sequence and twin-width




Maximum red degree $= 1$

**overall maximum red degree = 2**

# Contraction sequence and twin-width




Maximum red degree $= 1$

**overall maximum red degree = 2**

# Contraction sequence and twin-width


Maximum red degree = 0

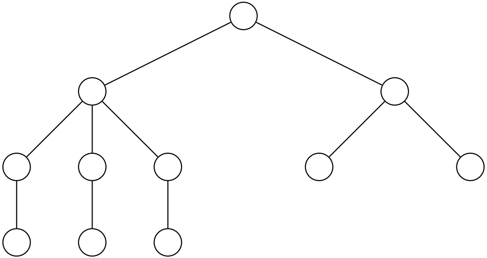
**overall maximum red degree = 2**

# Contraction sequence and twin-width

Sequence of 2-contractions or 2-sequence, twin-width at most 2

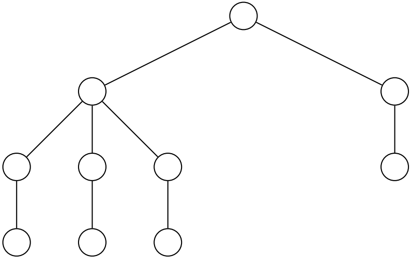

Maximum red degree = 0
**overall maximum red degree = 2**

# Graphs with bounded twin-width – trees



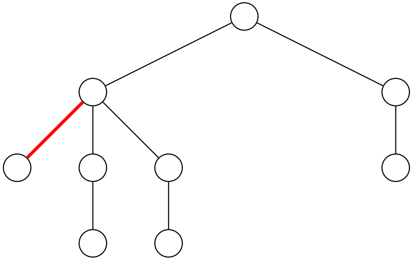
If possible, contract two twin leaves

# Graphs with bounded twin-width – trees



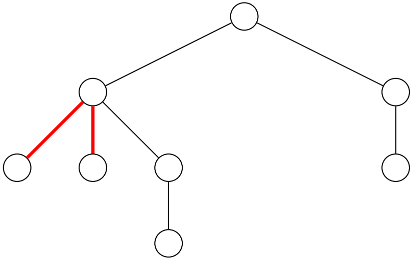
If not, contract a deepest leaf with its parent

# Graphs with bounded twin-width – trees



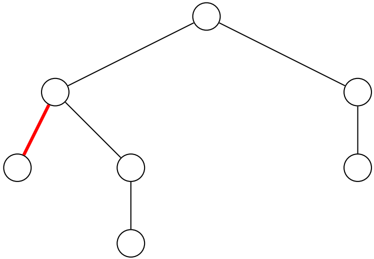
If not, contract a deepest leaf with its parent

# Graphs with bounded twin-width – trees



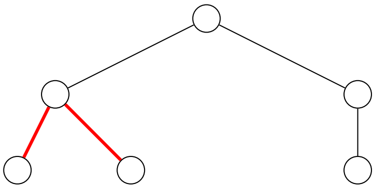
If possible, contract two twin leaves

# Graphs with bounded twin-width – trees



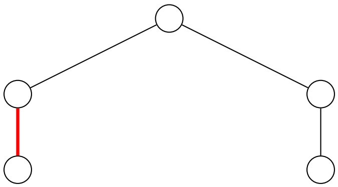
Cannot create a red degree-3 vertex

# Graphs with bounded twin-width – trees



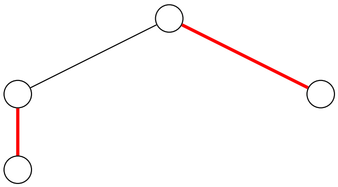
Cannot create a red degree-3 vertex

# Graphs with bounded twin-width – trees



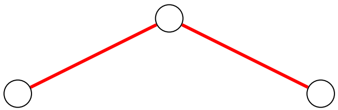
Cannot create a red degree-3 vertex

# Graphs with bounded twin-width – trees



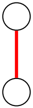
Cannot create a red degree-3 vertex

# Graphs with bounded twin-width – trees



Cannot create a red degree-3 vertex

# Graphs with bounded twin-width – trees

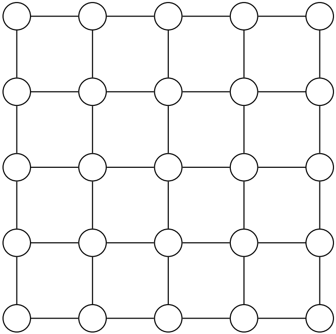


Cannot create a red degree-3 vertex

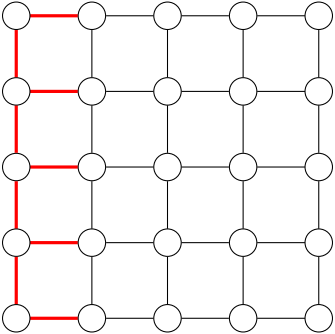
# Graphs with bounded twin-width – trees

Generalization to bounded treewidth and even bounded rank-width

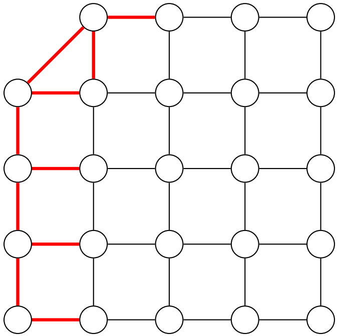
# Graphs with bounded twin-width – grids



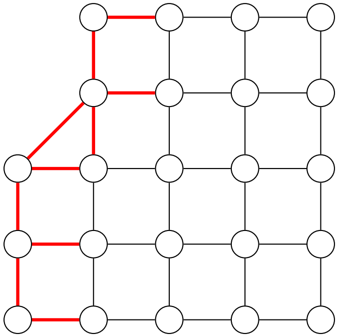
# Graphs with bounded twin-width – grids



