Szymon Toruńczyk

University of Warsaw

STRUCTURALLY TRACTABLE GRAPH CLASSES

Szymon Toruńczyk

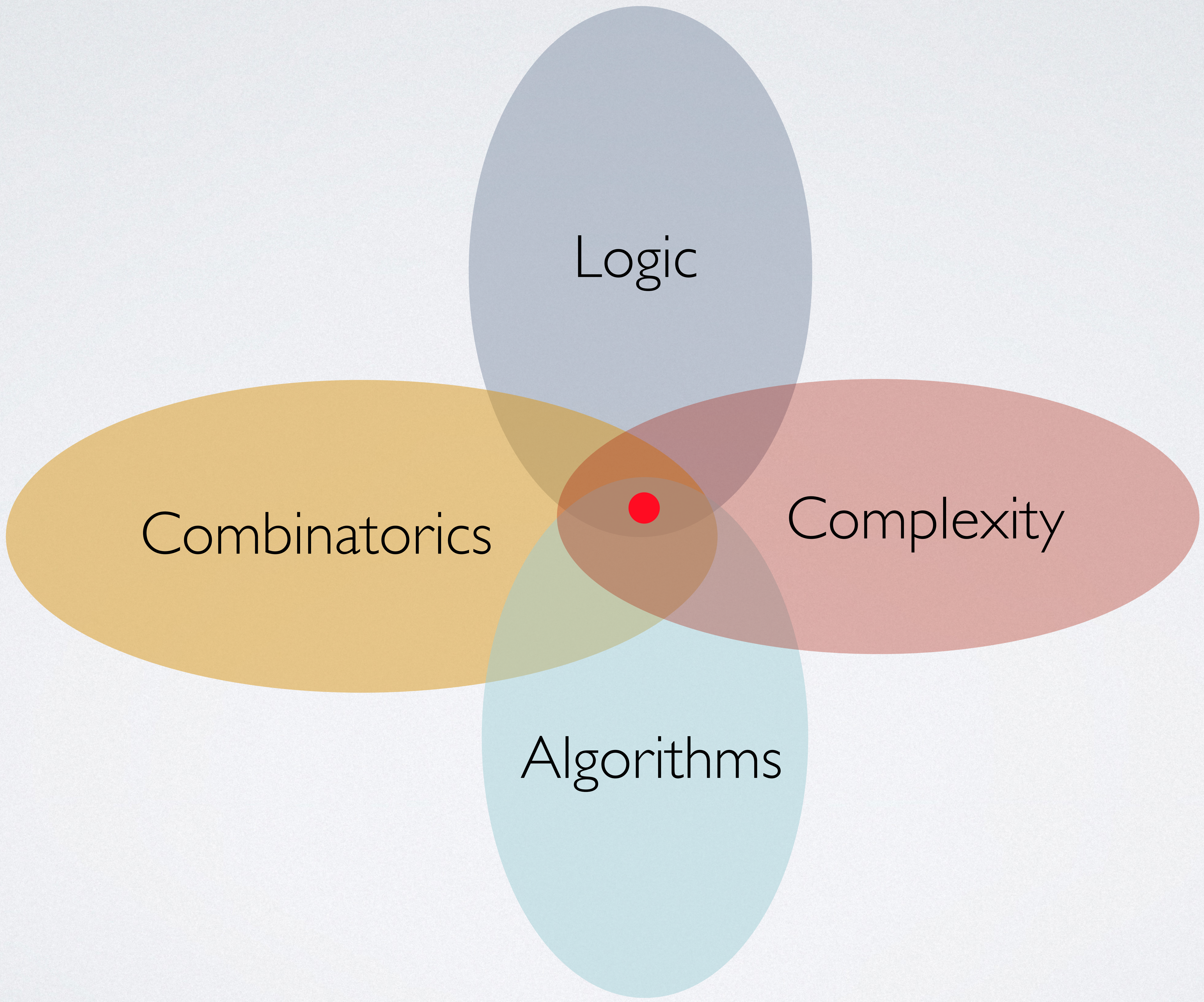
University of Warsaw

MONADICALLY DEPENDENT GRAPH CLASSES

ALGORITHMIC MODEL THEORY

algorithmic
model
theory

ALGORITHMIC MODEL THEORY



OUTLINE

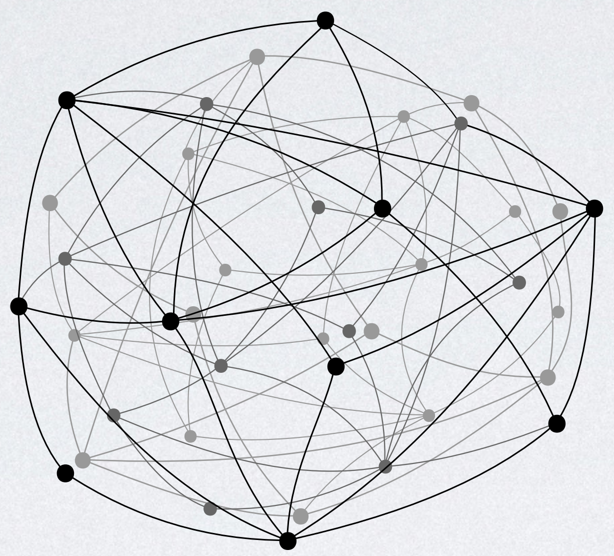
1. The model checking problem
2. Sparsity: monotone case
3. Twin-width: ordered case
4. Monadic dependence
5. Flip-breakability
6. Stability: orderless case

OUTLINE

1. **The model checking problem**
2. Sparsity: monotone case
3. Twin-width: ordered case
4. Monadic dependence
5. Flip-breakability
6. Stability: orderless case

FIRST-ORDER MODEL CHECKING

FIRST-ORDER MODEL CHECKING

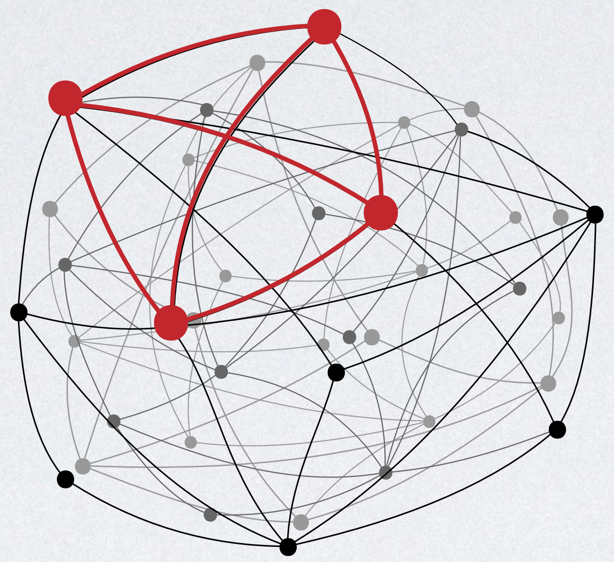


$$\overset{?}{\models} \quad \varphi$$

structure

first-order formula

FIRST-ORDER MODEL CHECKING



$\overset{?}{\models}$ φ

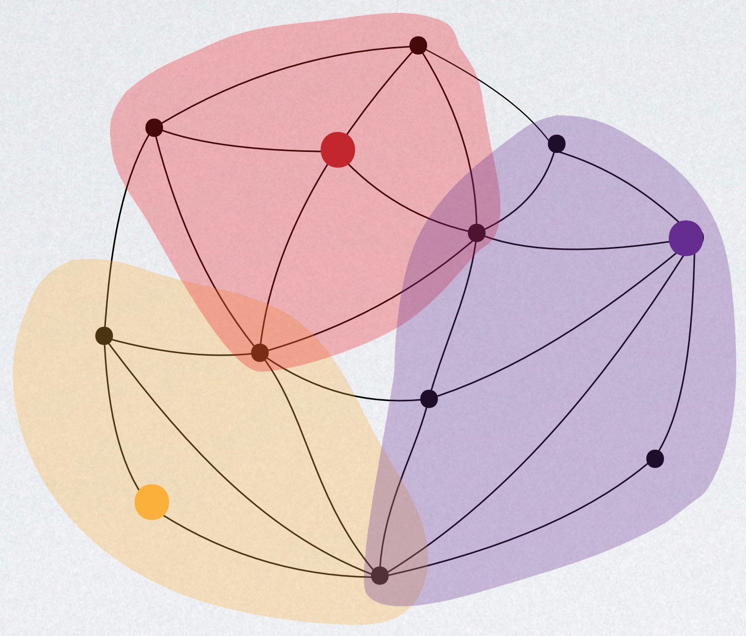
structure

first-order formula

e.g. $\varphi = \exists x.\ \exists y.\ \exists z.\ \exists t.\ (x{\sim}y) \wedge (y{\sim}z) \wedge (z{\sim}t) \wedge (t{\sim}x) \wedge (x{\sim}z) \wedge (y{\sim}t)$

"Is there a clique of size 4?"

FIRST-ORDER MODEL CHECKING



$\overset{?}{\models} \quad \varphi$

structure

first-order formula

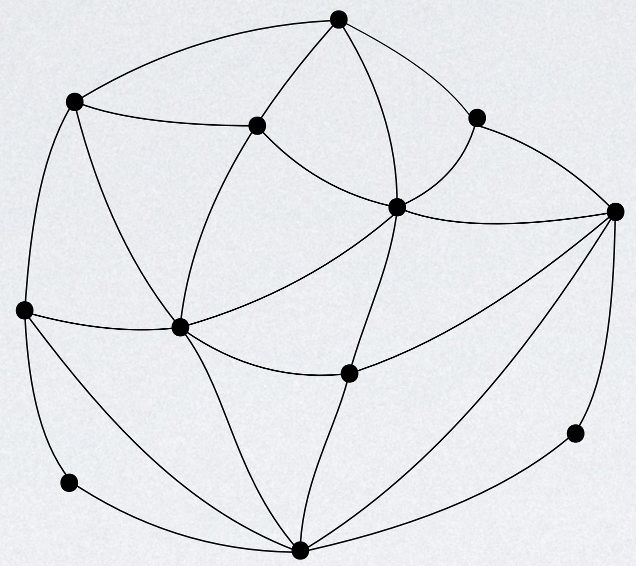
e.g. $\varphi = \exists x.\, \exists y.\, \exists z.\, \exists t.\, (x{\sim}y) \wedge (y{\sim}z) \wedge (z{\sim}t) \wedge (t{\sim}x) \wedge (x{\sim}z) \wedge (y{\sim}t)$

"Is there a clique of size 4?"

e.g. $\varphi = \exists x.\, \exists y.\, \exists z.\, \forall t.\, (x{\sim}t) \vee (y{\sim}t) \vee (z{\sim}t)$

"Are there 3 nodes whose neighborhoods include all nodes?"

FIRST-ORDER MODEL CHECKING



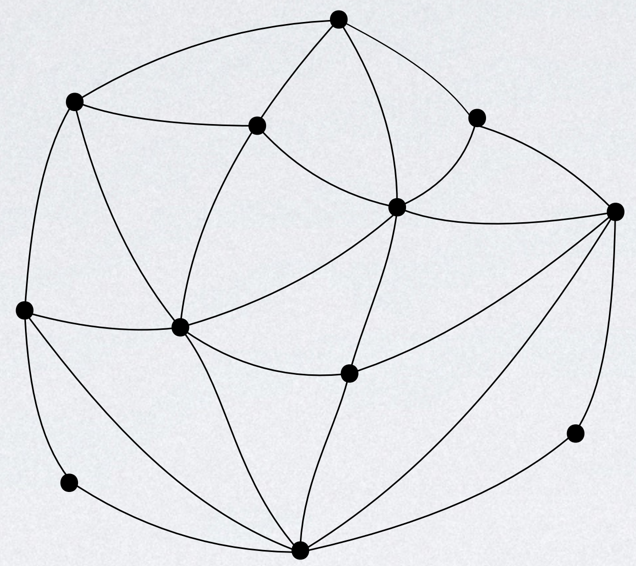
$\overset{?}{\models} \quad \varphi$

structure

first-order
formula

A fundamental problem in TCS

FIRST-ORDER MODEL CHECKING



$\overset{?}{\models} \quad \varphi$

structure

first-order formula

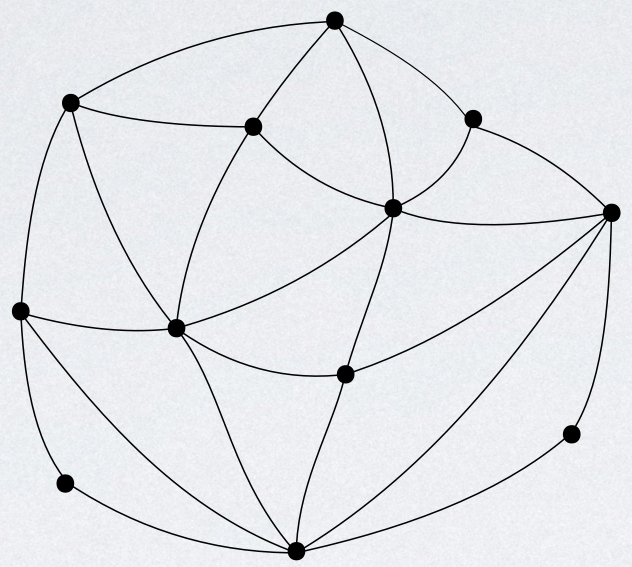
A fundamental problem in TCS

Central in

→ database theory

→ software verification (for other logics)

FIRST-ORDER MODEL CHECKING

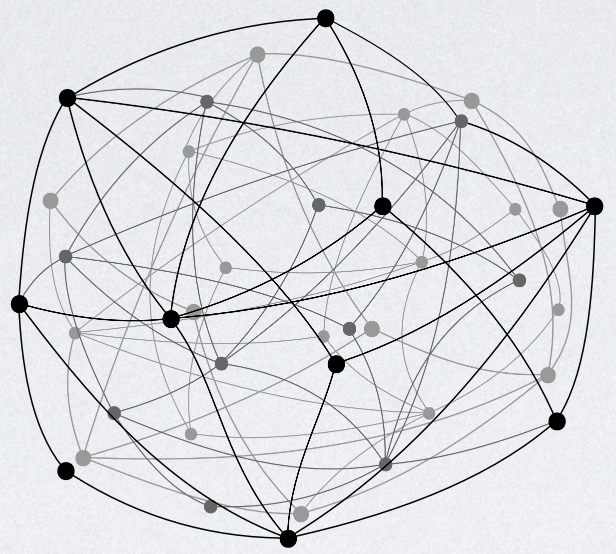


structure

φ

first-order
formula

QUERY EVALUATION



database

Q

SQL query

```
SELECT e1.x AS a, e2.x AS b, e3.x AS c, e4.x AS d
FROM edges e1 JOIN edges e2 ON e2.x = e1.x
JOIN edges e3 ON e3.x = e1.x
JOIN edges e4 ON e4.y = e1.y
JOIN edges e5 ON e5.x = e1.y
JOIN edges e6 ON e6.x = e2.y AND e6.y = e1.y
```

select all 4-cliques

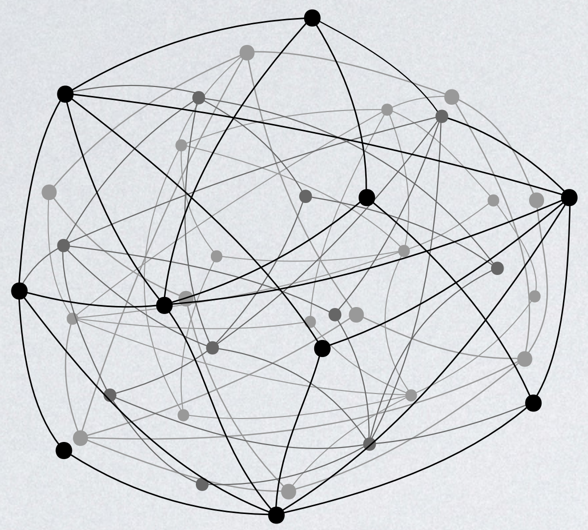
Goal: efficiently evaluate queries in large databases

FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY

FIRST-ORDER MODEL CHECKING

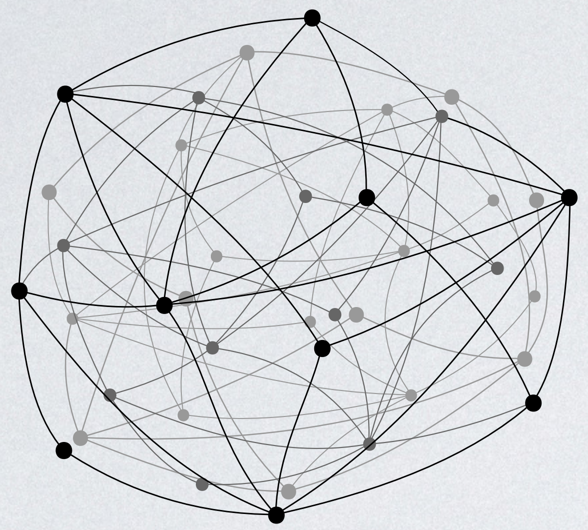
PARAMETERISED COMPLEXITY



General input graphs *highly intractable*: AW[$*$]-hard

FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



General input graphs *highly intractable*: AW[$*$]-hard

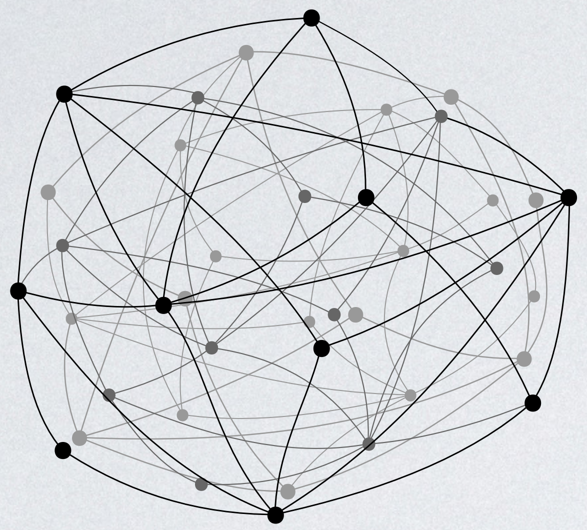
Does an n-vertex graph contain a clique of size k=4 ?

Naive algorithm: $O(n^k) = O(n^4)$ time

Best known algorithm: $O(n^{0.79k}) = O(n^{3.16})$ time

FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



General input graphs *highly intractable*: AW[$*$]-hard

Does an n-vertex graph contain a clique of size k=4 ?

Naive algorithm: $O(n^k) = O(n^4)$ time

Best known algorithm: $O(n^{0.79k}) = O(n^{3.16})$ time

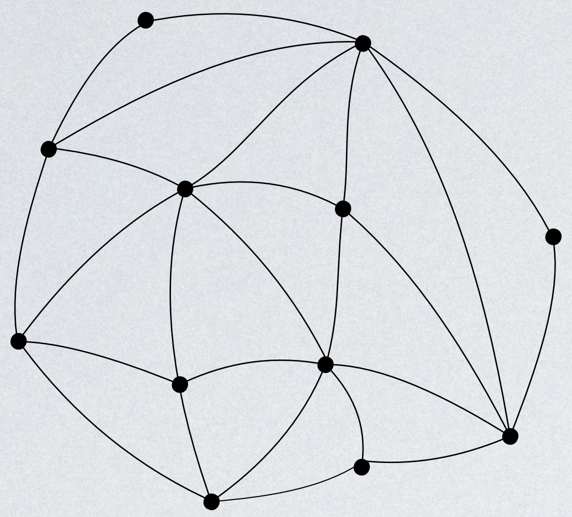
Impractical for n=100,000

FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY

FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY

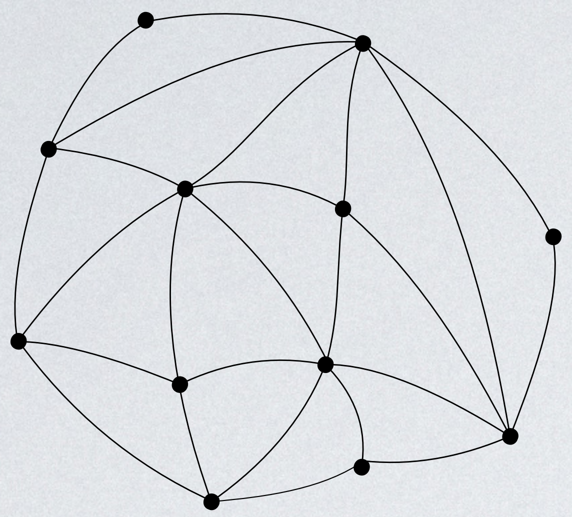


Theorem [Frick, Grohe, 2001]

Every first-order property can be tested in **linear time** on planar graphs.

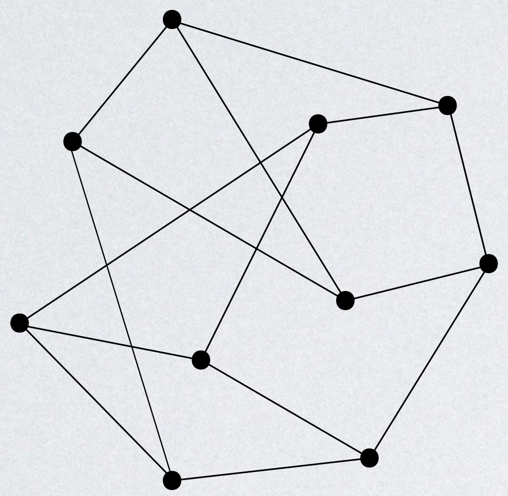
FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



Theorem [Frick, Grohe, 2001]

Every first-order property can be tested in **linear time** on planar graphs.

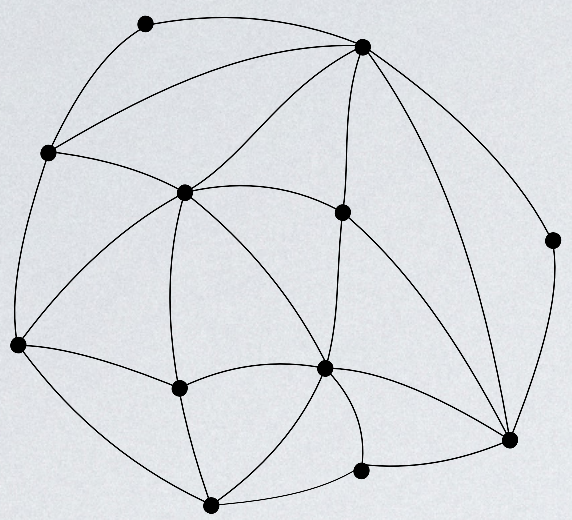


Theorem [Seese, 1996].

Same for graphs of maximum degree ≤ Δ, for any fixed Δ

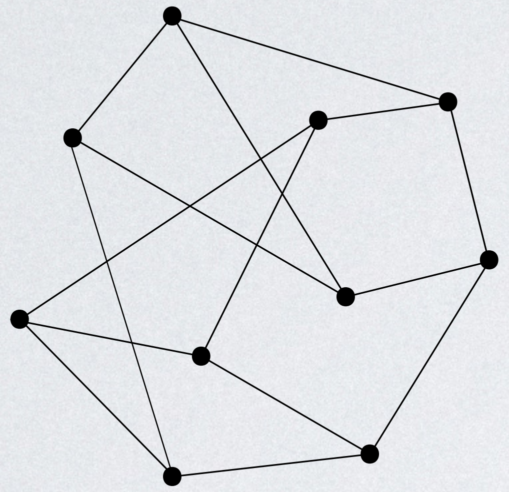
FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



Theorem [Frick, Grohe, 2001]

Every first-order property can be tested in **linear time** on planar graphs.



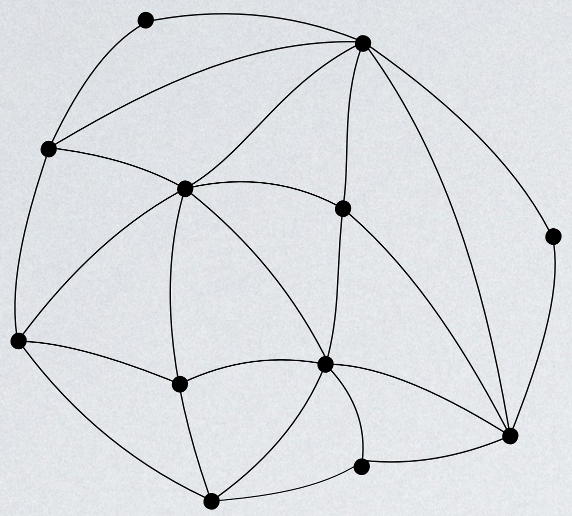
Theorem [Seese, 1996].

Same for graphs of maximum degree $\leq \Delta$, for any fixed Δ

Definition. FO model checking is *fixed-parameter tractable (fpt)* on a graph class C if

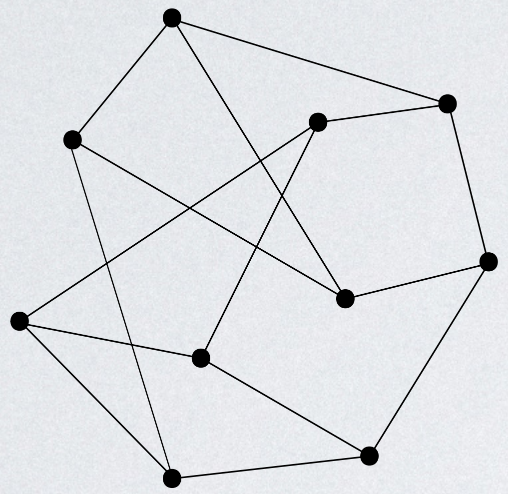
FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



Theorem [Frick, Grohe, 2001]

Every first-order property can be tested in **linear time** on planar graphs.



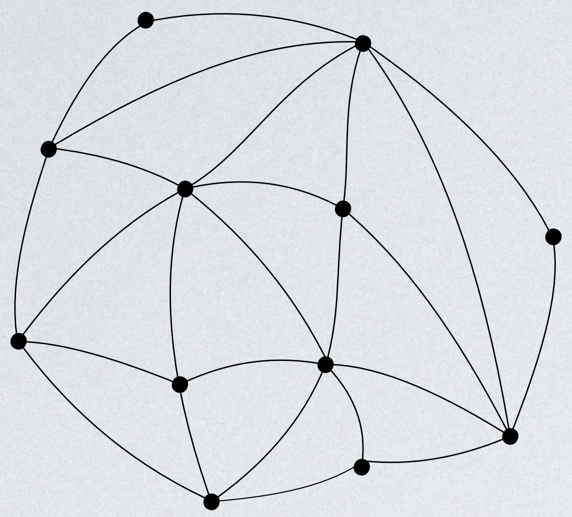
Theorem [Seese, 1996].

Same for graphs of maximum degree ≤ Δ, for any fixed Δ

Definition. FO model checking is *fixed-parameter tractable (fpt)* on a graph class *C* if for every $\phi \in$FO and graph $G \in C$,

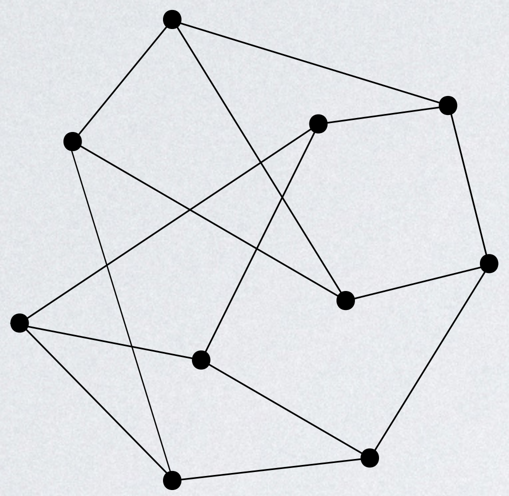
FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



Theorem [Frick, Grohe, 2001]

Every first-order property can be tested in **linear time** on planar graphs.



Theorem [Seese, 1996].

Same for graphs of maximum degree $\leq \Delta$, for any fixed Δ

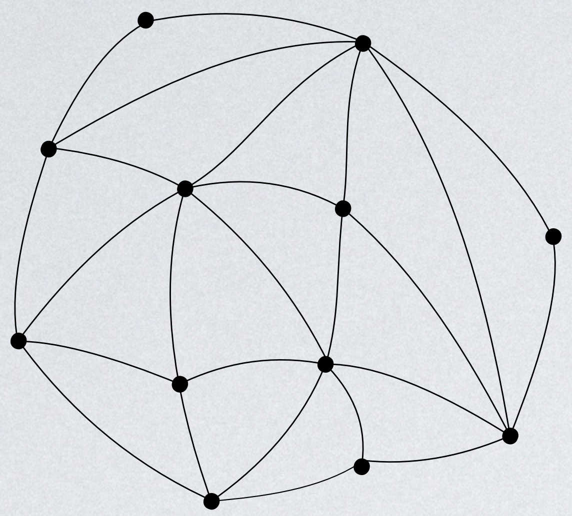
Definition. FO model checking is *fixed-parameter tractable (fpt)* on a graph class C if

for every $\phi \in$FO and graph $G \in C$,

$G \models \phi$ can be tested in time

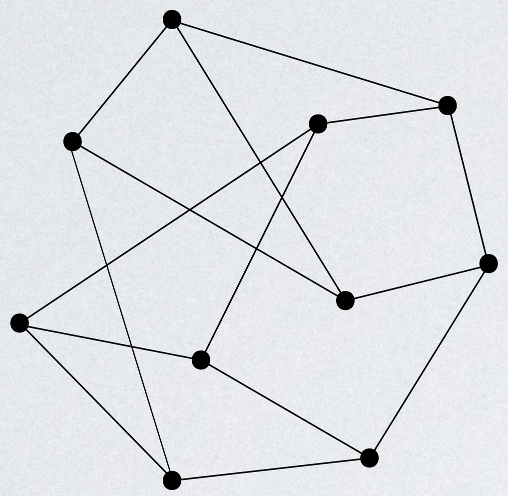
FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



Theorem [Frick, Grohe, 2001]

Every first-order property can be tested in **linear time** on planar graphs.



Theorem [Seese, 1996].

Same for graphs of maximum degree $\leq \Delta$, for any fixed Δ

Definition. FO model checking is *fixed-parameter tractable (fpt)* on a graph class C if

for every $\phi \in \mathrm{FO}$ and graph $G \in C$,

$G \models \phi$ can be tested in time

$$O_{\phi,C}(\ |G|^{d}\ )$$

constant d independent of ϕ.

MSO MODEL CHECKING

MSO MODEL CHECKING

Courcelle's theorem (1990)

MSO MODEL CHECKING

Courcelle's theorem (1990)

Every **monadic second-order** property can be tested in **linear time:**

MSO MODEL CHECKING

Courcelle's theorem (1990)

Every **monadic second-order** property can be tested in **linear time:**

For every $\phi \in$ MSO, $k \geq 1$, graph G of treewidth k,

$G \vDash \phi$ can be tested in time $O_{\phi,k}(\ |G|\ )$

MSO MODEL CHECKING

Courcelle's theorem (1990)

Every **monadic second-order** property can be tested in **linear time:**

For every $\phi \in$ MSO, $k \geq 1$, graph G of treewidth k,

$G \models \phi$ can be tested in time $O_{\phi,k}(\ |G|\ )$

Ingredients:

MSO MODEL CHECKING

Courcelle's theorem (1990)

Every **monadic second-order** property can be tested in **linear time:**

For every $\phi \in \mathrm{MSO}$, $k \geq 1$, graph G of treewidth k,

$G \models \phi$ can be tested in time $O_{\phi,k}(\ |G|\ )$

Ingredients:

1. existence of tree decompositions (by definition)

MSO MODEL CHECKING

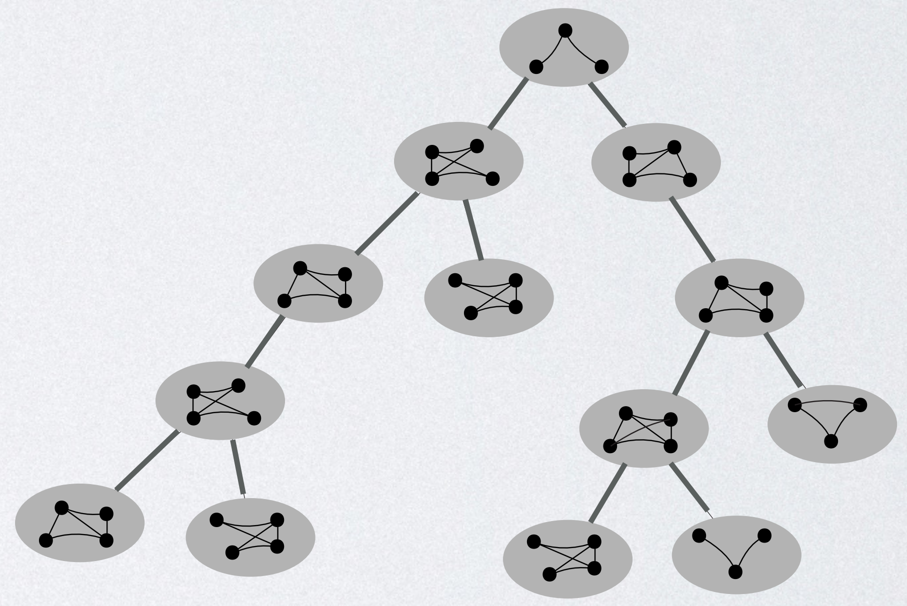
Courcelle's theorem (1990)

Every **monadic second-order** property can be tested in **linear time:**

For every $\phi \in$ MSO, $k \geq 1$, graph G of treewidth k,

$G \models \phi$ can be tested in time $O_{\phi,k}(\ |G|\ )$

Ingredients:



1. existence of tree decompositions (by definition)
2. efficient computation of decomposition (Bodlaender)

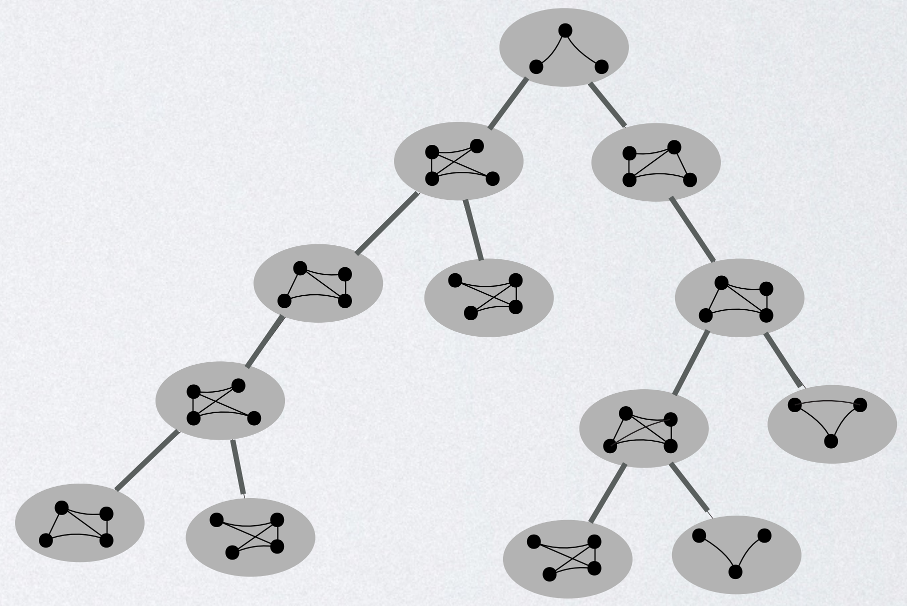
MSO MODEL CHECKING

Courcelle's theorem (1990)

Every **monadic second-order** property can be tested in **linear time:**

For every $\phi \in$ MSO, $k \geq 1$, graph G of treewidth k,

$G \models \phi$ can be tested in time $O_{\phi,k}(\ |G|\ )$



Ingredients:

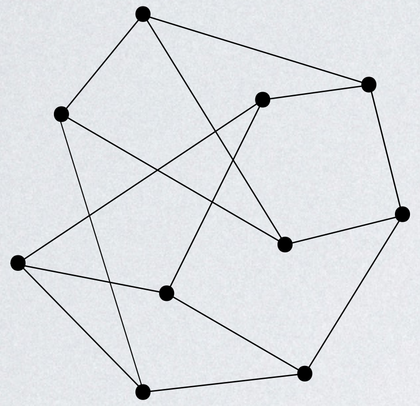
1. existence of tree decompositions (by definition)
2. efficient computation of decomposition (Bodlaender)
3. dynamic algorithm computing partial solutions to formulas (Courcelle)