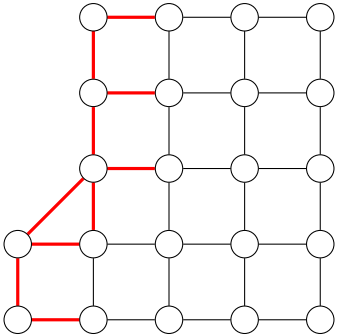
# Graphs with bounded twin-width – grids



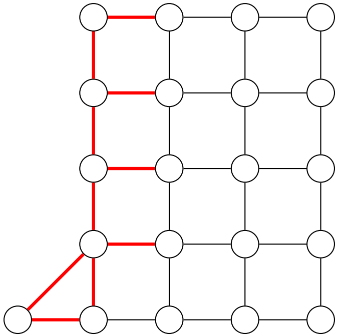
# Graphs with bounded twin-width – grids



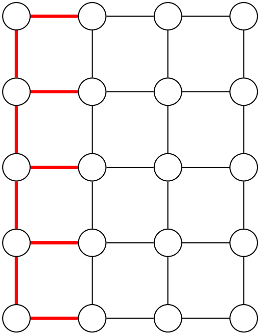
# Graphs with bounded twin-width – grids



# Graphs with bounded twin-width – grids



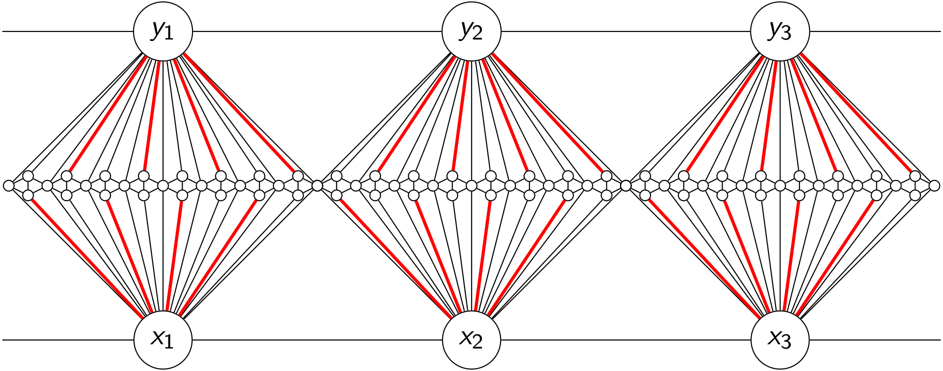
# Graphs with bounded twin-width – grids



4-sequence for planar grids, $3d$-sequence for $d$-dimensional grids

# Graphs with bounded twin-width – planar graphs?

# Graphs with bounded twin-width – planar graphs?




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

# More powerful tool needed

# First-order model checking on graphs

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

# First-order model checking on graphs

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

# FO interpretations and transductions

**FO interpretation:** redefine the edges by a first-order formula

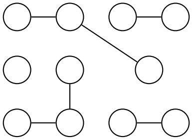
$\varphi(x, y) = \neg E(x, y)$ (complement)

$\varphi(x, y) = E(x, y) \vee \exists z E(x, z) \wedge E(z, y)$ (square)

# FO interpretations and transductions

**FO interpretation:** redefine the edges by a first-order formula
$\varphi(x, y) = \neg E(x, y)$ (complement)
$\varphi(x, y) = E(x, y) \vee \exists z E(x, z) \wedge E(z, y)$ (square)

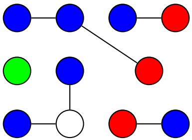
**FO transduction:** color by $O(1)$ unary relations, interpret, delete



# FO interpretations and transductions

**FO interpretation:** redefine the edges by a first-order formula
$\varphi(x, y) = \neg E(x, y)$ (complement)
$\varphi(x, y) = E(x, y) \vee \exists z E(x, z) \wedge E(z, y)$ (square)

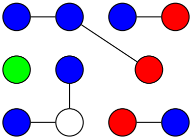
**FO transduction:** color by $O(1)$ unary relations, interpret, delete



# FO interpretations and transductions

**FO interpretation:** redefine the edges by a first-order formula
$\varphi(x,y) = \neg E(x,y)$ (complement)
$\varphi(x,y) = E(x,y) \vee \exists z E(x,z) \wedge E(z,y)$ (square)

**FO transduction:** color by $O(1)$ unary relations, interpret, delete

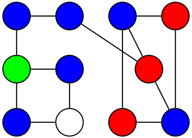


$$\phi(x,y) = E(x,y) \vee (G(x) \wedge B(y) \wedge \neg\exists z R(z) \wedge E(y,z))$$
$$\vee (R(x) \wedge B(y) \wedge \exists z R(z) \wedge E(y,z) \wedge \neg\exists z B(z) \wedge E(y,z))$$

# FO interpretations and transductions

**FO interpretation:** redefine the edges by a first-order formula
$\varphi(x, y) = \neg E(x, y)$ (complement)
$\varphi(x, y) = E(x, y) \vee \exists z E(x, z) \wedge E(z, y)$ (square)

**FO transduction:** color by $O(1)$ unary relations, interpret, delete

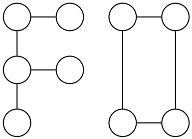


$$\phi(x, y) = E(x, y) \vee (G(x) \wedge B(y) \wedge \neg\exists z R(z) \wedge E(y, z))$$
$$\vee(R(x) \wedge B(y) \wedge \exists z R(z) \wedge E(y, z) \wedge \neg\exists z B(z) \wedge E(y, z))$$

# FO interpretations and transductions

**FO interpretation:** redefine the edges by a first-order formula
$\varphi(x, y) = \neg E(x, y)$ (complement)
$\varphi(x, y) = E(x, y) \vee \exists z E(x, z) \wedge E(z, y)$ (square)

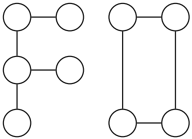
**FO transduction:** color by $O(1)$ unary relations, interpret, delete