FIRST-ORDER MODEL CHECKING

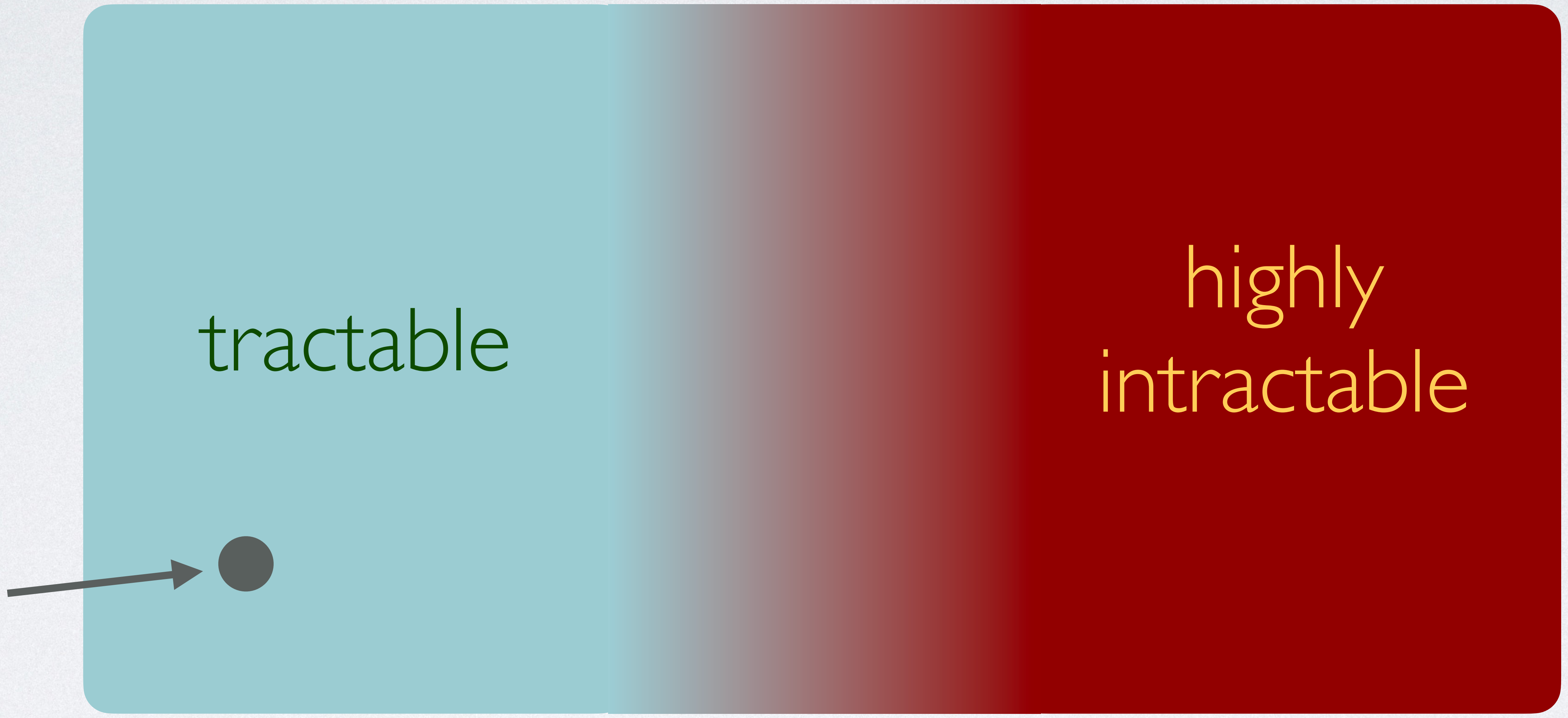
PARAMETERISED COMPLEXITY

FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY

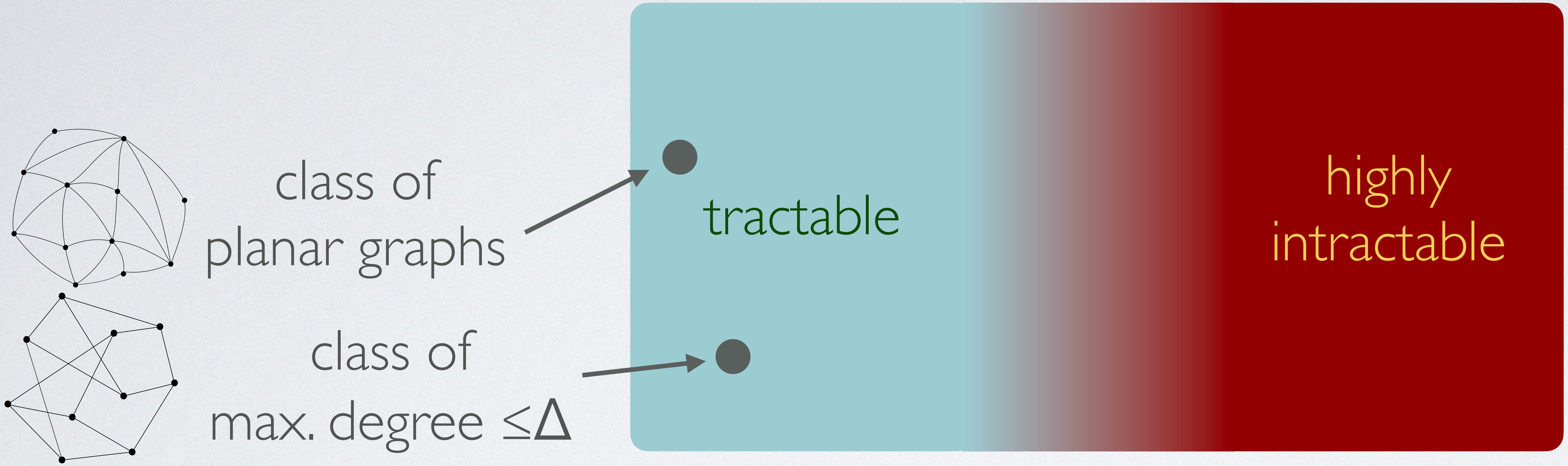


class of
max. degree $\leq\Delta$



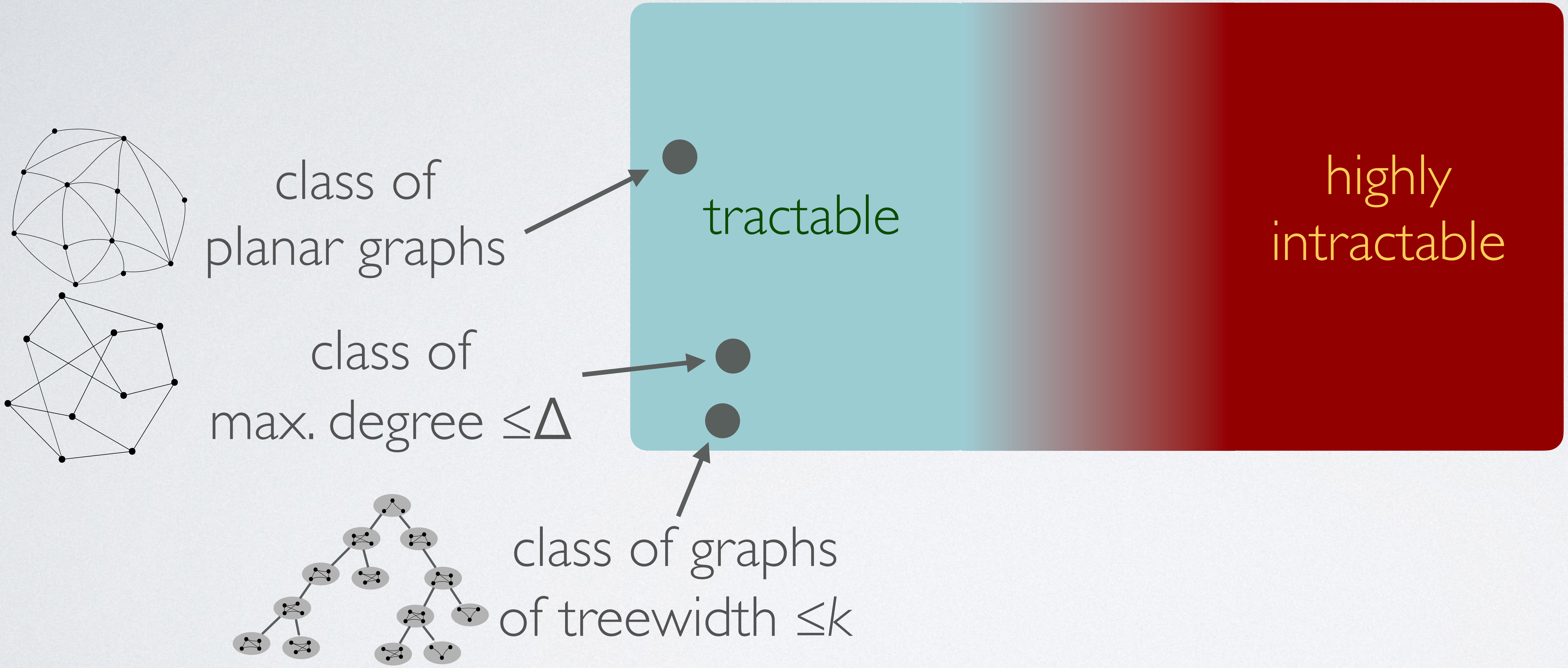
FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



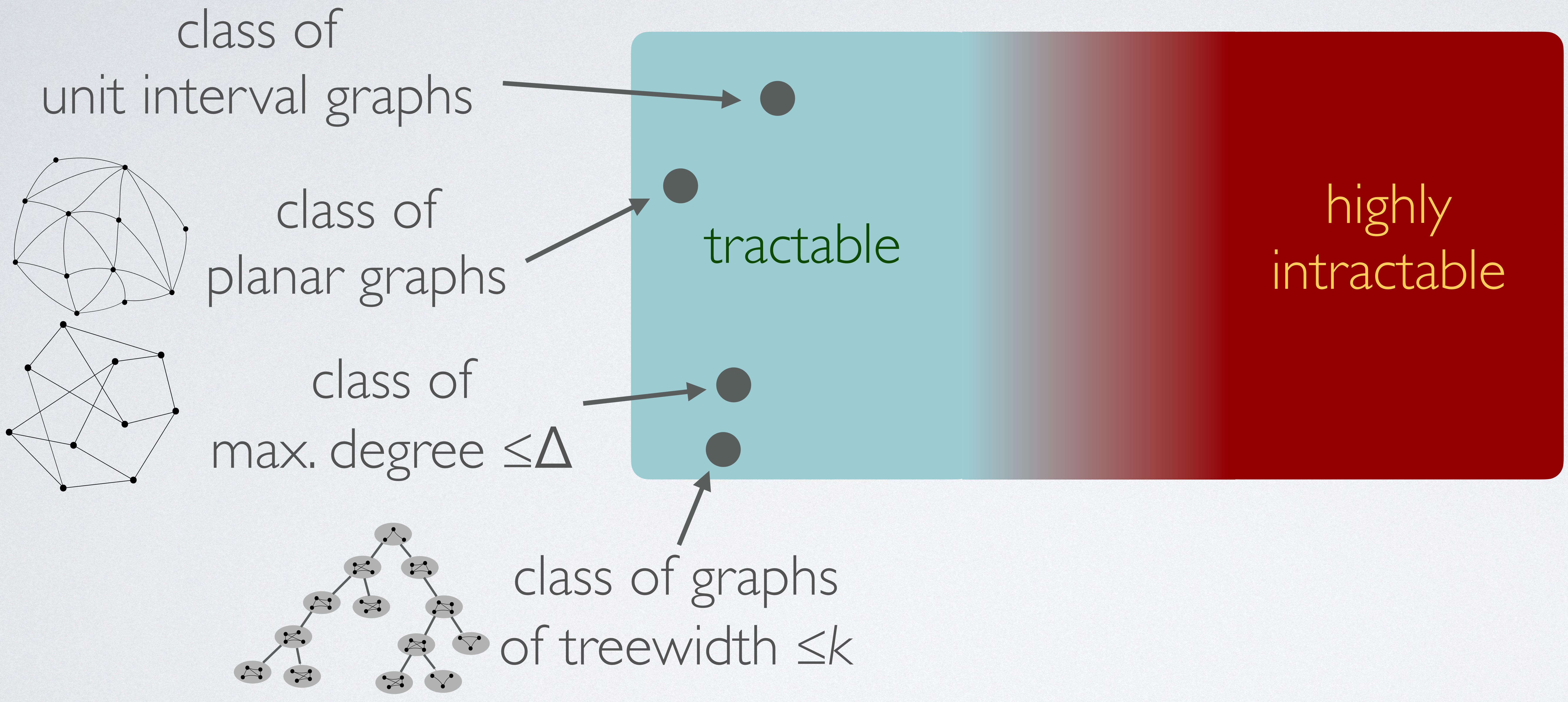
FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



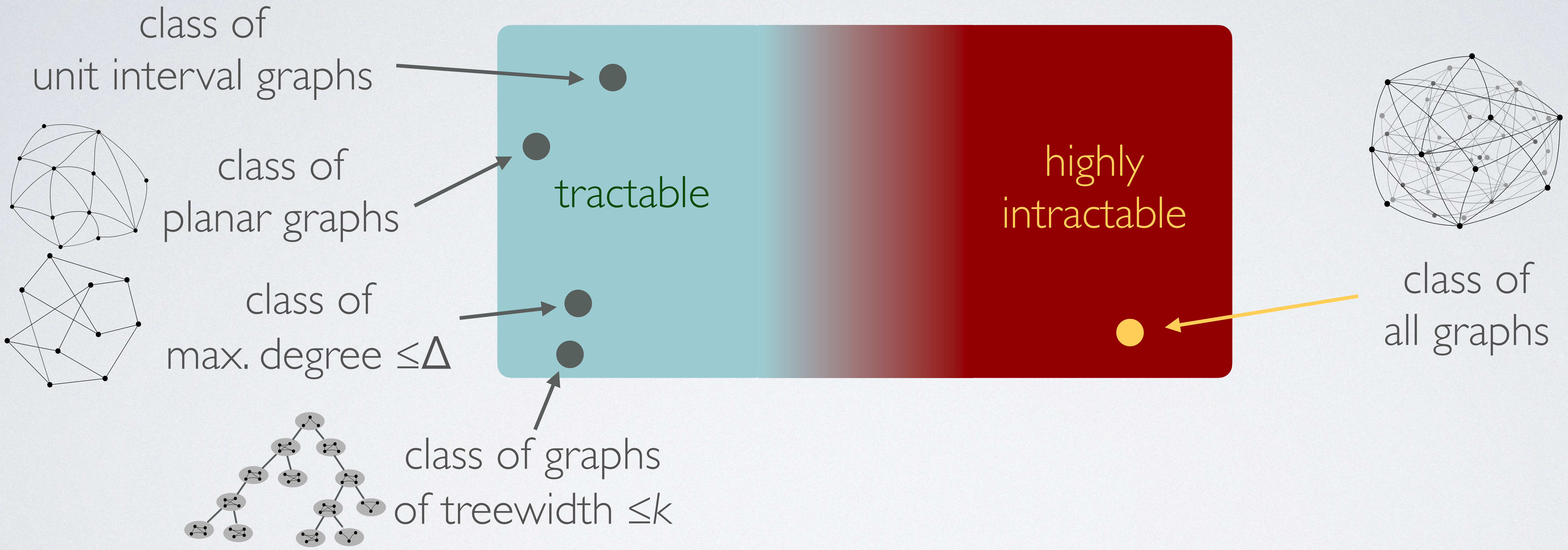
FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY

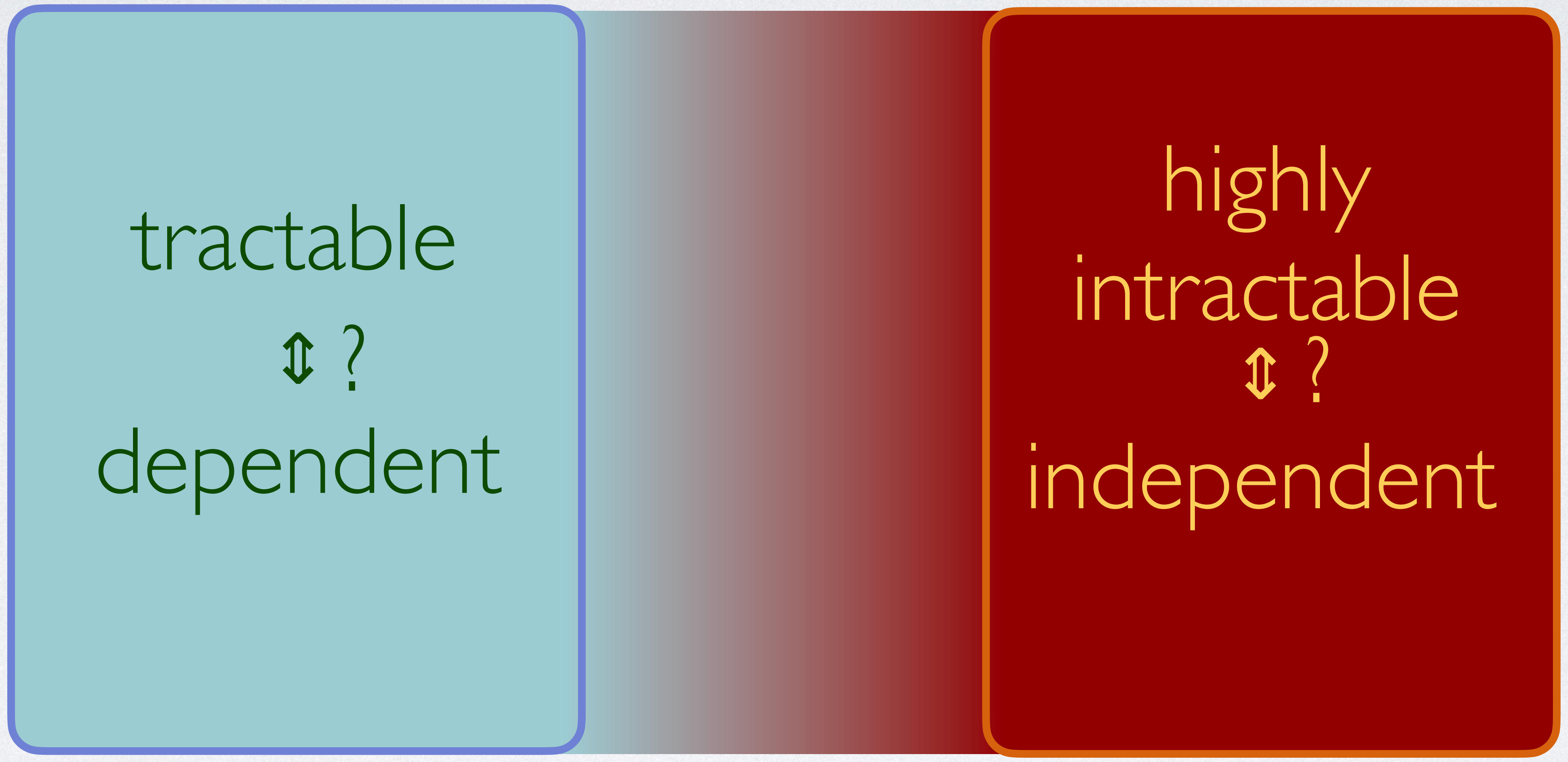
FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY

Quest: Characterise all hereditary graph classes with tractable FO model checking

FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY



Quest: Characterise all hereditary graph classes with tractable FO model checking

Conjecture: Those are exactly the *dependent* graph classes

FIRST-ORDER MODEL CHECKING

PARAMETERISED COMPLEXITY

tractable
$\Updownarrow$?
dependent

highly
intractable
$\Updownarrow$?
independent

Quest: Characterise all hereditary graph classes with tractable FO model checking

Conjecture: Those are exactly the *dependent* graph classes

HISTORY

parameterised complexity of the FO model-checking problem

HISTORY

parameterised complexity of the FO model-checking problem

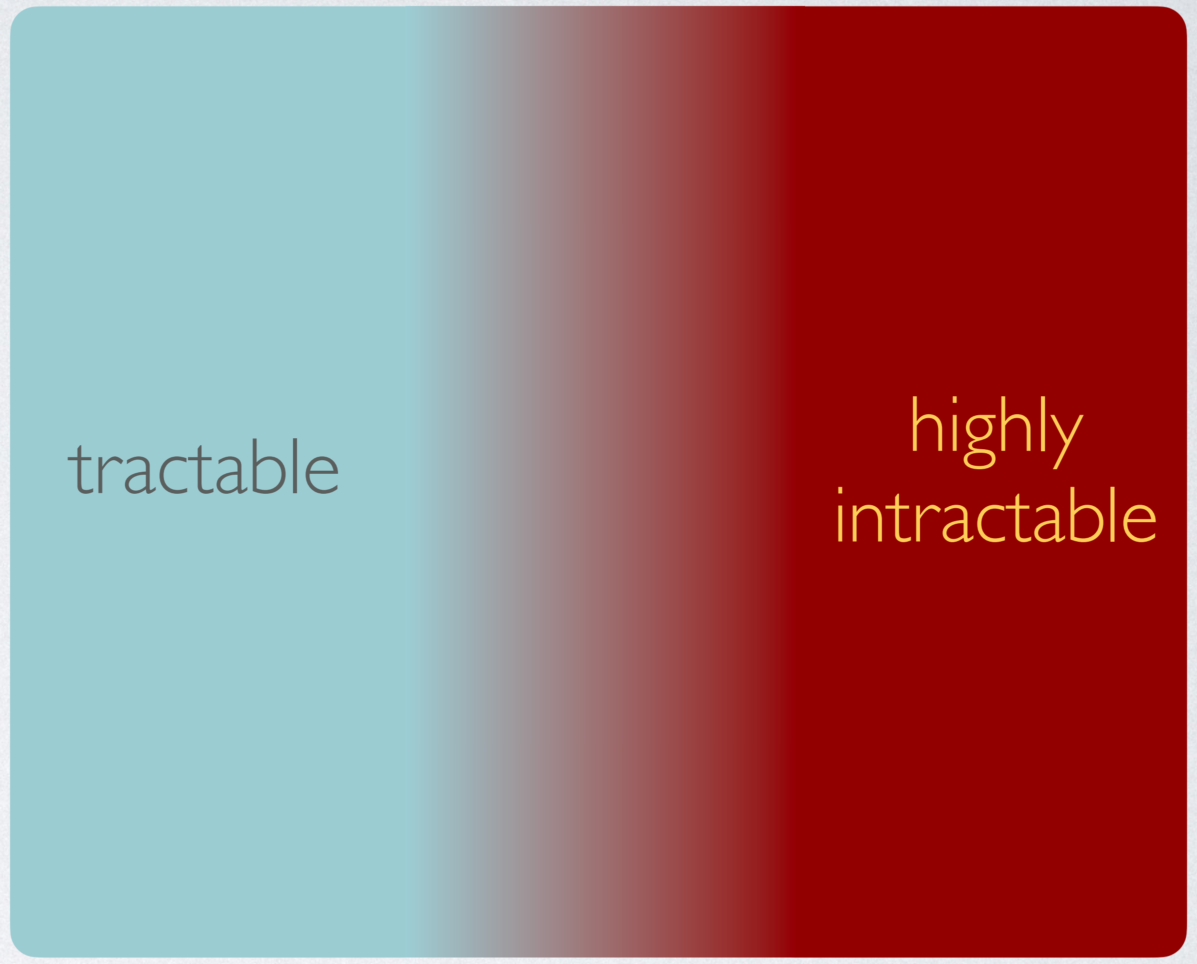
1990: Courcelle – MSO model checking on classes of bounded treewidth

HISTORY

parameterised complexity of the FO model-checking problem

1990: Courcelle – MSO model checking on classes of bounded treewidth

1999: Flum and Grohe – start systematic study via parameterised complexity

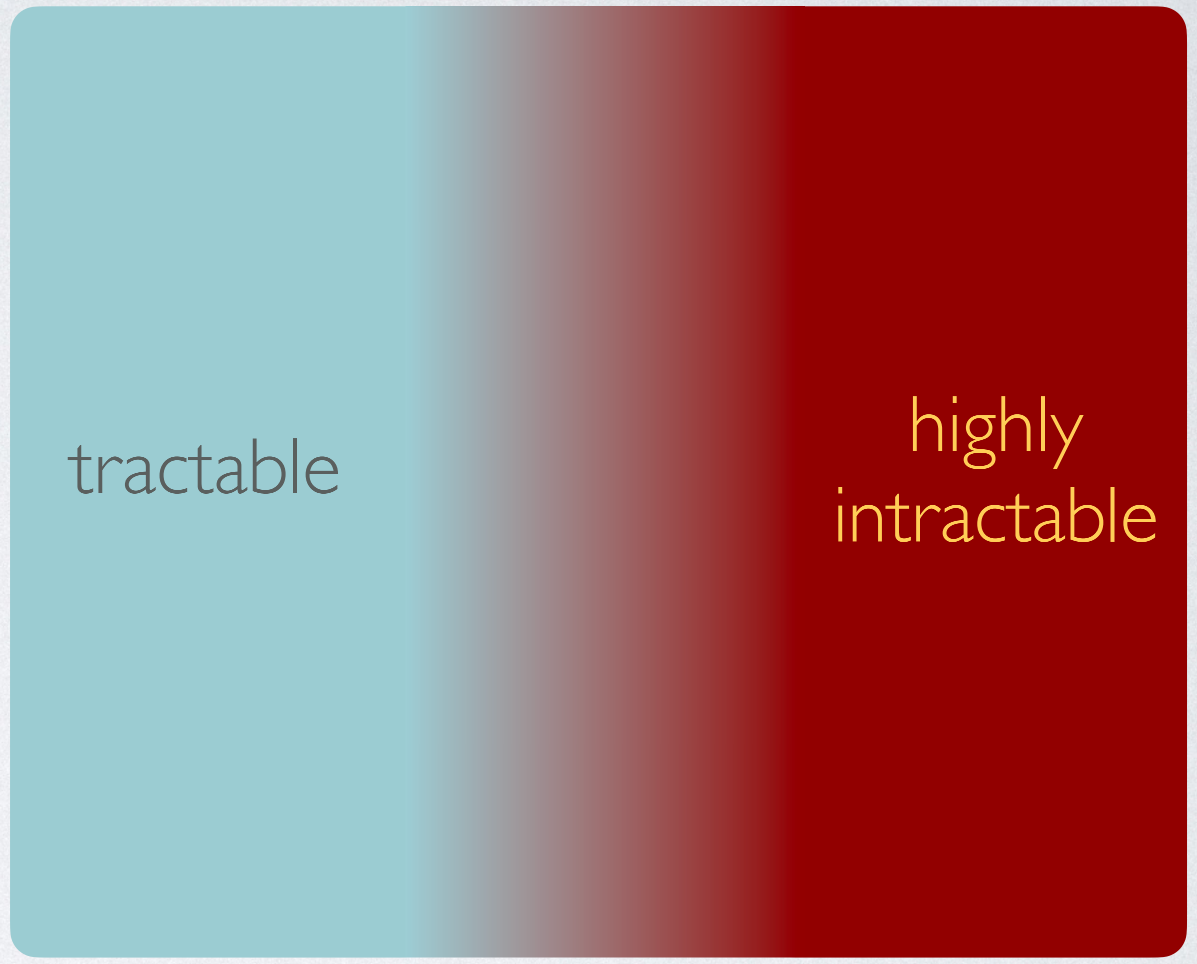


HISTORY

parameterised complexity of the FO model-checking problem

1990: Courcelle – MSO model checking on classes of bounded treewidth

1999: Flum and Grohe – start systematic study via parameterised complexity



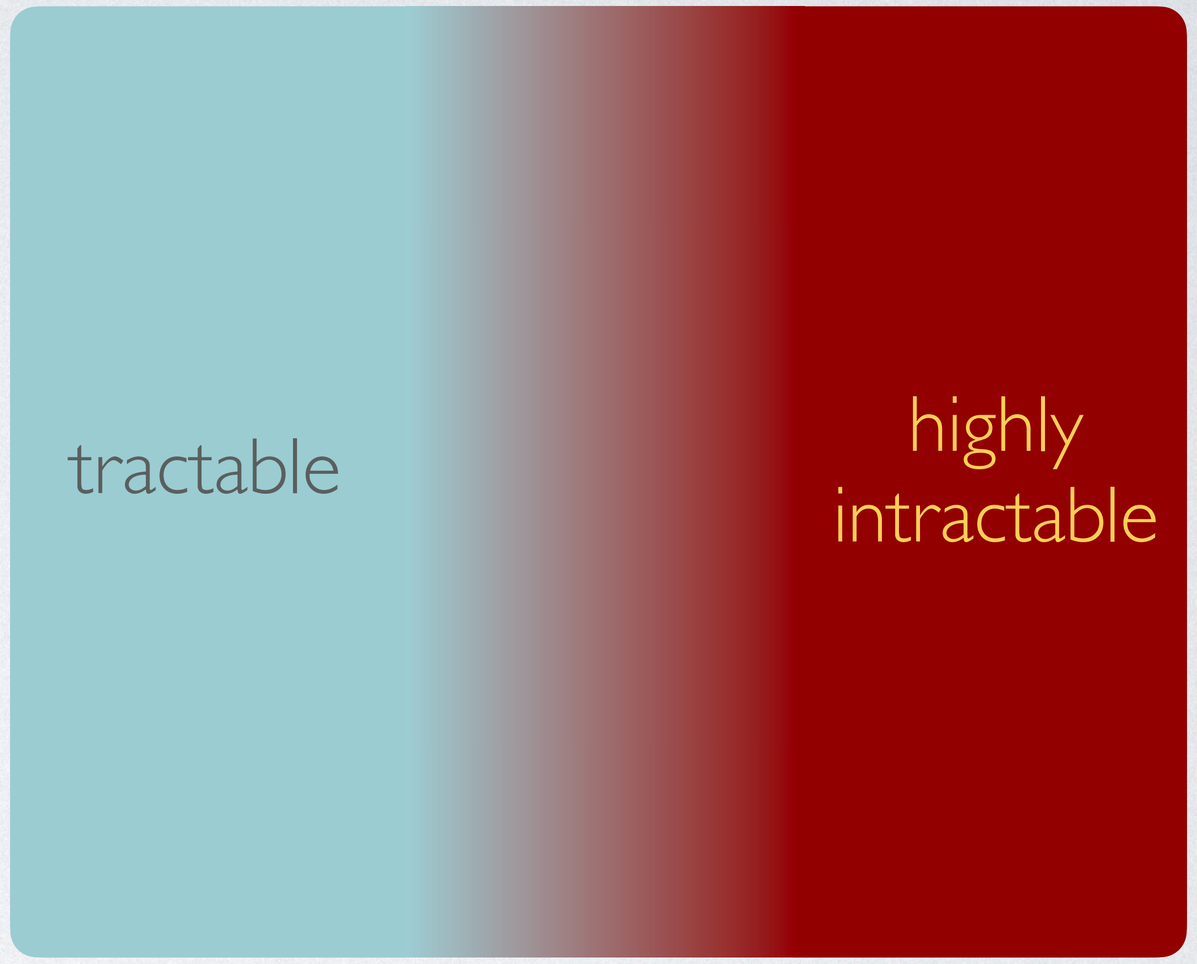
HISTORY

parameterised complexity of the FO model-checking problem

1990: Courcelle – MSO model checking on classes of bounded treewidth

1999: Flum and Grohe – start systematic study via parameterised complexity

1999–2007: initial progress via graph minor theory



HISTORY

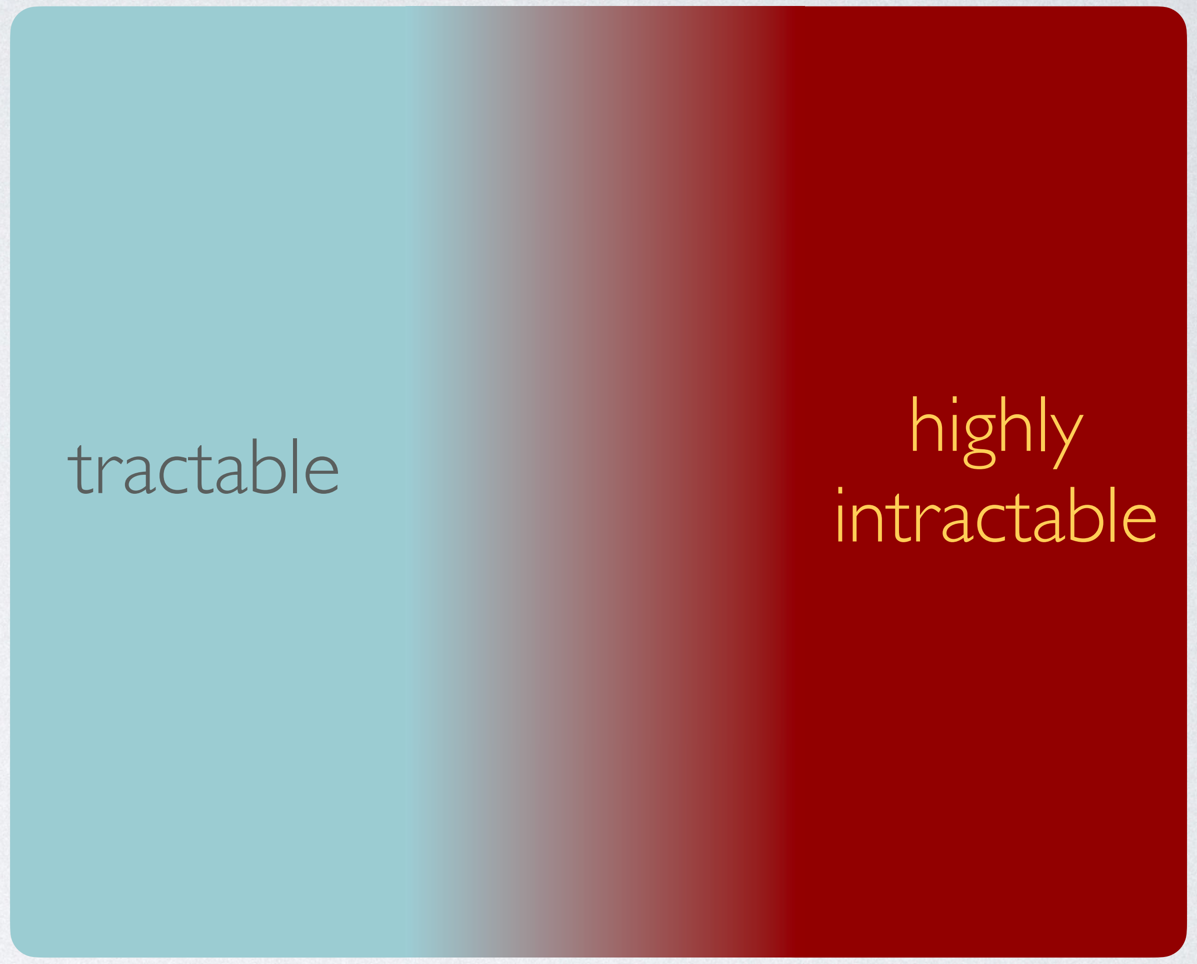
parameterised complexity of the FO model-checking problem

1990: Courcelle – MSO model checking on classes of bounded treewidth

1999: Flum and Grohe – start systematic study via parameterised complexity

1999–2007: initial progress via graph minor theory

STOC 2006–10: Nešetřil and Ossona de Mendez – ***sparsity theory***



HISTORY

parameterised complexity of the FO model-checking problem

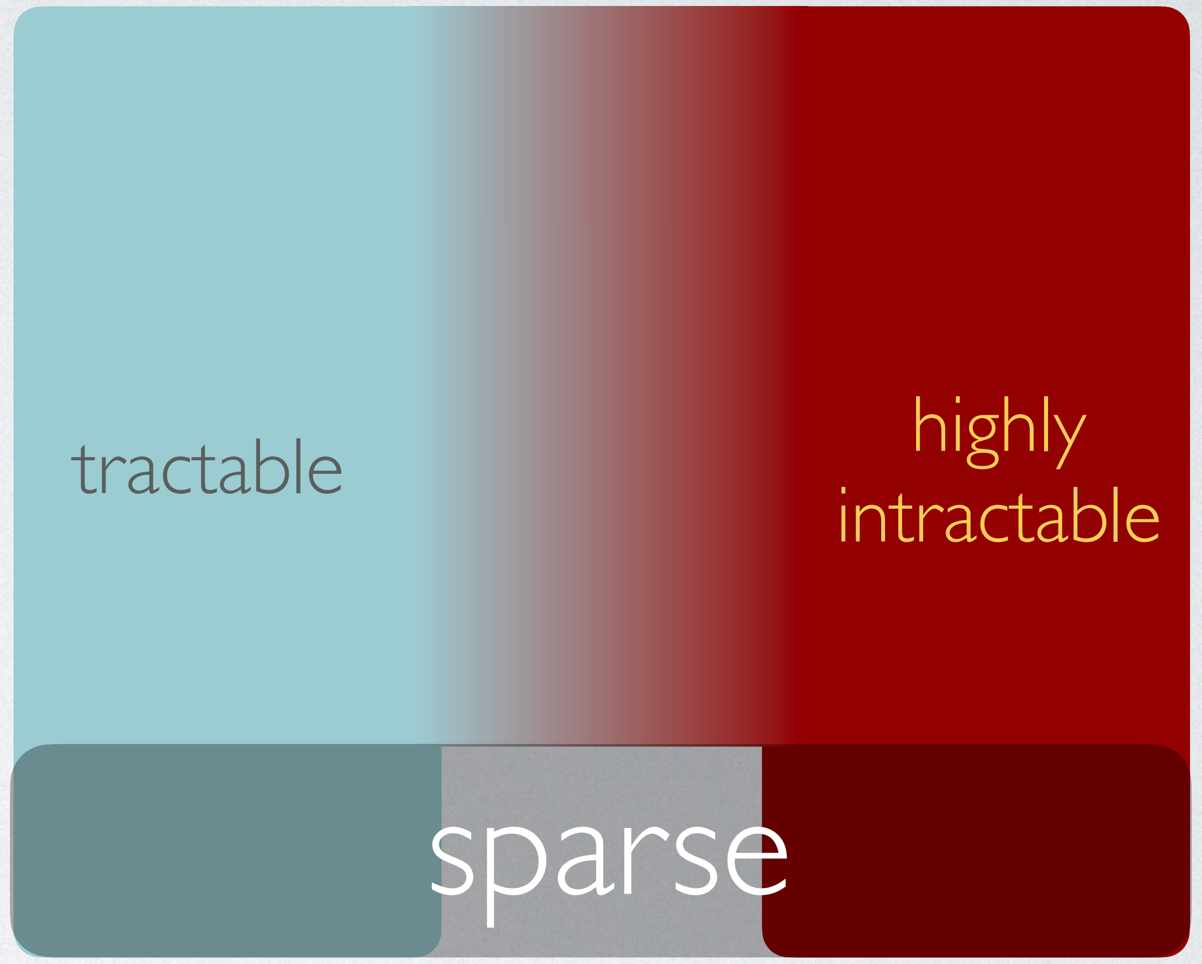
1990: Courcelle – MSO model checking on classes of bounded treewidth

1999: Flum and Grohe – start systematic study via parameterised complexity

1999–2007: initial progress via graph minor theory

STOC 2006–10: Nešetřil and Ossona de Mendez – ***sparsity theory***

STOC 2014: Grohe, Kreutzer, Siebertz – **sparse case**



HISTORY

parameterised complexity of the FO model-checking problem

1990: Courcelle – MSO model checking on classes of bounded treewidth

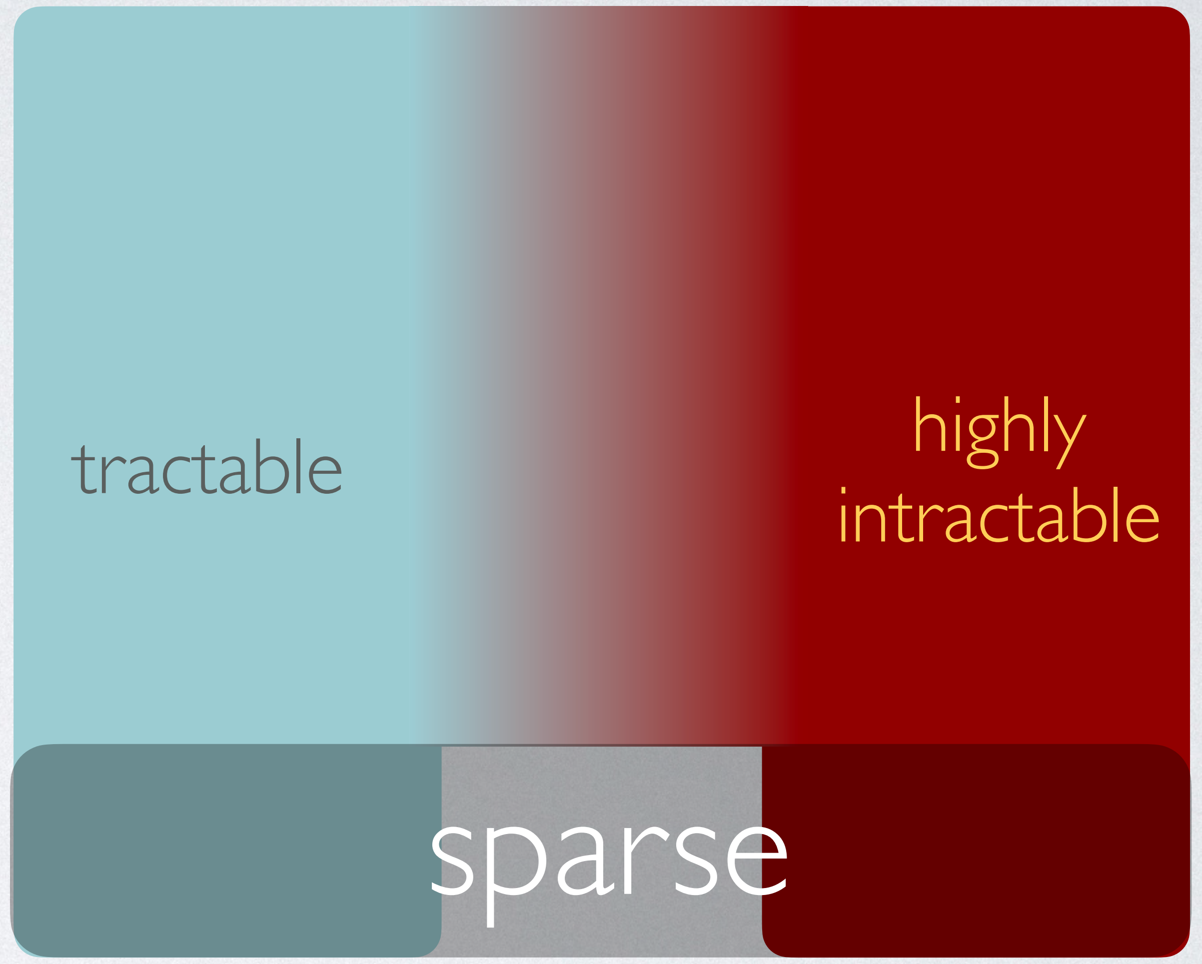
1999: Flum and Grohe – start systematic study via parameterised complexity