# FO interpretations and transductions

**FO interpretation:** redefine the edges by a first-order formula
$\varphi(x, y) = \neg E(x, y)$ (complement)
$\varphi(x, y) = E(x, y) \vee \exists z E(x, z) \wedge E(z, y)$ (square)

**FO transduction:** color by $O(1)$ unary relations, interpret, delete

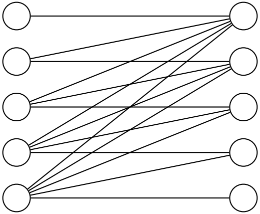


Theorem (B, Kim, Thomassé, Watrigant '20+)

*Bounded twin-width is preserved by transduction.*

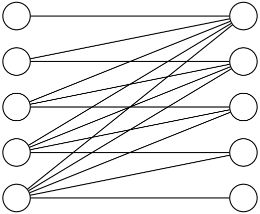
# Stable and NIP

**Stable class:** *not* all the ladders can be obtained by transduction
**NIP class:** *not* all the graphs can be obtained by transduction



# Stable and NIP

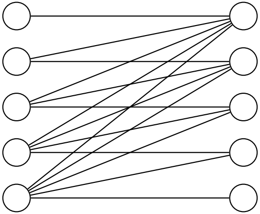
**Stable class:** *not* all the ladders can be obtained by transduction
**NIP class:** *not* all the graphs can be obtained by transduction



Bounded-degree graphs $\rightarrow$ stable
Unit interval graphs $\rightarrow$ NIP but not stable
Interval graphs $\rightarrow$ not NIP

# Stable and NIP

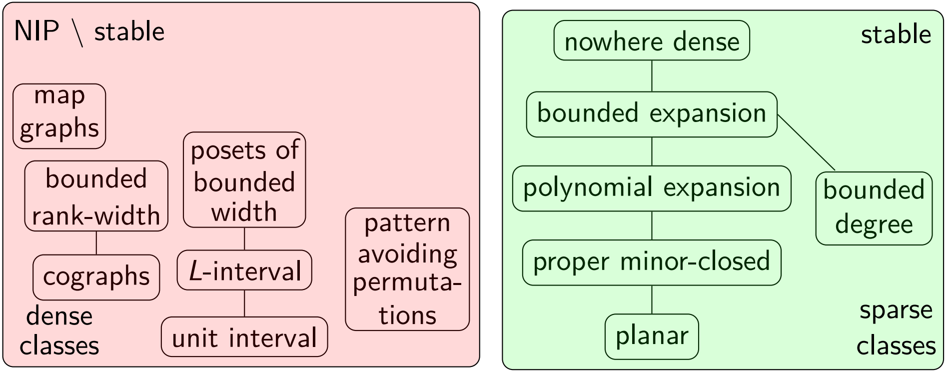
**Stable class:** *not* all the ladders can be obtained by transduction
**NIP class:** *not* all the graphs can be obtained by transduction



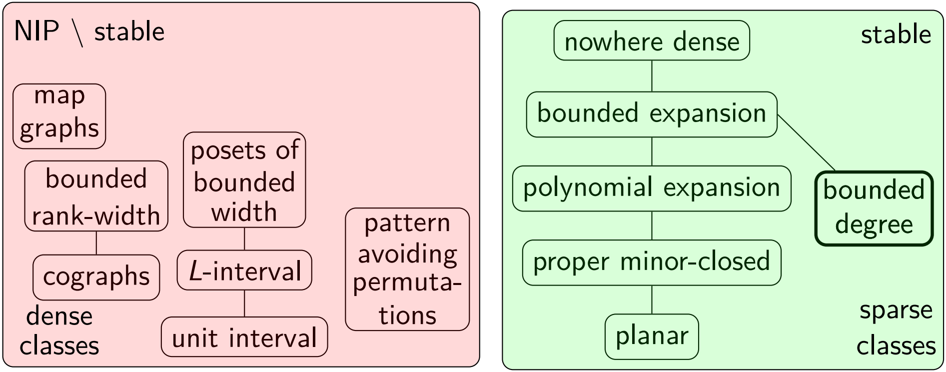
Bounded-degree graphs $\rightarrow$ stable
Unit interval graphs $\rightarrow$ NIP but not stable
Interval graphs $\rightarrow$ not NIP