1999–2007: initial progress via graph minor theory

STOC 2006–10: Nešetřil and Ossona de Mendez – ***sparsity theory***

STOC 2014: Grohe, Kreutzer, Siebertz – **sparse case**

FOCS 2020: Bonnet, Kim, Thomassé, Watrigant – ***twin-width***



HISTORY

parameterised complexity of the FO model-checking problem

1990: Courcelle – MSO model checking on classes of bounded treewidth

1999: Flum and Grohe – start systematic study via parameterised complexity

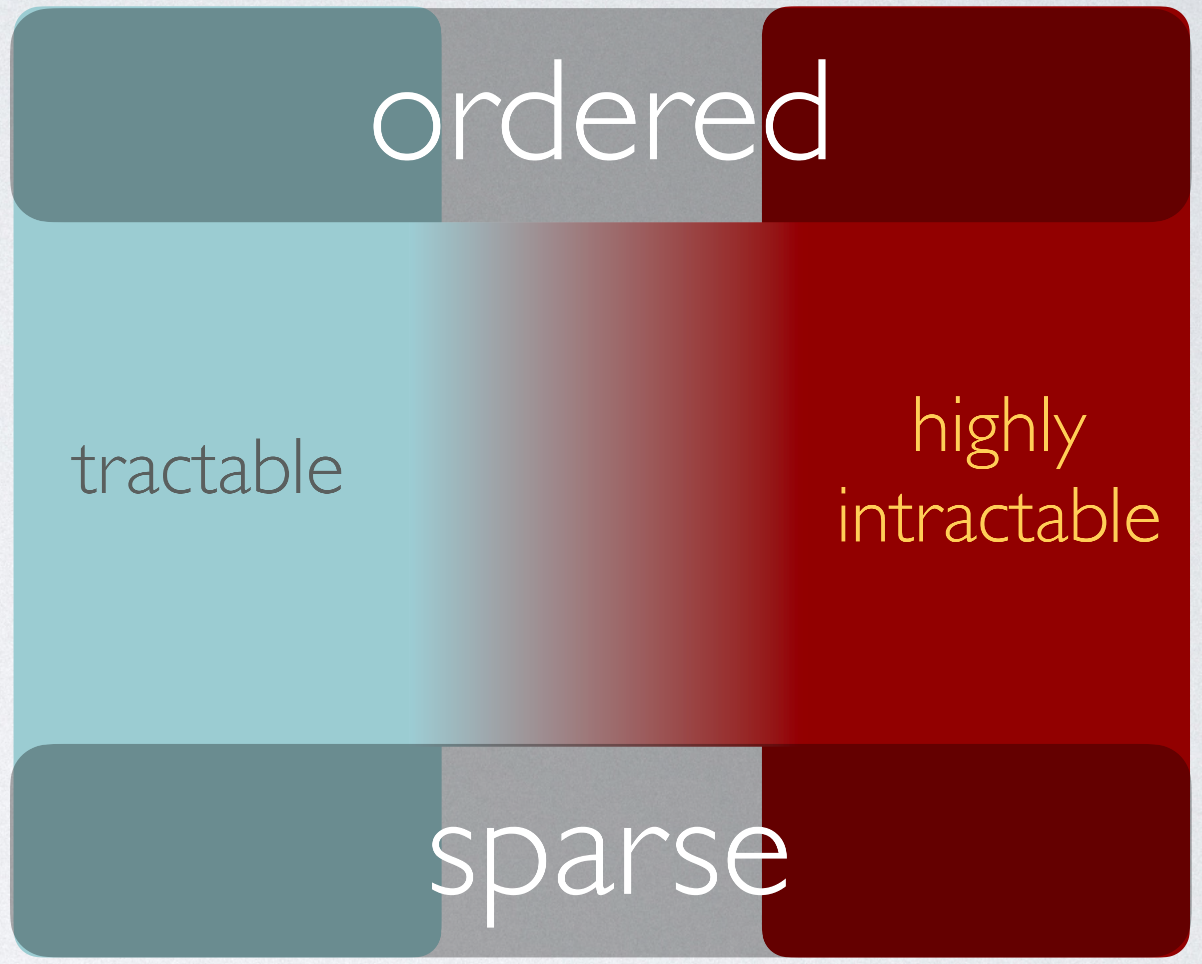
1999–2007: initial progress via graph minor theory

STOC 2006–10: Nešetřil and Ossona de Mendez – ***sparsity theory***

STOC 2014: Grohe, Kreutzer, Siebertz – **sparse case**

FOCS 2020: Bonnet, Kim, Thomassé, Watrigant – ***twin-width***

STOC 2022: Bonnet, O. de Mendez, Thomassé, Simon, **T**. **ordered case**



HISTORY

parameterised complexity of the FO model-checking problem

1990: Courcelle – MSO model checking on classes of bounded treewidth

1999: Flum and Grohe – start systematic study via parameterised complexity

1999–2007: initial progress via graph minor theory

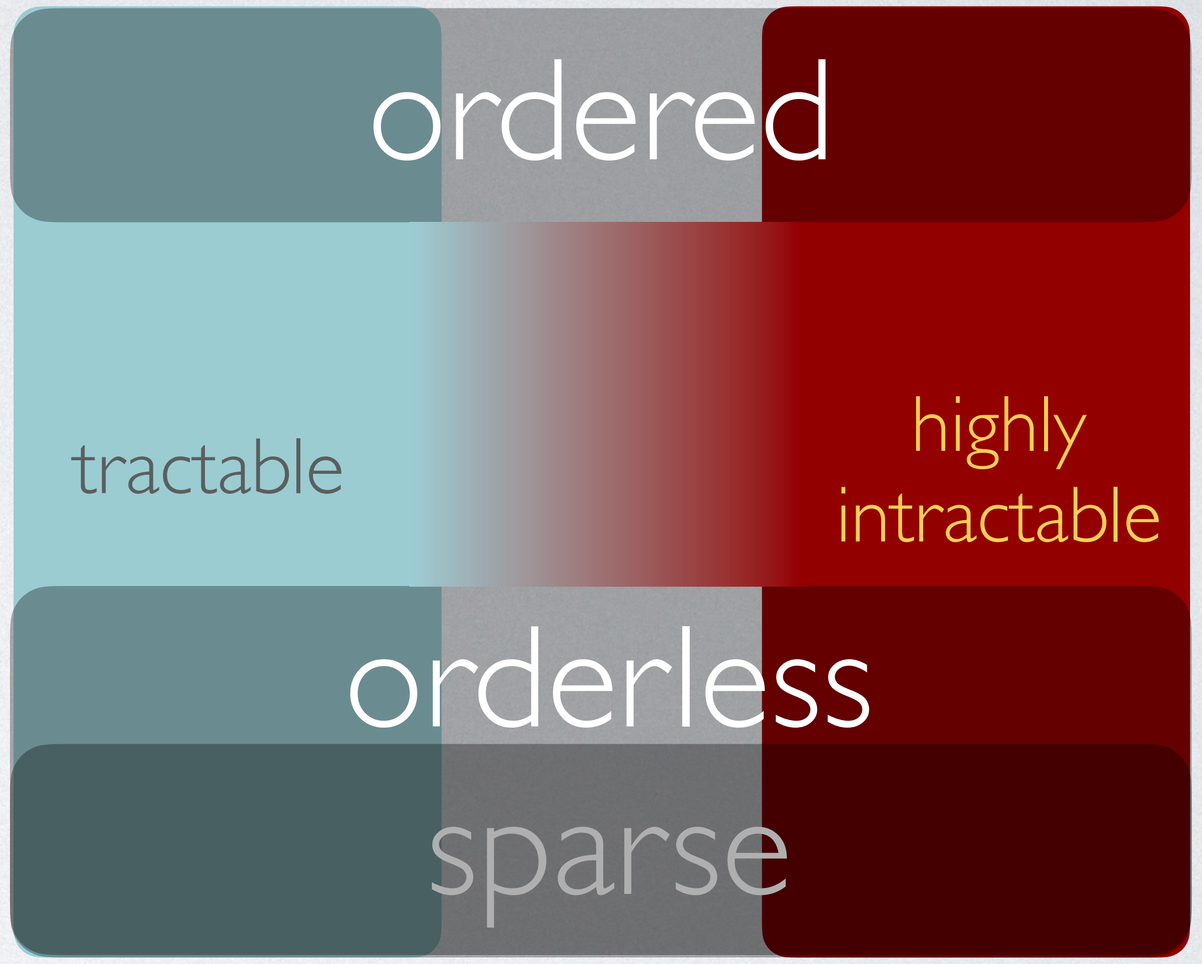
STOC 2006–10: Nešetřil and Ossona de Mendez – ***sparsity theory***

STOC 2014: Grohe, Kreutzer, Siebertz – **sparse case**

FOCS 2020: Bonnet, Kim, Thomassé, Watrigant – ***twin-width***

STOC 2022: Bonnet, O. de Mendez, Thomassé, Simon, **T**. **ordered case**

2024+: Dreier, Eleftheriadis, Mählmann, McCarty, Pilipczuk, **T**. **orderless case**



HISTORY

parameterised complexity of the FO model-checking problem

1990: Courcelle – MSO model checking on classes of bounded treewidth

1999: Flum and Grohe – start systematic study via parameterised complexity

1999–2007: initial progress via graph minor theory

STOC 2006–10: Nešetřil and Ossona de Mendez – ***sparsity theory***

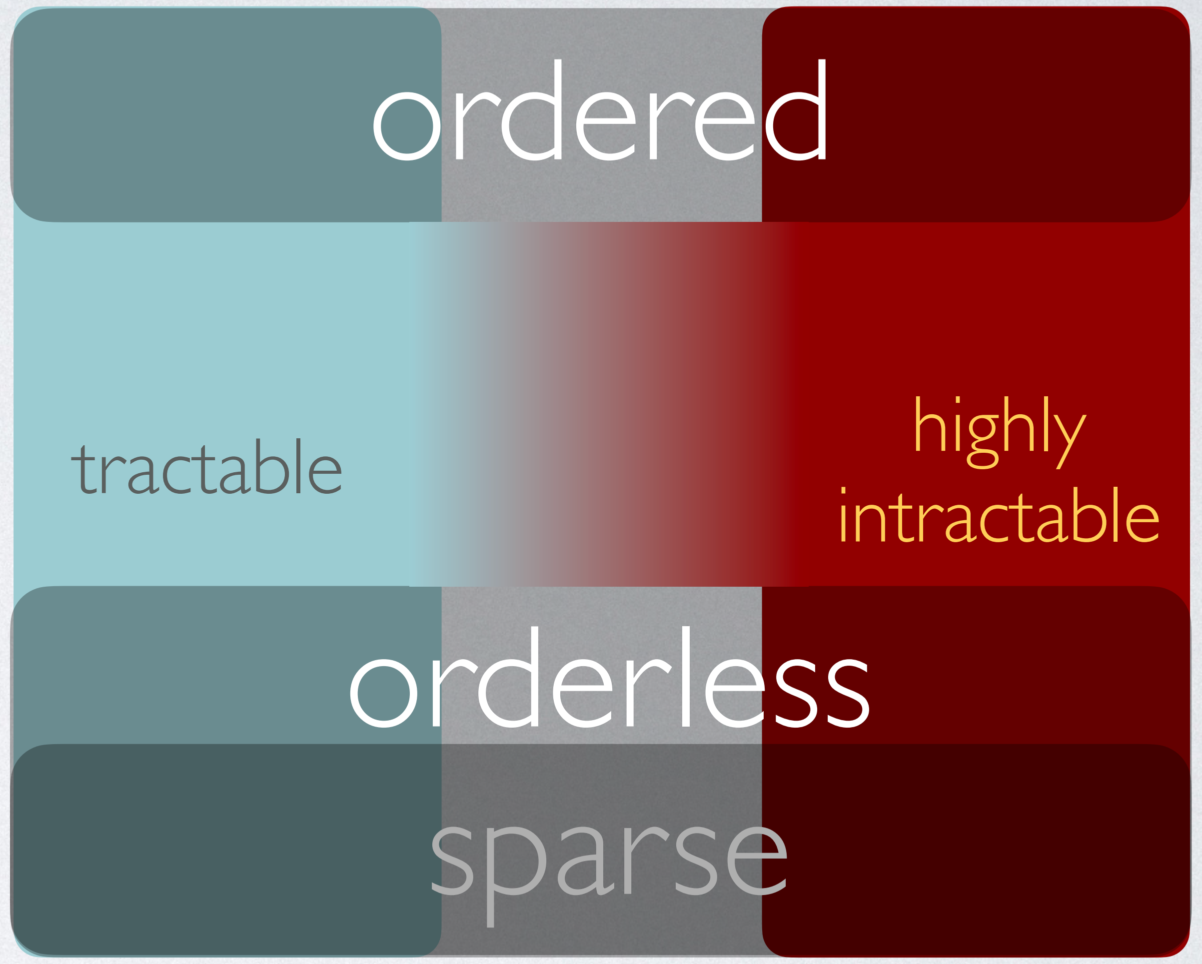
STOC 2014: Grohe, Kreutzer, Siebertz – **sparse case**

FOCS 2020: Bonnet, Kim, Thomassé, Watrigant – ***twin-width***

STOC 2022: Bonnet, O. de Mendez, Thomassé, Simon, **T**. **ordered case**

2024+: Dreier, Eleftheriadis, Mählmann, McCarty, Pilipczuk, **T**. **orderless case**

STOC 2024: Dreier, Mählmann, **T**. initial progress in general case



OUTLINE

1. The model checking problem
2. **Sparsity: monotone case**
3. Twin-width: ordered case
4. Monadic dependence
5. Flip-breakability
6. Stability: orderless case

SPARSITY

Nešetřil and Ossona de Mendez

SPARSITY

Nešetřil and Ossona de Mendez

Definition. *C* is *nowhere dense* if

SPARSITY

Nešetřil and Ossona de Mendez

Definition. C is *nowhere dense* if

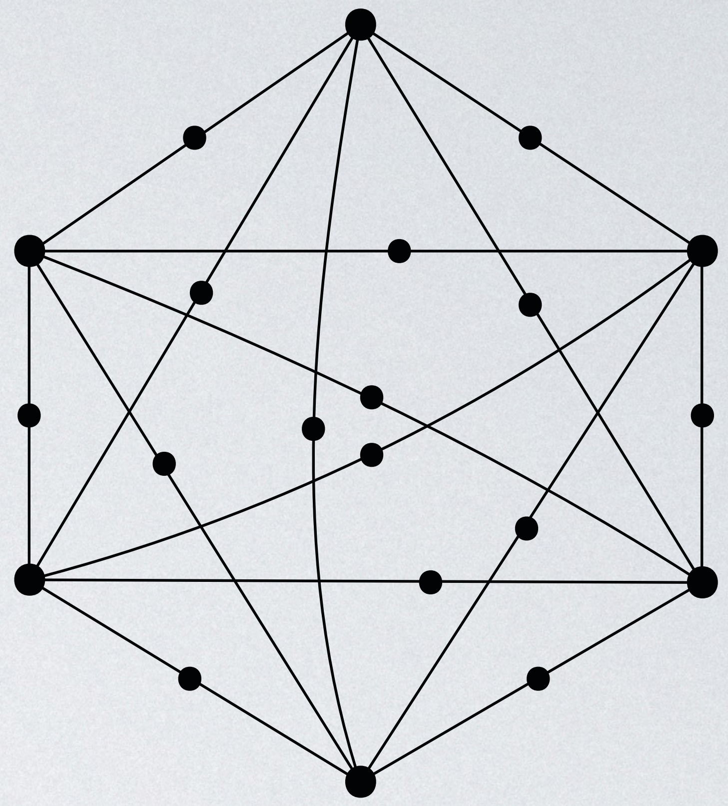
$\forall r \geq 0\ \exists k_r.$ C avoids the r-subdivided k_r-clique as a subgraph

SPARSITY

Nešetřil and Ossona de Mendez

Definition. *C* is *nowhere dense* if

$\forall r \geq 0\ \exists k_r.$ C avoids the r-subdivided k_r-clique as a subgraph



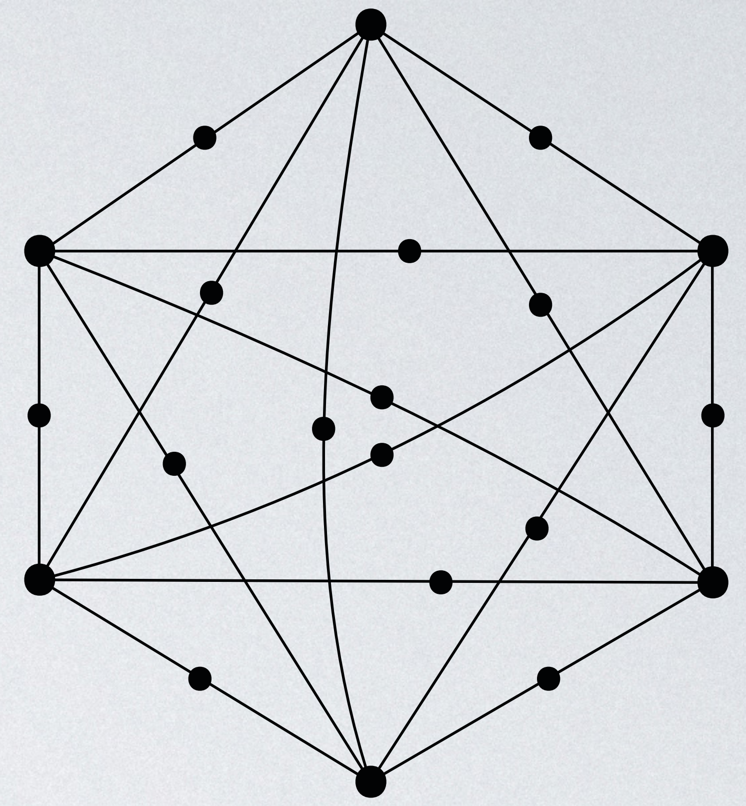
1-subdivided
6-clique

SPARSITY

Nešetřil and Ossona de Mendez

Definition. C is *nowhere dense* if

$\forall r \geq 0\ \exists k_r.$ C avoids the r-subdivided k_r-clique as a subgraph



1-subdivided 6-clique

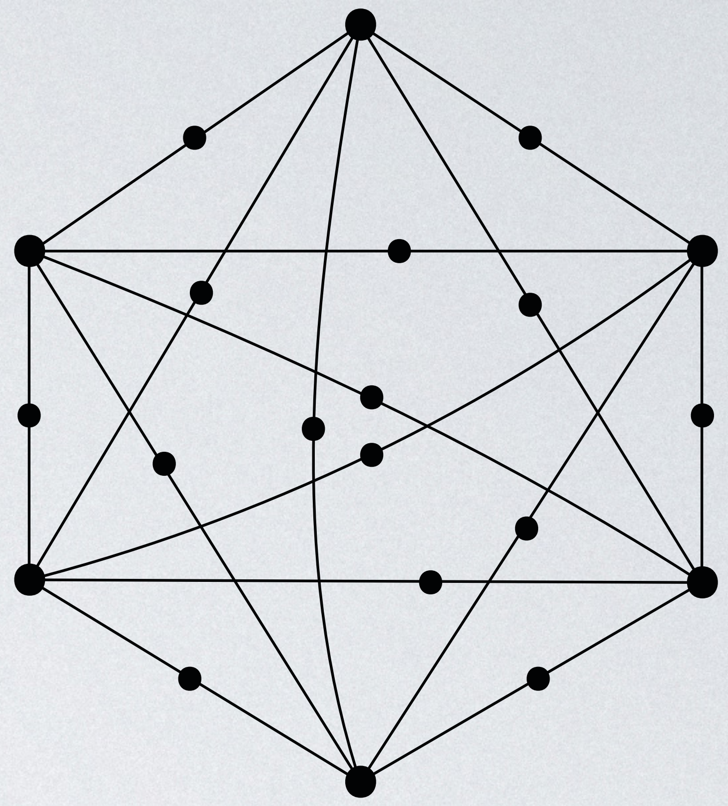
Example. Class of planar graphs: $\forall r \geq 0, k_r := 5.$

SPARSITY

Nešetřil and Ossona de Mendez

Definition. C is *nowhere dense* if

$\forall r \geq 0\ \exists k_r.$ C avoids the r-subdivided k_r-clique as a subgraph



1-subdivided
6-clique

Example. Class of planar graphs: $\forall r \geq 0, k_r := 5.$

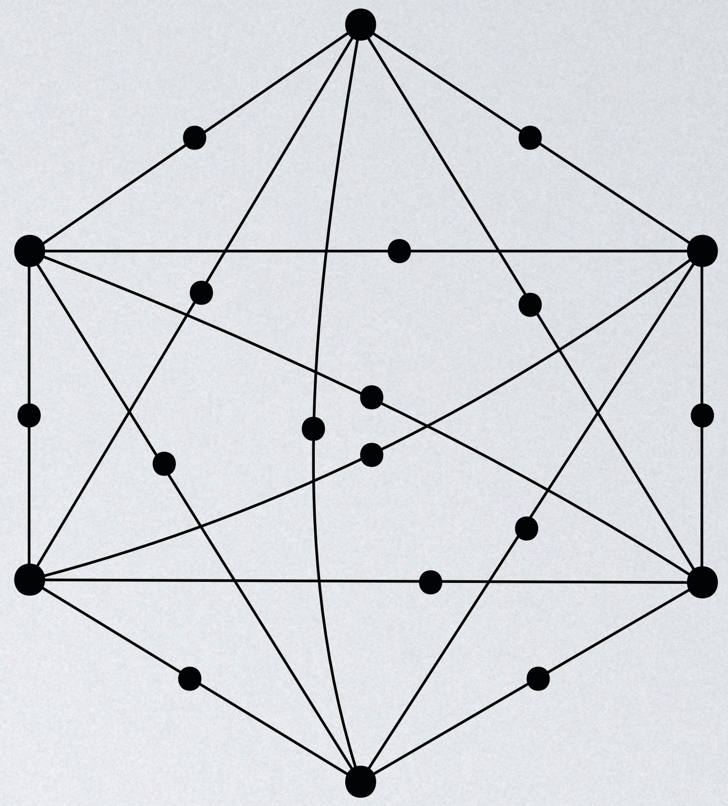
Example. Class of graphs of max. degree ≤ 3: $\forall r \geq 0, k_r := 5.$

SPARSITY

Nešetřil and Ossona de Mendez

Definition. C is *nowhere dense* if

$\forall r \geq 0\ \exists k_r.$ C avoids the r-subdivided k_r-clique as a subgraph



1-subdivided
6-clique

Example. Class of planar graphs: $\forall r \geq 0, k_r := 5.$

Example. Class of graphs of max. degree ≤ 3: $\forall r \geq 0, k_r := 5.$

Fact. If C is nowhere dense, $\varepsilon > 0$, then every $G \in C$ has $O_{\varepsilon,C}(\ |V(G)|^{1+\varepsilon})$ edges.

SPARSITY

bounded treewidth

↓

excluding a minor

↓

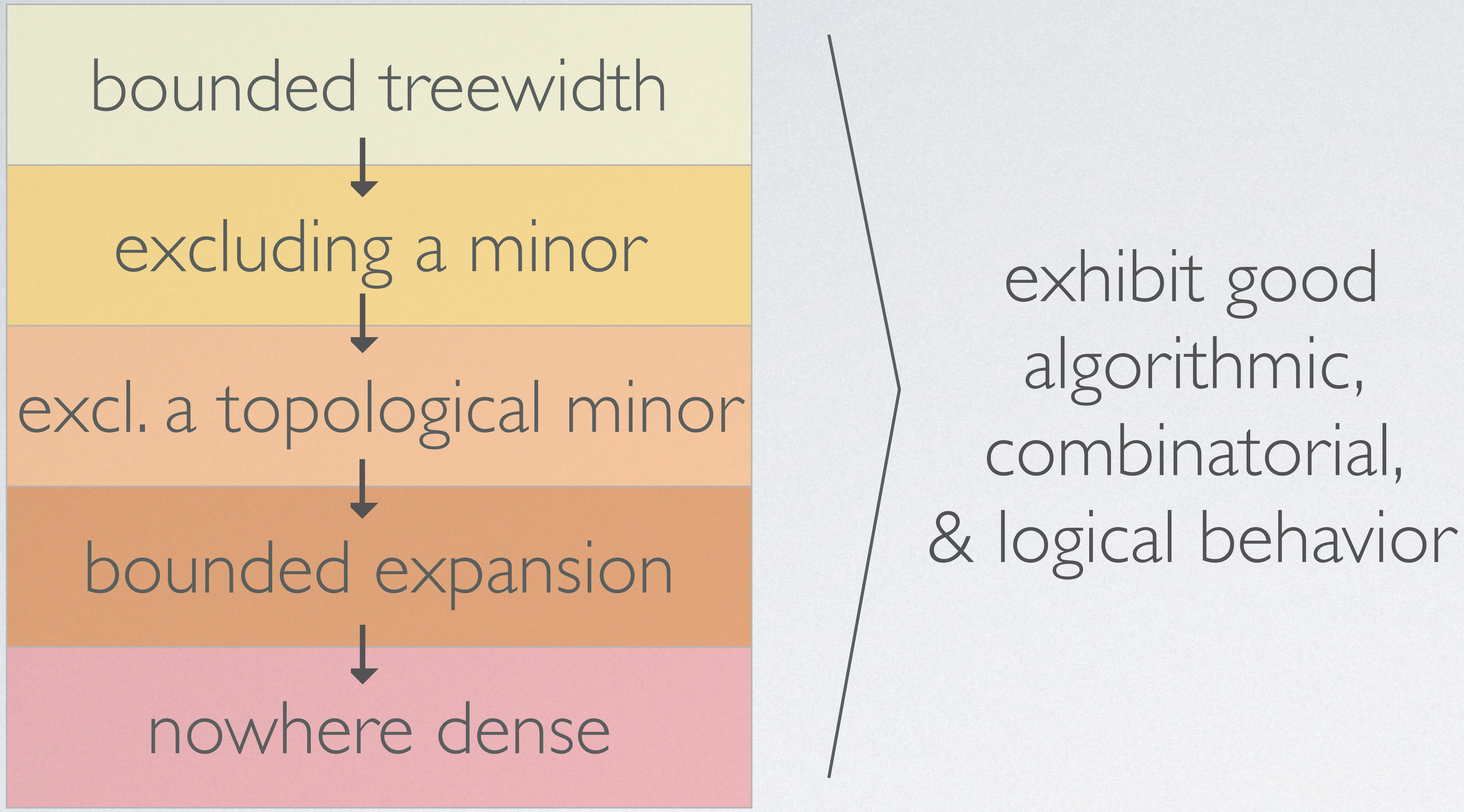
excl. a topological minor

↓

bounded expansion

↓

nowhere dense


bounded treewidth
excluding a minor
excl. a topological minor
bounded expansion
nowhere dense
exhibit good
algorithmic,
combinatorial,
& logical behavior

bounded treewidth

↓

excluding a minor

↓

excl. a topological minor

↓

bounded expansion

↓

nowhere dense

bounded treewidth
↓
excluding a minor
↓
excl. a topological minor
↓
bounded expansion
↓
nowhere dense

Theorem (Courcelle 1990)
Model checking MSO logic is fpt
on every class of bounded treewidth

bounded treewidth
↓
excluding a minor
↓
excl. a topological minor
↓
bounded expansion
↓
nowhere dense

Theorem (Courcelle 1990)
Model checking MSO logic is fpt
on every class of bounded treewidth

Theorem (Grohe, Kreutzer, Siebertz, 2014)
Model checking FO logic is fpt
on every nowhere dense class.

MODEL CHECKING ON SPARSE CLASSES

Theorem (Grohe, Kreutzer, Siebertz, 2014)
Model checking FO logic is fpt on every nowhere dense class C:

MODEL CHECKING ON SPARSE CLASSES

Theorem (Grohe, Kreutzer, Siebertz, 2014)
Model checking FO logic is fpt on every nowhere dense class C:

Given $\varepsilon>0$, $\phi\in$FO, $G\in C$, $G \models \phi$ can be tested in time $O_{\phi,\varepsilon}(\,|G|^{1+\varepsilon}\,)$.

MODEL CHECKING ON SPARSE CLASSES

Theorem (Grohe, Kreutzer, Siebertz, 2014)
Model checking FO logic is fpt on every nowhere dense class C:

Given $\varepsilon>0$, $\phi\in$FO, $G\in C$, $G \models \phi$ can be tested in time $O_{\phi,\varepsilon}(\ |G|^{1+\varepsilon}\ )$.

Ingredients:

MODEL CHECKING ON SPARSE CLASSES

Theorem (Grohe, Kreutzer, Siebertz, 2014)
Model checking FO logic is fpt on every nowhere dense class C:

Given $\varepsilon>0$, $\phi\in$FO, $G\in C$, $G \models \phi$ can be tested in time $O_{\phi,\varepsilon}(\ |G|^{1+\varepsilon}\ )$.

Ingredients:

1. existence of a treelike decomposition

MODEL CHECKING ON SPARSE CLASSES

Theorem (Grohe, Kreutzer, Siebertz, 2014)
Model checking FO logic is fpt on every nowhere dense class C:

Given $\varepsilon>0$, $\phi\in$FO, $G\in C$, $G \models \phi$ can be tested in time $O_{\phi,\varepsilon}(\,|G|^{1+\varepsilon}\,)$.

Ingredients:

1. existence of a treelike decomposition
2. efficient computation of a decomposition

MODEL CHECKING ON SPARSE CLASSES

Theorem (Grohe, Kreutzer, Siebertz, 2014)
Model checking FO logic is fpt on every nowhere dense class C:

Given $\varepsilon>0$, $\phi\in$FO, $G\in C$, $G \models \phi$ can be tested in time $O_{\phi,\varepsilon}(\,|G|^{1+\varepsilon}\,)$.

Ingredients:

1. existence of a treelike decomposition
2. efficient computation of a decomposition
3. dynamic algorithm computing partial solutions to formulas – uses locality of FO

MONOTONE CLASSES

=subgraph closed classes

Corollary. Let C be a monotone graph class. Then:

C is nowhere dense $^*\Leftrightarrow$ FO-model checking is fpt on C.

*assuming AW[$*$]≠FPT

MONOTONE CLASSES

=subgraph closed classes

Corollary. Let C be a monotone graph class. Then:

C is nowhere dense $^*\Leftrightarrow$ FO-model checking is fpt on C.

"$\Leftarrow$":

*assuming AW[$*$]$\neq$FPT

MONOTONE CLASSES

=subgraph closed classes

Corollary. Let C be a monotone graph class. Then:

C is nowhere dense $^*\Leftrightarrow$ FO-model checking is fpt on C.

"$\Leftarrow$":

C is monotone and not nowhere dense $\Rightarrow$

*assuming AW[$*$]$\neq$FPT

MONOTONE CLASSES

=subgraph closed classes

Corollary. Let C be a monotone graph class. Then:

C is nowhere dense $^*\Leftrightarrow$ FO-model checking is fpt on C.

"$\Leftarrow$":

C is monotone and not nowhere dense $\Rightarrow$

$\exists r \geq 0 \forall n$ C contains the r-subdivided clique K_n $\Rightarrow$

*assuming AW[$*$]$\neq$FPT

MONOTONE CLASSES

=subgraph closed classes

Corollary. Let C be a monotone graph class. Then:

C is nowhere dense $^*\Leftrightarrow$ FO-model checking is fpt on C.

"$\Leftarrow$":

C is monotone and not nowhere dense $\Rightarrow$

$\exists r\geq 0 \forall n$ C contains the r-subdivided clique $K_n \Rightarrow$

$\exists r\geq 0 \forall G$ C contains the r-subdivision of $G \Rightarrow$ *hardness*

*assuming AW[$*$]$\neq$FPT

BEYOND SPARSITY

BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

- retain many good properties of treewidth
- applicable to dense graphs

BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

- retain many good properties of treewidth
- applicable to dense graphs

Definition A graph G has cliquewidth $\leq k$ $\Leftrightarrow$ G can be created using operations:

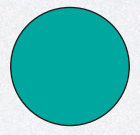
BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

- retain many good properties of treewidth
- applicable to dense graphs

Definition A graph G has cliquewidth $\leq k \Leftrightarrow$ G can be created using operations:

- Create new vertex with color $i \in [k]$



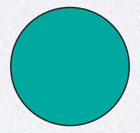
BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

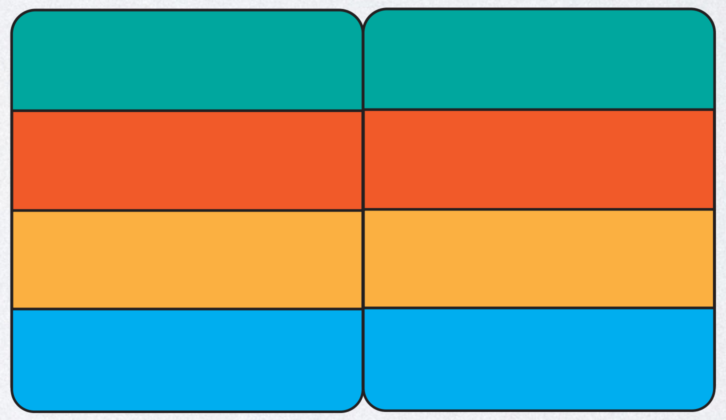
- retain many good properties of treewidth
- applicable to dense graphs

Definition A graph G has cliquewidth $\leq k$ $\Leftrightarrow$ G can be created using operations:

- Create new vertex with color $i \in [k]$



- Take disjoint union of two colored graphs



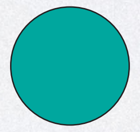
BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

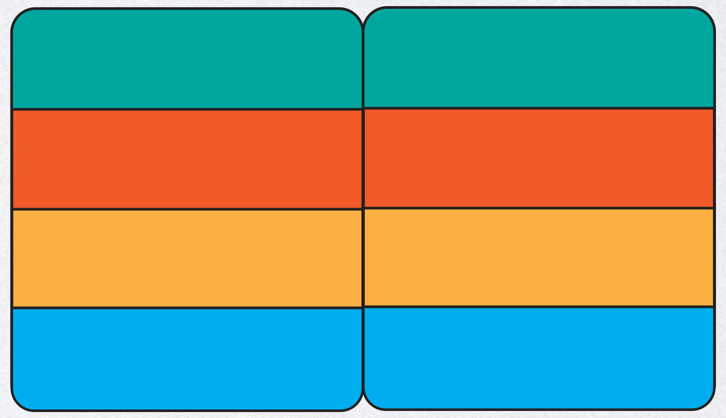
- retain many good properties of treewidth
- applicable to dense graphs

Definition A graph G has cliquewidth $\leq k \Leftrightarrow$ G can be created using operations:

- Create new vertex with color $i \in [k]$



- Take disjoint union of two colored graphs



- Join by an edge every vertex colored i to every vertex colored j

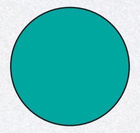
BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

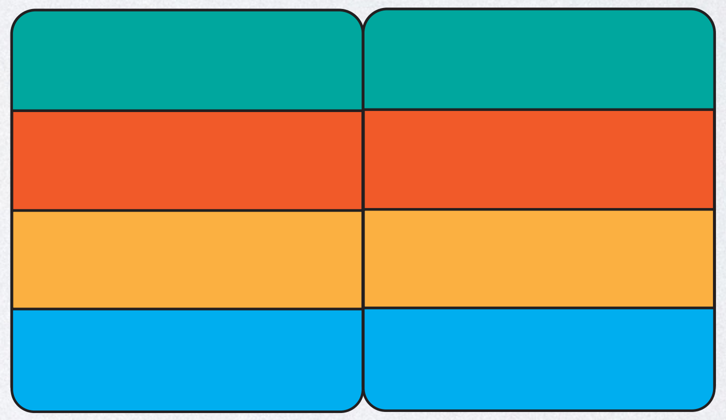
- retain many good properties of treewidth
- applicable to dense graphs

Definition A graph G has cliquewidth $\leq k \Leftrightarrow$ G can be created using operations:

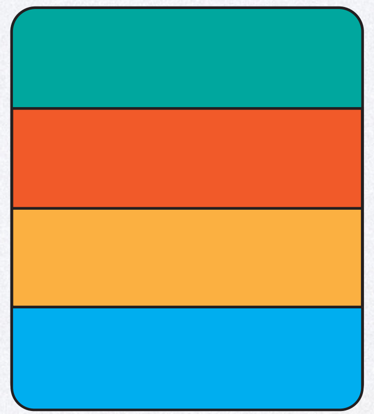
- Create new vertex with color $i \in [k]$



- Take disjoint union of two colored graphs



- Join by an edge every vertex colored i to every vertex colored j



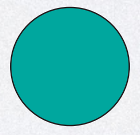
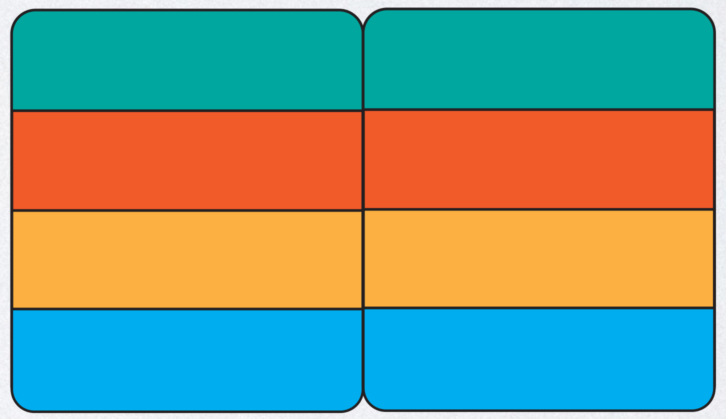
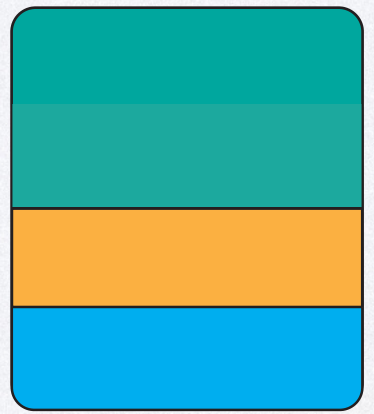
- Recolor i to color j

BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

- retain many good properties of treewidth
- applicable to dense graphs

Definition A graph G has cliquewidth $\leq k \Leftrightarrow$ G can be created using operations:

- Create new vertex with color $i \in [k]$ 
- Take disjoint union of two colored graphs 
- Join by an edge every vertex colored i to every vertex colored j
- Recolor i to color j 

BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

- retain many good properties of treewidth
- applicable to dense graphs

Theorem [Courcelle-Rotics-Makowsky + Oum-Seymour]
Model checking MSO is fpt on classes of bounded cliquewidth

BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

- retain many good properties of treewidth
- applicable to dense graphs

Theorem [Courcelle-Rotics-Makowsky + Oum-Seymour]
Model checking MSO is fpt on classes of bounded cliquewidth

Ingredients:

BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

- retain many good properties of treewidth
- applicable to dense graphs

Theorem [Courcelle-Rotics-Makowsky + Oum-Seymour]
Model checking MSO is fpt on classes of bounded cliquewidth

Ingredients:

1. existence of treelike decompositions – by definition

BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

- retain many good properties of treewidth
- applicable to dense graphs

Theorem [Courcelle-Rotics-Makowsky + Oum-Seymour]
Model checking MSO is fpt on classes of bounded cliquewidth

Ingredients:

1. existence of treelike decompositions – by definition
2. efficient computation of decomposition [Oum, Seymour 2006]

BEYOND SPARSITY

Prototype: treewidth $\mapsto$ cliquewith/rankwidth

- retain many good properties of treewidth
- applicable to dense graphs

Theorem [Courcelle-Rotics-Makowsky + Oum-Seymour]
Model checking MSO is fpt on classes of bounded cliquewidth

Ingredients:

1. existence of treelike decompositions – by definition
2. efficient computation of decomposition [Oum, Seymour 2006]
3. dynamic algorithm computing partial solutions to formulas [Courcelle, Rotics, Makowsky 2000]

BEYOND SPARSITY

Project: extend from monotone classes to hereditary classes

monotone		*hereditary*
bounded treewidth	→	bounded cliquewidth
↓		↓
excluding a minor	→	?
↓		↓
excl. a top. minor	→	?
↓		↓
bounded expansion	→	?
↓		↓
nowhere dense	→	?