**Bounded twin-width classes $\rightarrow$ NIP but not stable in general**

# Classes with known tractable FO model checking

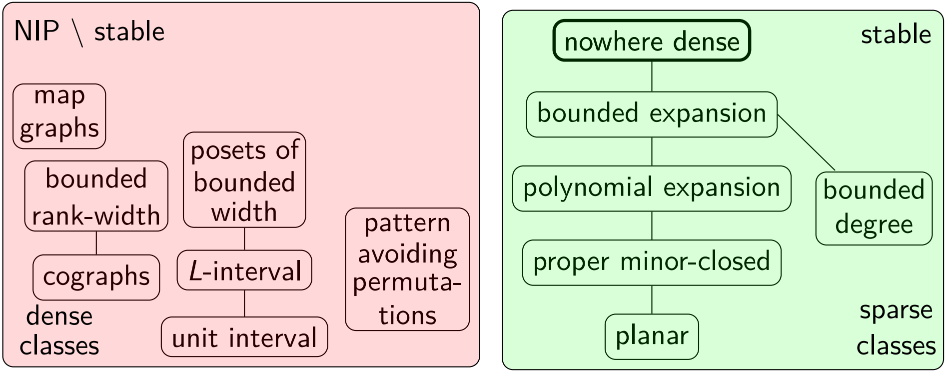


# Classes with known tractable FO model checking



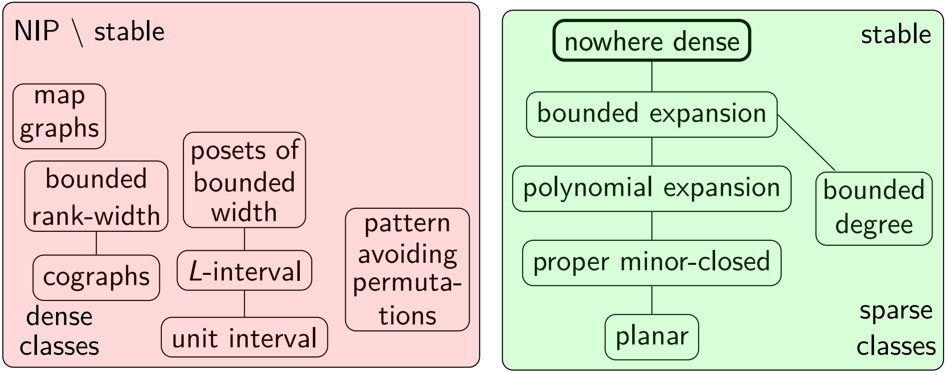

FO Model Checking solvable in $f(|\varphi|)n$ on bounded-degree graphs [Seese '96]

# 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]

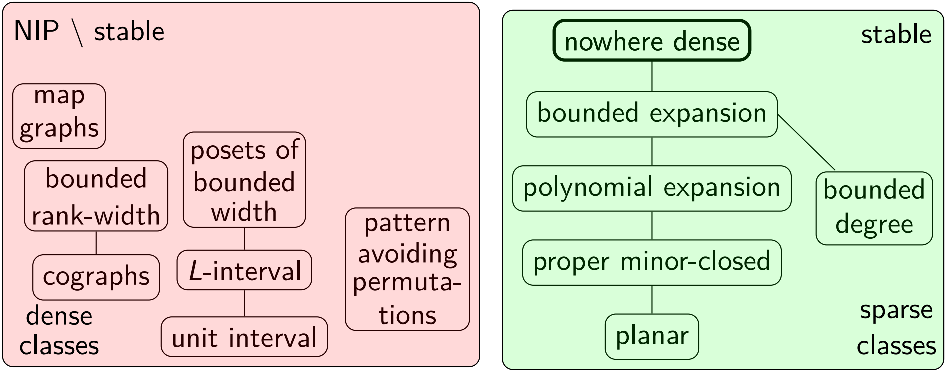
# Classes with known tractable FO model checking




End of the story for the classes closed by taking subgraphs

tractable FO MODEL CHECKING ⇔ nowhere dense ⇔ stable

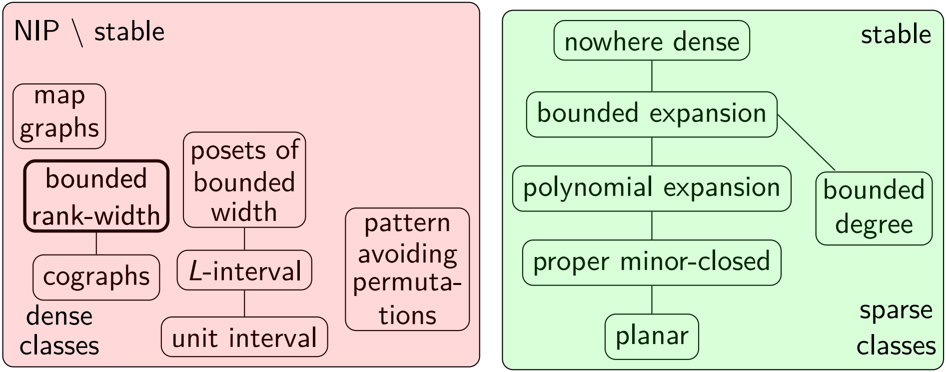
# Classes with known tractable FO model checking




New program: transductions of nowhere dense classes

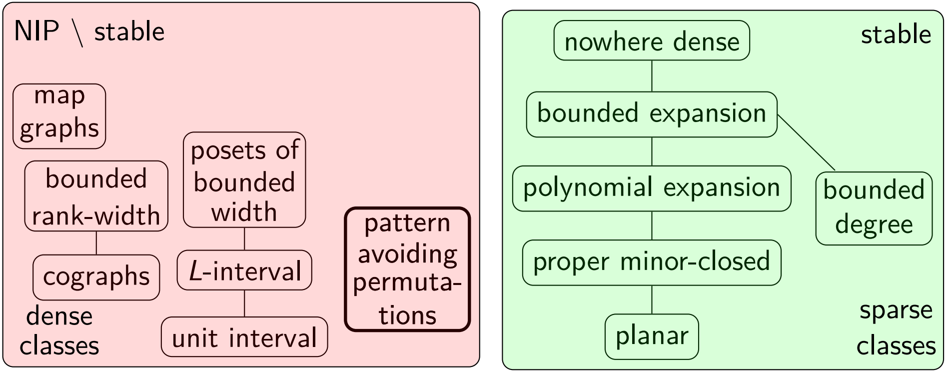
Not sparse anymore but still stable

# Classes with known tractable FO model checking



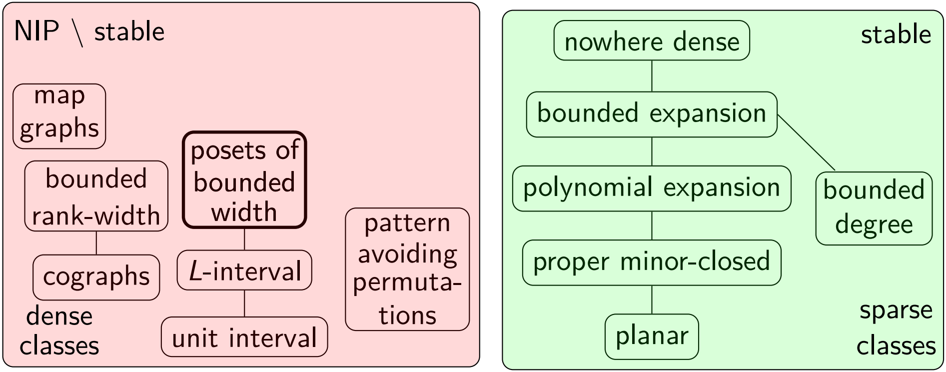

$\mathrm{MSO}_1$ MODEL CHECKING solvable in $f(|\varphi|, w)n$ on graphs of rank-width $w$ [Courcelle, Makowsky, Rotics '00]