OUTLINE

1. The model checking problem
2. Sparsity: monotone case
3. **Twin-width: ordered case**
4. Monadic dependence
5. Flip-breakability
6. Stability: orderless case

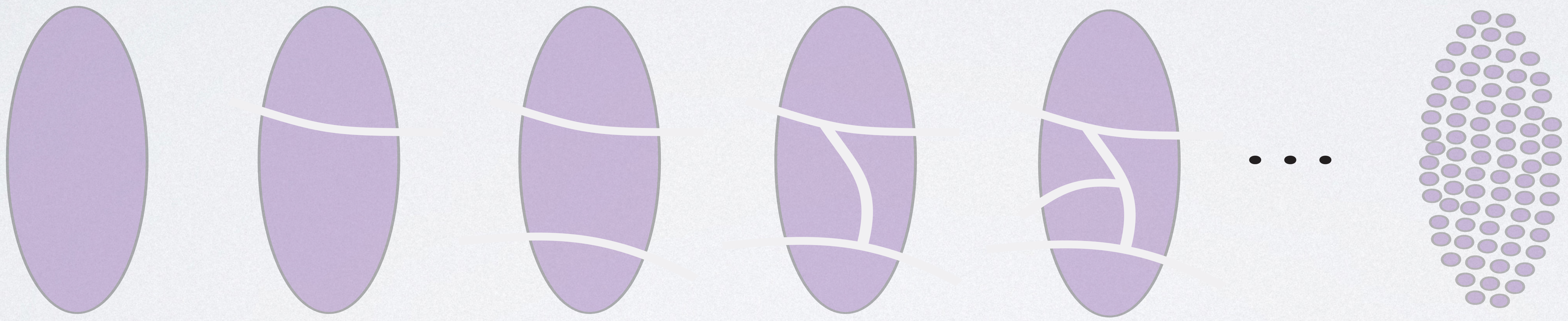
TWIN-WIDTH

Bonnet, Thomassé, and coauthors

TWIN-WIDTH

Bonnet, Thomassé, and coauthors

Definition. G has *twin-width* $\leq d$ if there is a refining sequence $P_1,\ldots,P_n$ of partitions of $V(G)$ – a *merge sequence* –

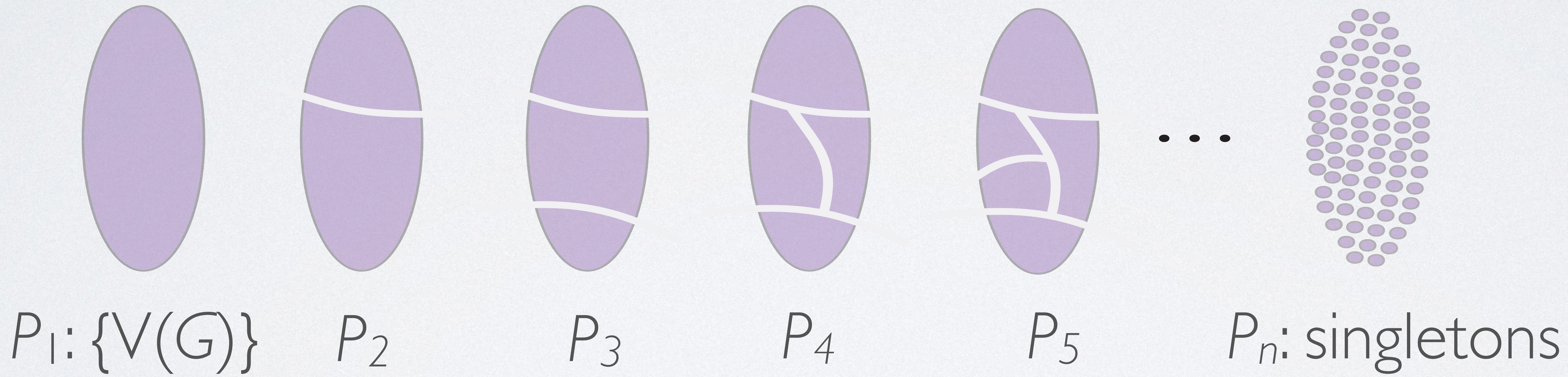


P_1: $\{V(G)\}$ P_2 P_3 P_4 P_5 … P_n: singletons

TWIN-WIDTH

Bonnet, Thomassé, and coauthors

Definition. G has *twin-width* $\leq d$ if there is a refining sequence $P_1, \ldots, P_n$ of partitions of $V(G)$ – a *merge sequence* –

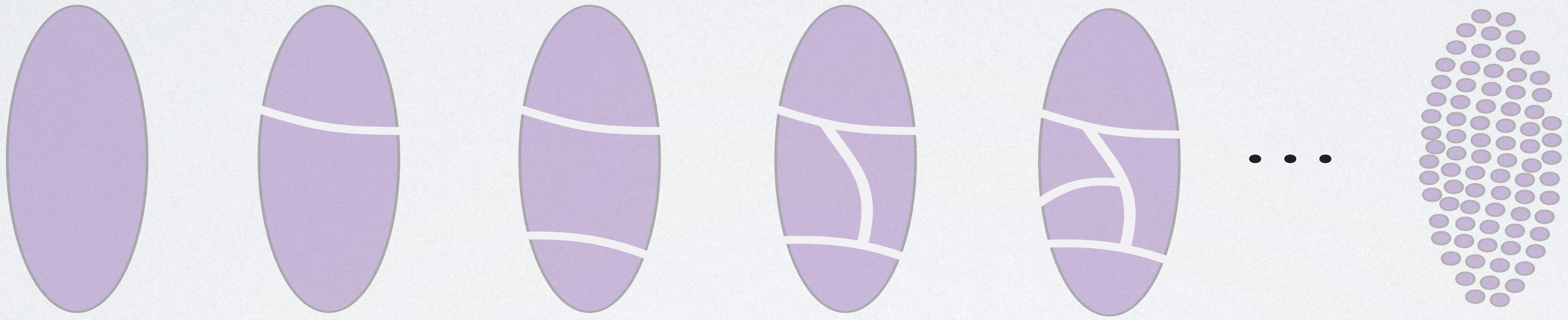


s.t. in each P_i, every part A is *impure* towards $\leq d$ parts:

TWIN-WIDTH

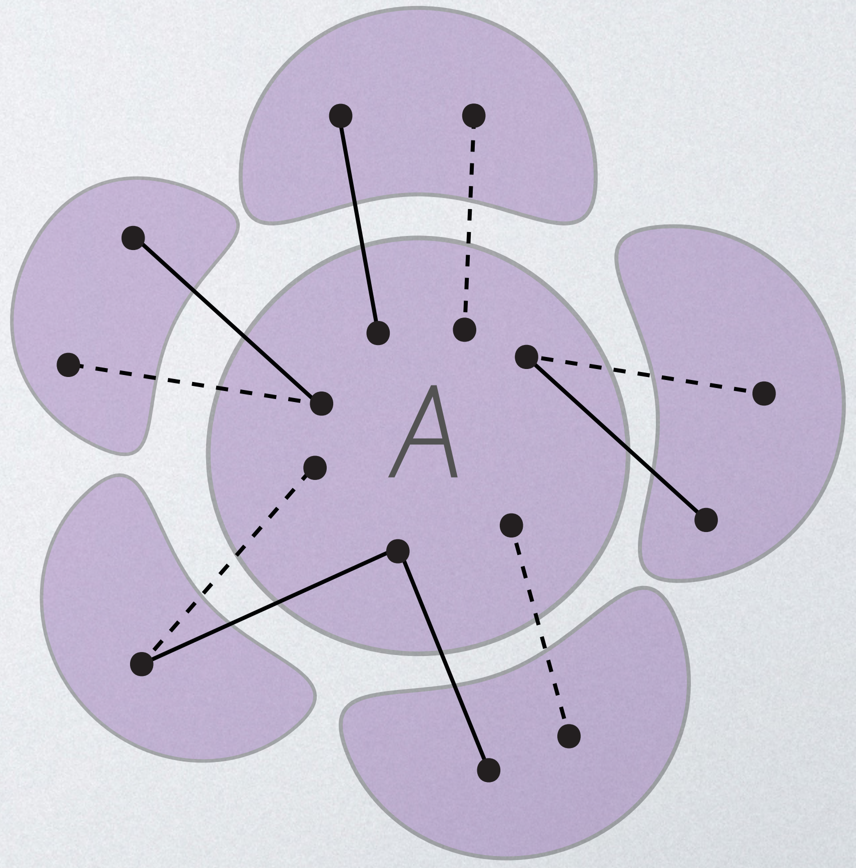
Bonnet, Thomassé, and coauthors

Definition. G has *twin-width* $\leq d$ if there is a refining sequence $P_1,\ldots,P_n$ of partitions of $V(G)$ – a *merge sequence* –



P_1: $\{V(G)\}$ P_2 P_3 P_4 P_5 P_n: singletons

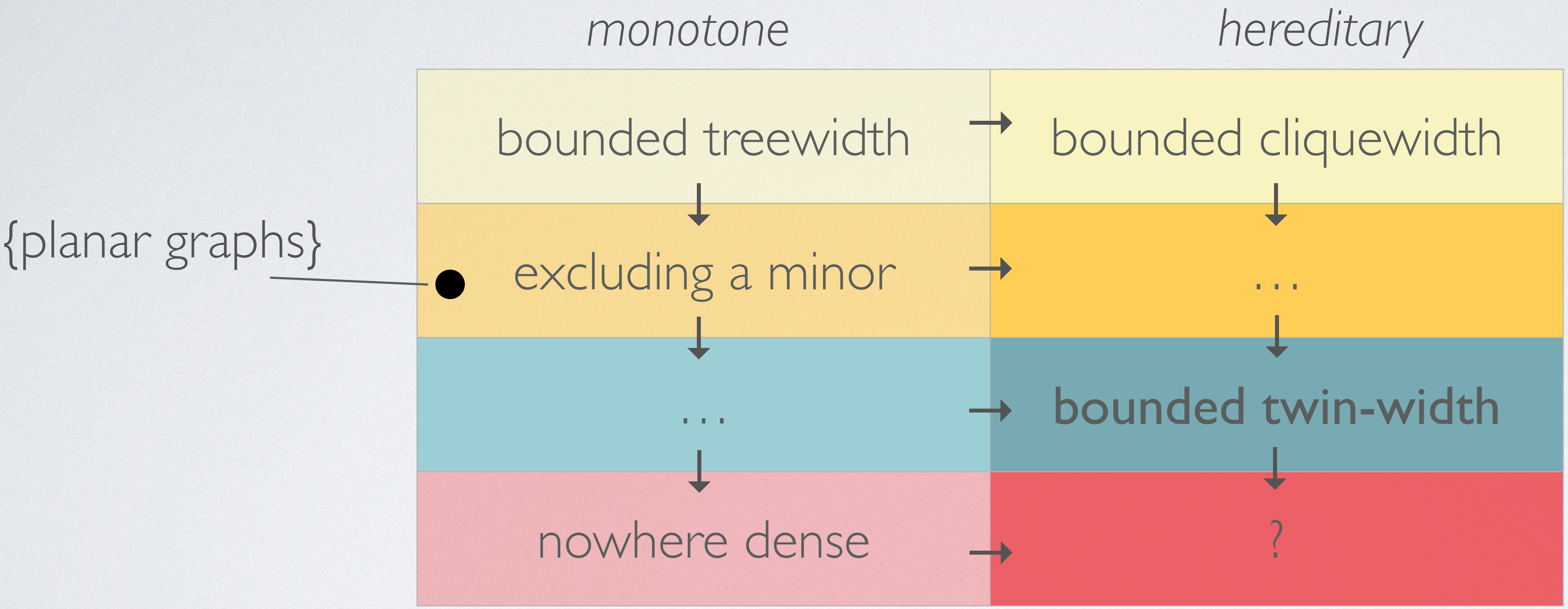
s.t. in each P_i, every part A is *impure* towards $\leq d$ parts:



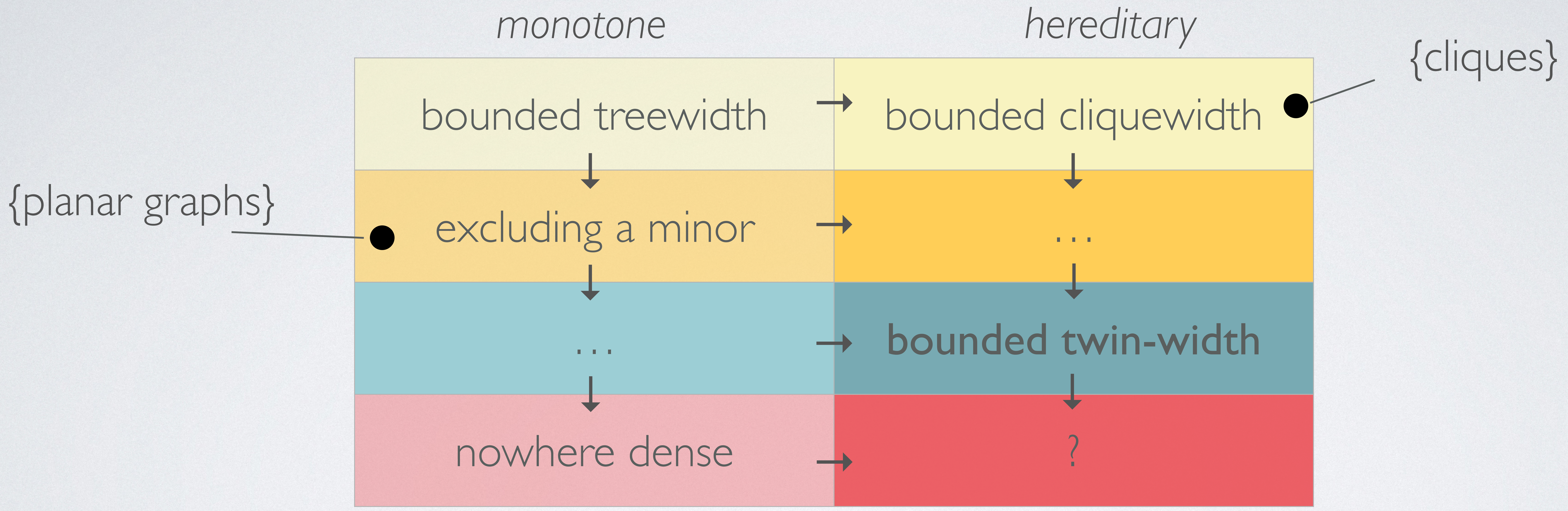
TWIN-WIDTH

monotone	*hereditary*
bounded treewidth	bounded cliquewidth
excluding a minor	…
…	**bounded twin-width**
nowhere dense	?

TWIN-WIDTH


monotone
hereditary
bounded treewidth
bounded cliquewidth
{planar graphs}
excluding a minor
. . .
. . .
bounded twin-width
nowhere dense
?

TWIN-WIDTH


monotone
hereditary
{cliques}
bounded treewidth
bounded cliquewidth
{planar graphs}
excluding a minor
. . .
. . .
bounded twin-width
nowhere dense
?

TWIN-WIDTH


monotone
hereditary
{cliques}
bounded treewidth
bounded cliquewidth
{planar graphs}
excluding a minor
. . .
{unit interval graphs}
. . .
bounded twin-width
nowhere dense
?

TWIN-WIDTH

monotone	*hereditary*
bounded treewidth	bounded cliquewidth
excluding a minor	…
…	**bounded twin-width**
nowhere dense	?

{cliques}

{planar graphs}

{unit interval graphs}

any proper hereditary class of permutation graphs

TWIN-WIDTH

monotone
hereditary
{cliques}
bounded treewidth
bounded cliquewidth
{planar graphs}
excluding a minor
. . .
{unit interval graphs}
. . .
bounded twin-width
nowhere dense
?
any proper hereditary class of permutation graphs
{graphs with max. degree 3}

MODEL CHECKING ON CLASSES OF BOUNDED TWIN-WIDTH

MODEL CHECKING ON CLASSES OF BOUNDED TWIN-WIDTH

Theorem (Bonnet, Kim, Thomassé, Watrigant, *Twin-width I*, 2020)
Given $\phi \in FO$, a graph G of twin-width $\leq d$ **with its merge sequence,**

MODEL CHECKING ON CLASSES OF BOUNDED TWIN-WIDTH

Theorem (Bonnet, Kim, Thomassé, Watrigant, *Twin-width I*, 2020)
Given $\phi \in$FO, a graph G of twin-width $\leq d$ **with its merge sequence,**

$G \models \phi$ can be tested in time $O_{\phi,d}(\ |G|\ )$

MODEL CHECKING ON CLASSES OF BOUNDED TWIN-WIDTH

Theorem (Bonnet, Kim, Thomassé, Watrigant, *Twin-width I*, 2020)
Given $\phi \in$FO, a graph G of twin-width $\leq d$ **with its merge sequence,**

$G \models \phi$ can be tested in time $O_{\phi,d}(\ |G|\ )$

Ingredients:

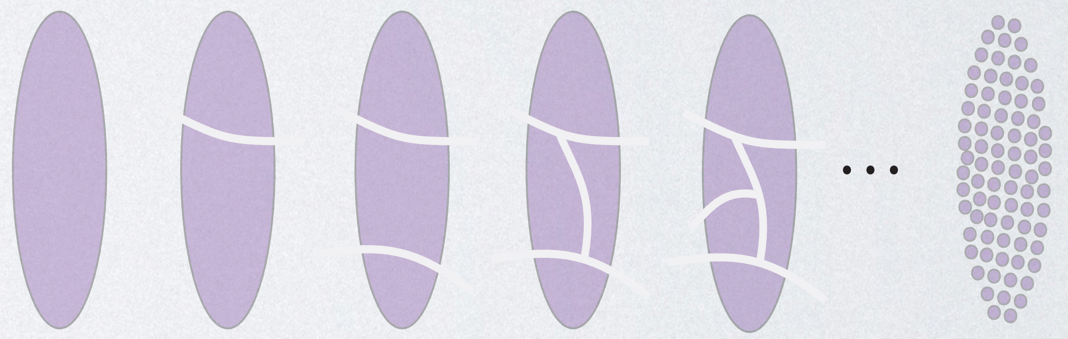
MODEL CHECKING ON CLASSES OF BOUNDED TWIN-WIDTH

Theorem (Bonnet, Kim, Thomassé, Watrigant, *Twin-width I*, 2020)
Given $\phi \in \mathrm{FO}$, a graph G of twin-width $\leq d$ **with its merge sequence,**

$G \models \phi$ can be tested in time $\quad O_{\phi,d}(\,|G|\,)$

Ingredients:

1. existence of a treelike decomposition – contraction sequence



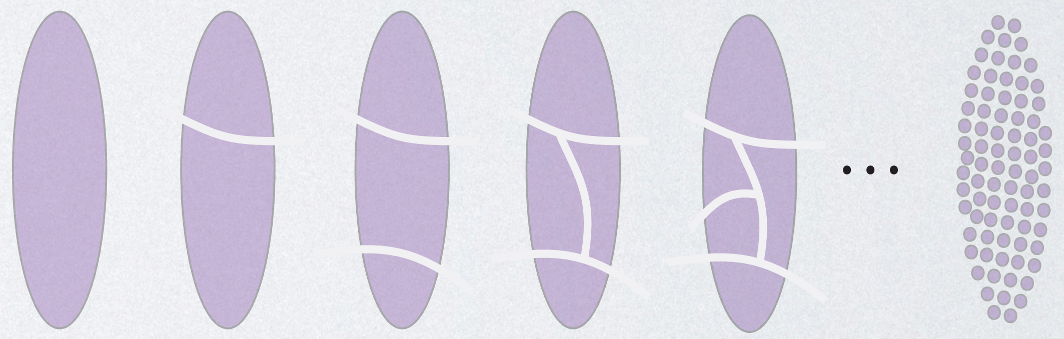
MODEL CHECKING ON CLASSES OF BOUNDED TWIN-WIDTH

Theorem (Bonnet, Kim, Thomassé, Watrigant, *Twin-width I*, 2020)
Given $\phi \in$FO, a graph G of twin-width $\leq d$ **with its merge sequence,**

$G \models \phi$ can be tested in time $O_{\phi,d}(\ |G|\ )$

Ingredients:

1. existence of a treelike decomposition – contraction sequence
2. efficient computation of the decomposition – **missing**



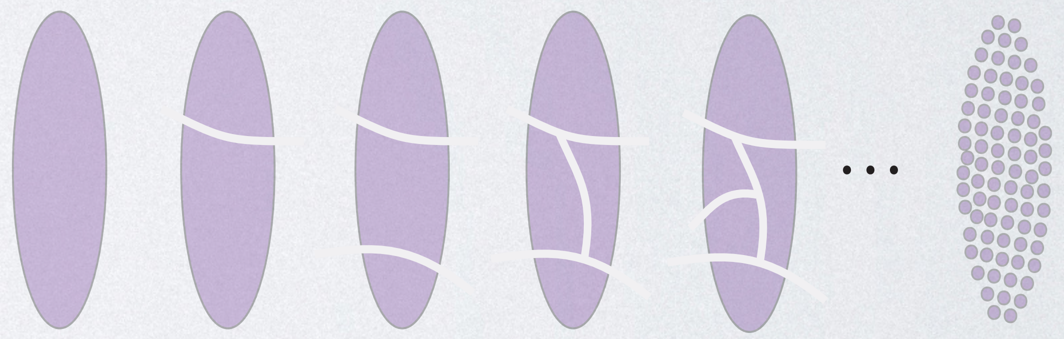
MODEL CHECKING ON CLASSES OF BOUNDED TWIN-WIDTH

Theorem (Bonnet, Kim, Thomassé, Watrigant, *Twin-width I*, 2020)
Given $\phi \in$FO, a graph G of twin-width $\leq d$ **with its merge sequence,**

$$G \models \phi \text{ can be tested in time } \quad O_{\phi,d}(\,|G|\,)$$

Ingredients:

1. existence of a treelike decomposition – contraction sequence
2. efficient computation of the decomposition – **missing**
3. dynamic algorithm – uses locality of FO



ORDERED GRAPHS

$G = (V,E,<)$

$<$ contributes to the logic, and to the twin-width.