# Classes with known tractable FO model checking



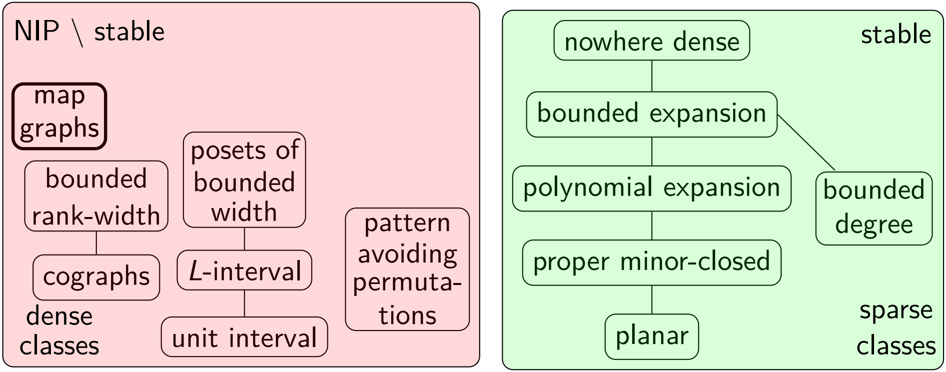

*Is $\sigma$ a subpermutation of $\tau$?* solvable in $f(|\sigma|)|\tau|$
[Guillemot, Marx '14]

# Classes with known tractable FO model checking



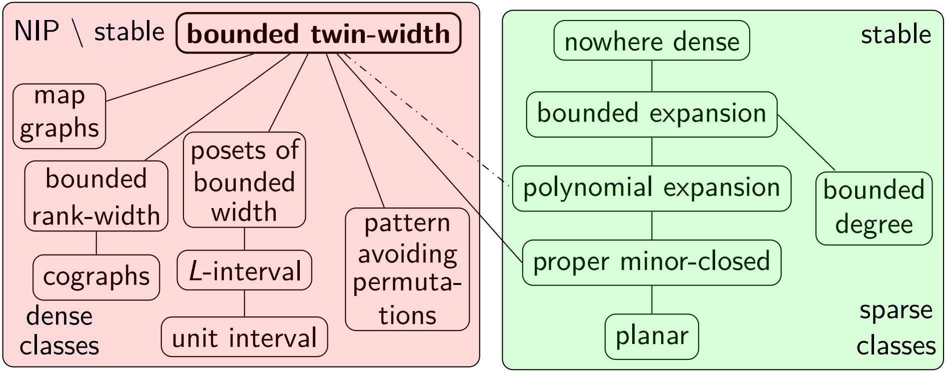

FO MODEL CHECKING solvable in $f(|\varphi|, w)n^2$ on posets of width $w$
[GHLOORS '15]

# Classes with known tractable FO model checking



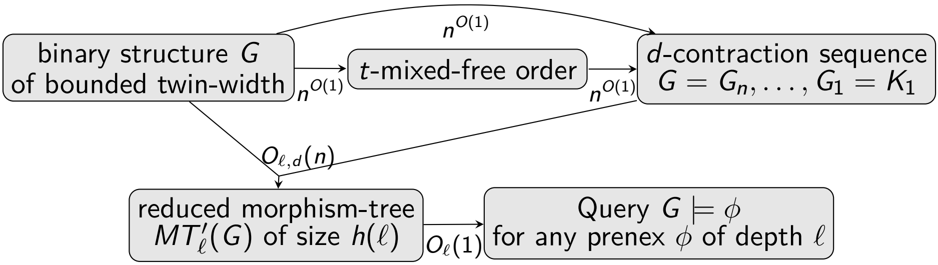

FO MODEL CHECKING solvable in $f(|\varphi|)n^{O(1)}$ on map graphs
[Eickmeyer, Kawarabayashi '17]

# Classes with known tractable FO model checking

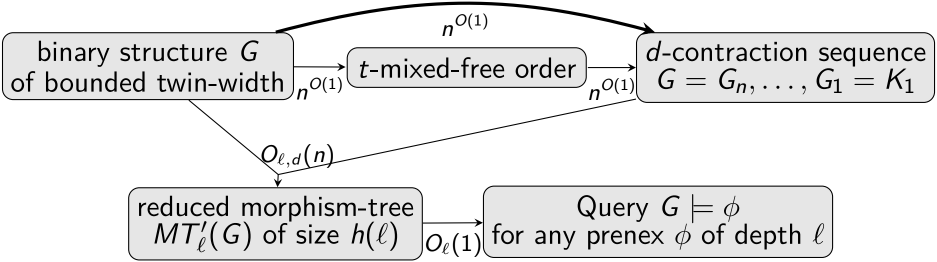



FO MODEL CHECKING solvable in $f(|\varphi|, d)n$ on graphs with a $d$-sequence
[B, Kim, Thomassé, Watrigant '20+]

# Workflow of our FO model checking algorithm

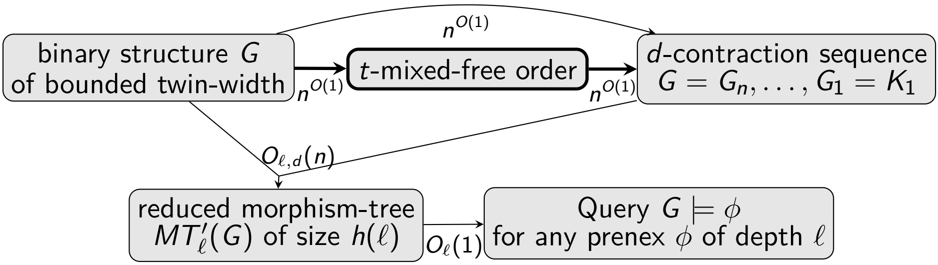


# Workflow of our FO model checking algorithm



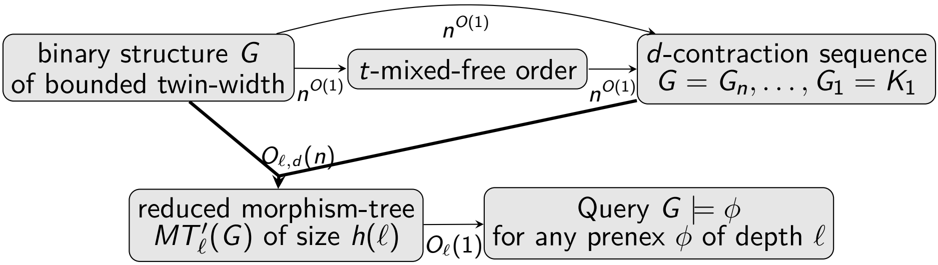

Direct examples: **trees**, bounded rank-width, **grids**, $d$-dimensional grids, unit interval graphs, $K_t$-free unit ball graphs

# Workflow of our FO model checking algorithm



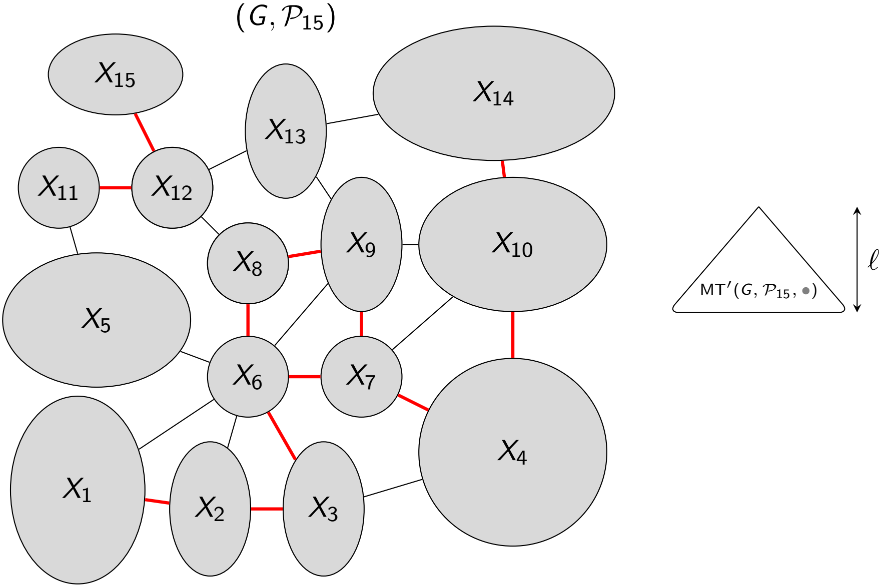

We now explore the detour via mixed minor for:
pattern-avoiding permutations, bounded width posets, $K_t$-minor free graphs

# Workflow of our FO model checking algorithm

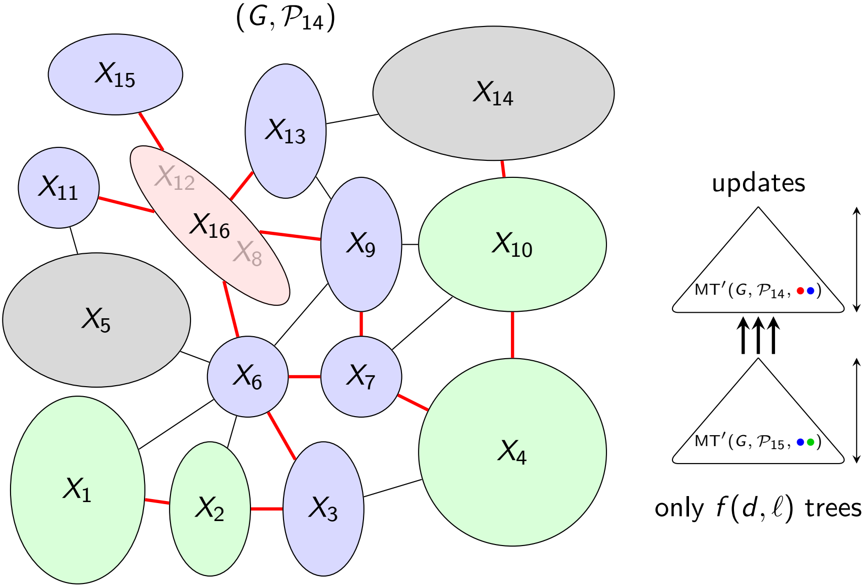



But before we give a snapshot of the FO model checking

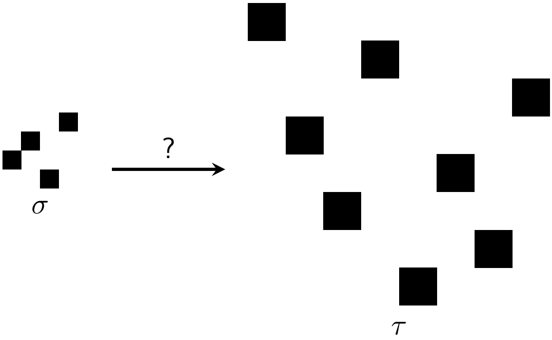
# DP for FO model checking with $d$-sequence



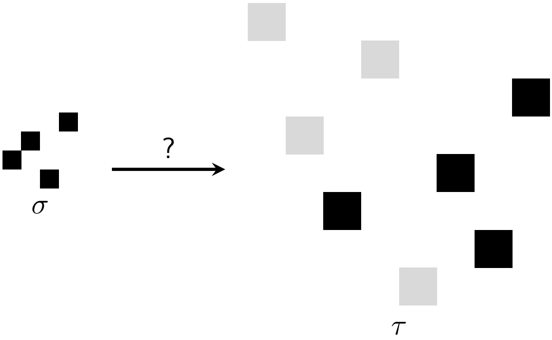
# DP for FO model checking with $d$-sequence



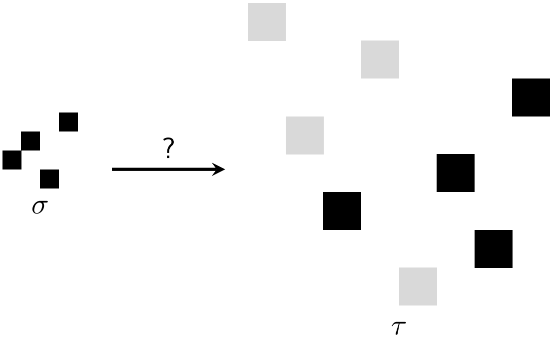
# Permutation Pattern



# Permutation Pattern



# Permutation Pattern




Theorem (Guillemot, Marx '14)

Permutation Pattern *can be solved in time* $2^{|\sigma|^2}|\tau|$.

# Guillemot and Marx's win-win algorithm

**Theorem (Marcus, Tardos '04)**

*$\forall t$, $\exists c_t$ $\forall$ $n \times n$ 0,1-matrix with $\geqslant c_t n$ entries 1 has a t-grid minor.*

$$\text{4-grid minor} \left[\begin{array}{cc|cc|cc|cc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
\hline
0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
\hline
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 & 0 & 1
\end{array}\right]$$

# Guillemot and Marx's win-win algorithm

**Theorem (Marcus, Tardos '04)**

*$\forall t$, $\exists c_t$ $\forall$ $n \times n$ 0,1-matrix with $\geqslant c_t n$ entries 1 has a $t$-grid minor.*

$$\text{4-grid minor} \quad \left[\begin{array}{cc|cc|cc|cc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ \hline 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ \hline 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

A) $\geqslant c_{|\sigma|} n$ entries 1 $\rightarrow$ YES from the $|\sigma|$-grid minor.

B) $< c_{|\sigma|} n$ entries 1 $\rightarrow$ merge of two "similar" rectangles

# Guillemot and Marx's win-win algorithm

**Theorem (Marcus, Tardos '04)**

*$\forall t$, $\exists c_t$ $\forall$ $n \times n$ 0,1-matrix with $\geqslant c_t n$ entries 1 has a t-grid minor.*

$$\text{4-grid minor} \quad \left[\begin{array}{cc|cc|cc|cc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ \hline 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ \hline 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

A) $\geqslant c_{|\sigma|} n$ entries 1 $\rightarrow$ YES from the $|\sigma|$-grid minor.
B) $< c_{|\sigma|} n$ entries 1 $\rightarrow$ merge of two "similar" rectangles

If B) always happens $\rightarrow$ DP on this merge sequence

# Our generalization to the dense case – mixed minor

Mixed zone: 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

# Our generalization to the dense case – mixed minor

Mixed zone: 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

# Grid minor theorem for twin-width

Theorem (B, Kim, Thomassé, Watrigant 20+)

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

# Grid minor theorem for twin-width

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 $\mathcal{C}$

# Grid minor theorem for twin-width

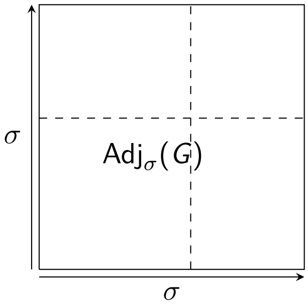
Theorem (B, Kim, Thomassé, Watrigant 20+)

*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 $\mathcal{C}$




Cutting after the $t/2$-th division of the $t$-grid minor

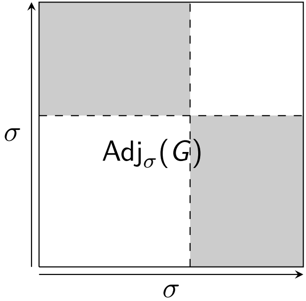
# Grid minor theorem for twin-width

Theorem (B, Kim, Thomassé, Watrigant 20+)

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

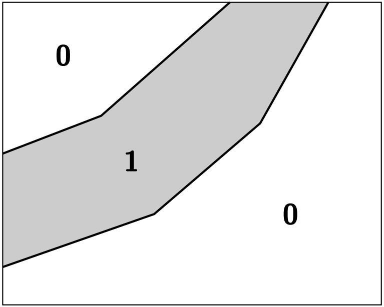
Now to bound the twin-width of a class $\mathcal{C}$:

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



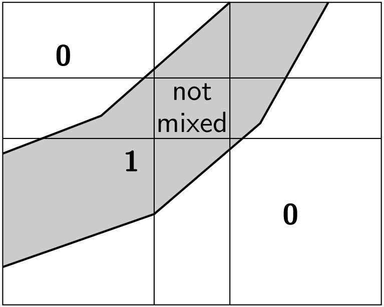

One of the shaded areas contains a $t/2$-grid minor on disjoint sets

# Bounded twin-width – posets of bounded antichain




Warm-up with unit interval graphs: order by left endpoints

# Bounded twin-width – posets of bounded antichain

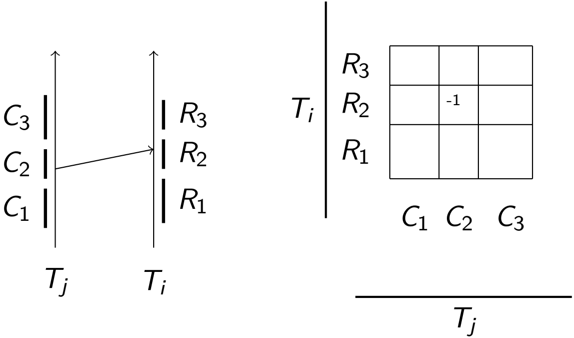



No 3-by-3 grid has all 9 cells crossed by two non-decreasing curves

# Bounded twin-width – posets of bounded antichain

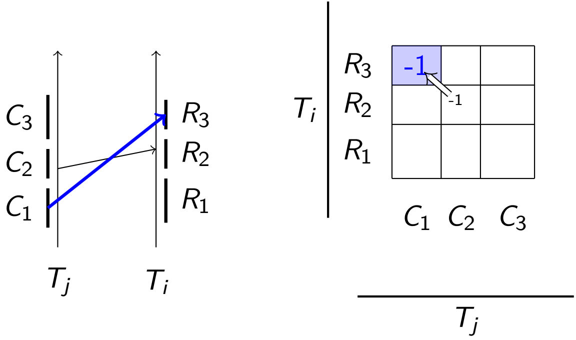

Put the $k$ chains in order one after the other

# Bounded twin-width – posets of bounded antichain



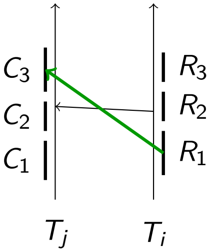

A $3k$-mixed minor implies a 3-mixed minor between two chains

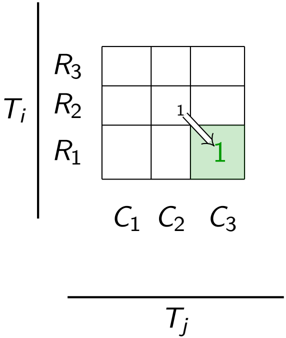
# Bounded twin-width – posets of bounded antichain




Transitivity implies that a zone is constant

# Bounded twin-width – posets of bounded antichain



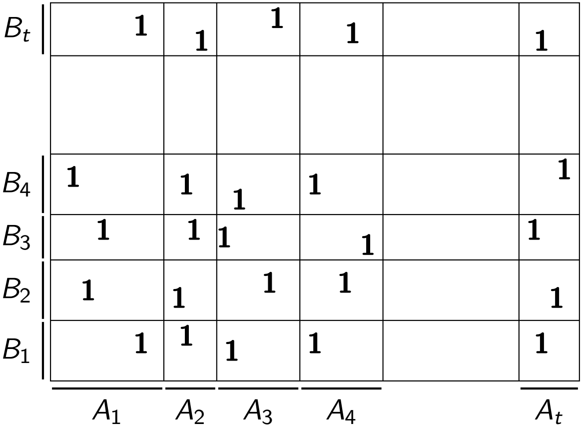




And symmetrically

# Bounded twin-width – $K_t$-minor free graphs

Given a hamiltonian path, we would just use this order

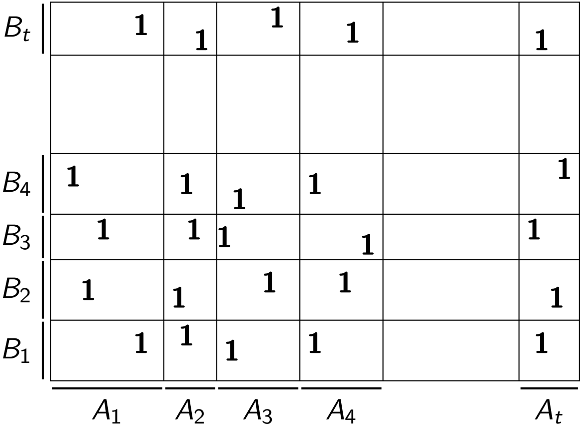
# Bounded twin-width – $K_t$-minor free graphs




Contracting[1] the $2t$ subpaths yields a $K_{t,t}$-minor

[1] Here it is an actual contration, not a mere identification

# Bounded twin-width – $K_t$-minor free graphs




Instead we use a specially crafted lex-DFS discovery order

# Small classes

Classes[1] with at most $n!c^n$ labeled graphs on $[n]$.

Theorem (B, Geniet, Kim, Thomassé, Watrigant 20+)

*Bounded twin-width classes are small.*

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

[1] sets closed by taking induced subgraphs

# Small classes

Classes[1] with at most $n!c^n$ labeled graphs on $[n]$.

Theorem (B, Geniet, Kim, Thomassé, Watrigant 20+)

*Bounded twin-width classes are small.*

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

[1] sets closed by taking induced subgraphs

# Small classes

Classes[1] with at most $n!c^n$ labeled graphs on $[n]$.

Theorem (B, Geniet, Kim, Thomassé, Watrigant 20+)

*Bounded twin-width classes are small.*

Is the converse true?

Conjecture (small conjecture)

*A class has bounded twin-width if and only if it is small.*

[1] sets closed by taking induced subgraphs

# Future directions

**Obvious questions:**
Algorithm to compute/approximate twin-width in general
Fully classify classes with tractable FO model checking
Small conjecture, polynomial expansion

# Future directions

**Obvious questions:**
Algorithm to compute/approximate twin-width in general
Fully classify classes with tractable FO model checking
Small conjecture, polynomial expansion

**Other directions we are exploring:**
Better approximation algorithms on bounded twin-width classes
Extended nested dissection to bounded twin-width
Twin-width of groups
⋮

# Future directions

**Obvious questions:**
Algorithm to compute/approximate twin-width in general
Fully classify classes with tractable FO model checking
Small conjecture, polynomial expansion

**Other directions we are exploring:**
Better approximation algorithms on bounded twin-width classes
Extended nested dissection to bounded twin-width
Twin-width of groups
⋮

On arxiv
Twin-width I: tractable FO model checking [BKTW '20]
Twin-width II: small classes [BGKTW '20]
Twin-width III: Max Independent Set and Coloring [BGKTW '20]