Theorem (Bonnet, Ossona de Mendez, Thomassé, Simon, **T.**, 2022)
Given $\phi \in FO$, an ordered graph G of twin-width $\leq d$,

ORDERED GRAPHS

$G = (V,E,<)$

$<$ contributes to the logic, and to the twin-width.

Theorem (Bonnet, Ossona de Mendez, Thomassé, Simon, **T.**, 2022)
Given $\phi \in$FO, an ordered graph G of twin-width $\leq d$,

$$G \models \phi \text{ can be tested in time } \quad O_{\phi,d}(\ |G|^3\ )$$

ORDERED GRAPHS

$G = (V,E,<)$

$<$ contributes to the logic, and to the twin-width.

Theorem (Bonnet, Ossona de Mendez, Thomassé, Simon, **T.**, 2022)
Given $\phi \in FO$, an ordered graph G of twin-width $\leq d$,

$G \models \phi$ can be tested in time $O_{\phi,d}(\ |G|^3\ )$

Ingredients:

ORDERED GRAPHS

$G = (V,E,<)$

$<$ contributes to the logic, and to the twin-width.

Theorem (Bonnet, Ossona de Mendez, Thomassé, Simon, **T.**, 2022)
Given $\phi \in FO$, an ordered graph G of twin-width $\leq d$,

$G \models \phi$ can be tested in time $O_{\phi,d}(\ |G|^3\ )$

Ingredients:

1. existence of a treelike decomposition – by definition

ORDERED GRAPHS

$G = (V,E,<)$

$<$ contributes to the logic, and to the twin-width.

Theorem (Bonnet, Ossona de Mendez, Thomassé, Simon, **T.**, 2022)
Given $\phi \in$ FO, an ordered graph G of twin-width $\leq d$,

$G \models \phi$ can be tested in time $O_{\phi,d}(\,|G|^3\,)$

Ingredients:

1. existence of a treelike decomposition – by definition
2. efficient computation of decomposition – for **ordered graphs** of bounded twin-width

ORDERED GRAPHS

$G = (V,E,<)$

$<$ contributes to the logic, and to the twin-width.

Theorem (Bonnet, Ossona de Mendez, Thomassé, Simon, **T.**, 2022)
Given $\phi \in$FO, an ordered graph G of twin-width $\leq d$,

$$G \models \phi \text{ can be tested in time } \quad O_{\phi,d}(\ |G|^3\ )$$

Ingredients:

1. existence of a treelike decomposition – by definition
2. efficient computation of decomposition – for **ordered graphs** of bounded twin-width
3. dynamic algorithm [Twin-width I]

CLASSES OF ORDERED GRAPHS

Theorem (Bonnet, O. de Mendez, Thomassé, Simon, **T.**, 2022).
Let C be a hereditary class of *ordered* graphs. Then:

C has bounded twin-width $^{*}\Leftrightarrow$ FO-model checking is fpt on C

*assuming AW[$*$]$\neq$FPT

GRAND UNIFICATION

Common generalization of Sparsity and Twin-width

monotone	*hereditary*
bounded treewidth	bounded cliquewidth
excluding a minor	…
…	bounded twin-width
nowhere dense	?

OUTLINE

1. The model checking problem
2. Sparsity: monotone case
3. Twin-width: ordered case
4. **Monadic dependence**
5. Flip-breakability
6. Stability: orderless case

TRANSDUCTION

TRANSDUCTION

specified by a formula $\phi(x,y)$ with k color predicates

TRANSDUCTION

specified by a formula $\phi(x,y)$ with k color predicates

e.g. $\phi(x,y) = \exists z.\ \mathrm{Red}(z) \wedge (x \sim z) \wedge (y \sim z)$

TRANSDUCTION

specified by a formula $\phi(x,y)$ with k color predicates

e.g. $\phi(x,y) = \exists z.\ \mathrm{Red}(z) \wedge (x \sim z) \wedge (y \sim z)$

1. **Input**:
graph G

TRANSDUCTION

specified by a formula $\phi(x,y)$ with k color predicates

e.g. $\phi(x,y) = \exists z.\ \text{Red}(z) \wedge (x \sim z) \wedge (y \sim z)$

1. **Input**: graph G

2. Color G with k colors

TRANSDUCTION

specified by a formula $\phi(x,y)$ with k color predicates

e.g. $\phi(x,y) = \exists z.\, \mathrm{Red}(z) \wedge (x \sim z) \wedge (y \sim z)$

1. **Input**: graph G

2. Color G with k colors

3. Define new edges using $\phi(x,y)$

TRANSDUCTION

specified by a formula $\phi(x,y)$ with k color predicates

e.g. $\phi(x,y) = \exists z.$ Red$(z) \wedge (x \sim z) \wedge (y \sim z)$

1. **Input**: graph G

2. Color G with k colors

3. Define new edges using $\phi(x,y)$

4. **Output:** any induced subgraph H

TRANSDUCTION

specified by a formula $\phi(x,y)$ with k color predicates

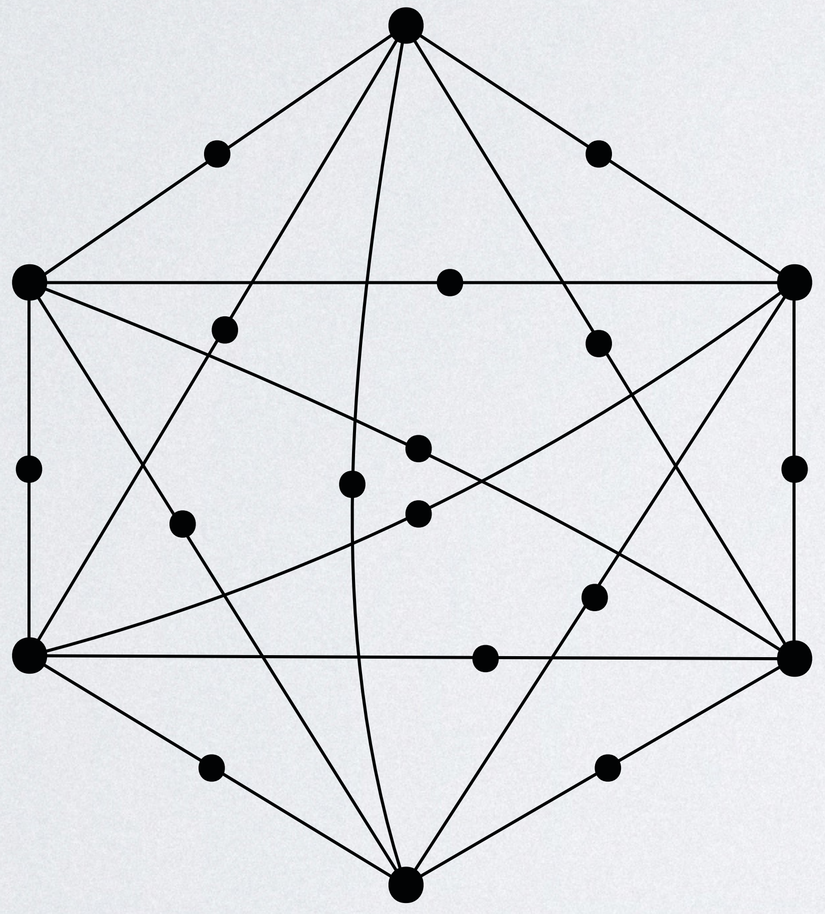
e.g. $\phi(x,y) = \exists z.$ Red$(z) \wedge (x \sim z) \wedge (y \sim z)$

1. **Input**: graph G

2. Color G with k colors

3. Define new edges using $\phi(x,y)$

4. **Output:** any induced subgraph H



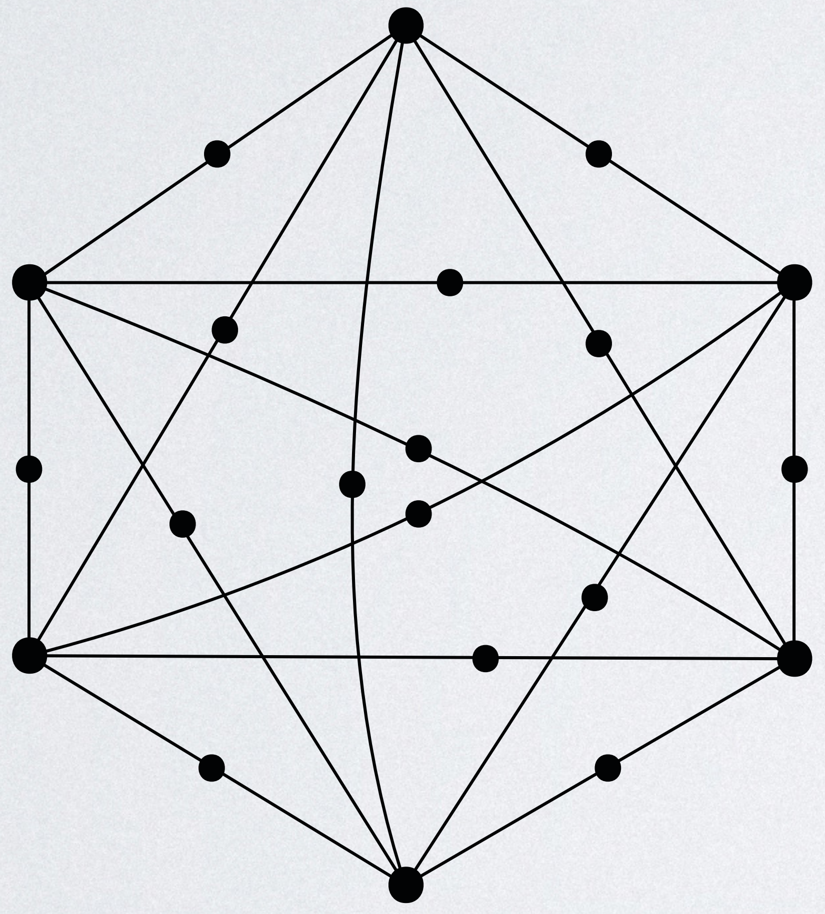
1-subdivided 6-clique

TRANSDUCTION

specified by a formula $\phi(x,y)$ with k color predicates

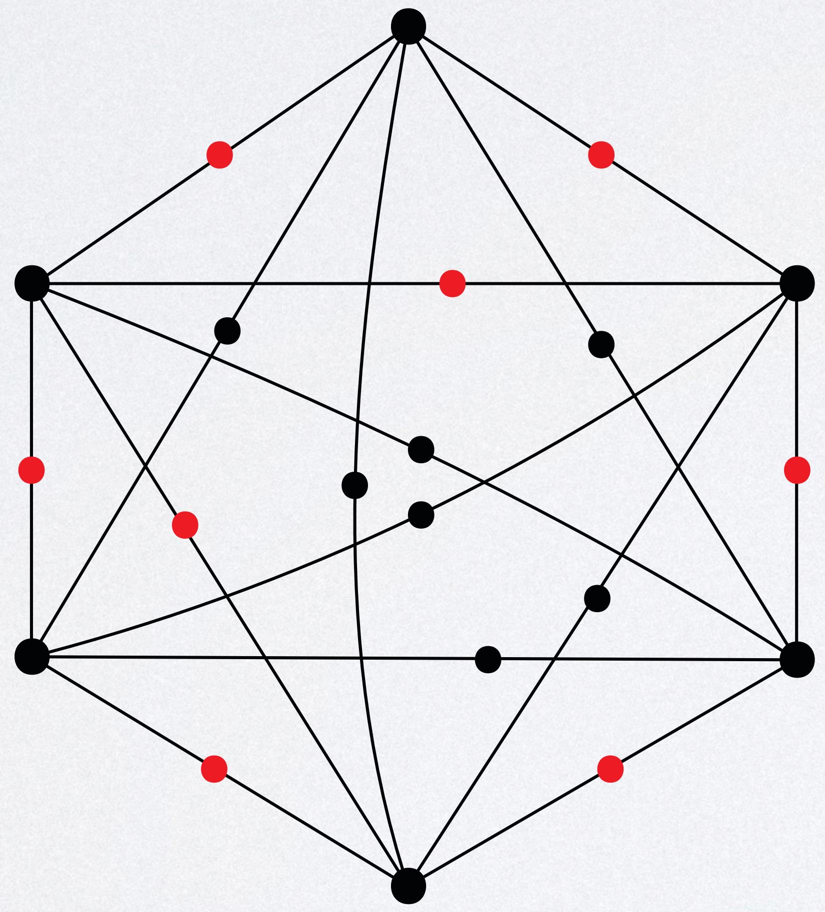
e.g. $\phi(x,y) = \exists z.$ Red$(z) \wedge (x \sim z) \wedge (y \sim z)$

1. **Input:** graph G



1-subdivided 6-clique

2. Color G with k colors



3. Define new edges using $\phi(x,y)$

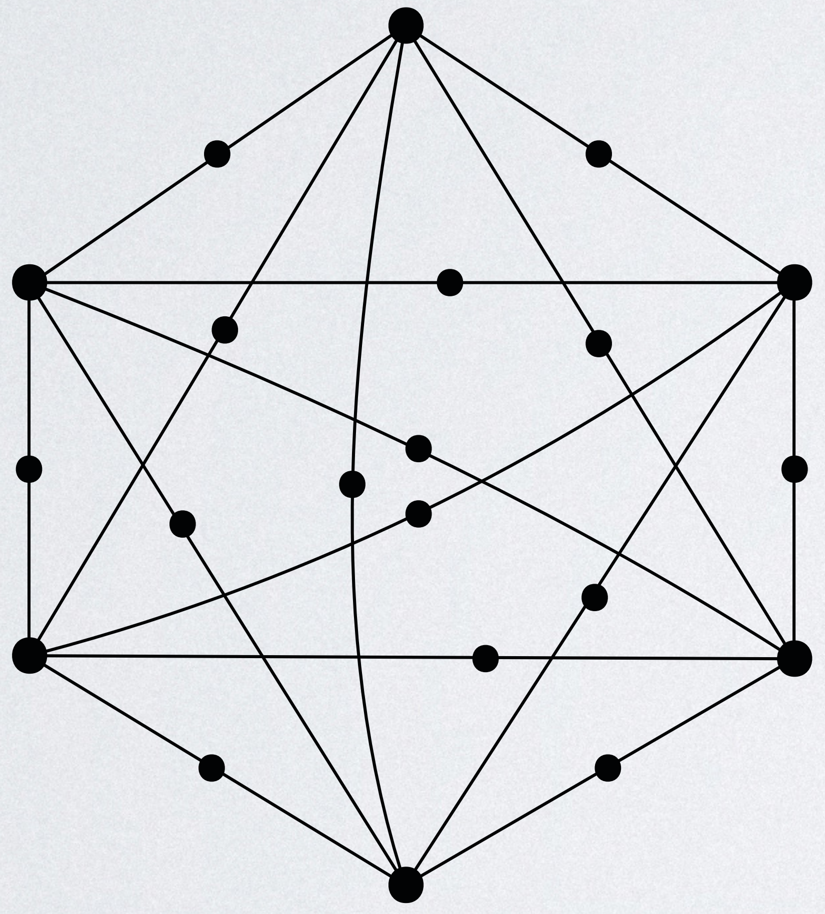
4. **Output:** any induced subgraph H

TRANSDUCTION

specified by a formula $\phi(x,y)$ with k color predicates

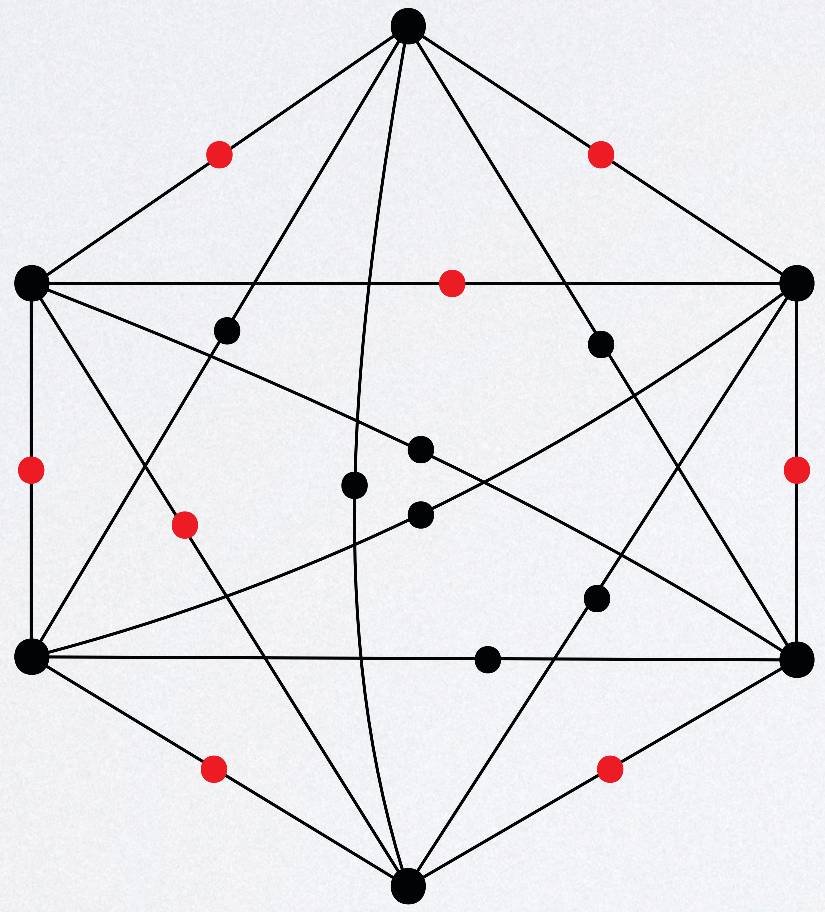
e.g. $\phi(x,y) = \exists z.\ \textcolor{red}{\mathrm{Red}}(z) \wedge (x \sim z) \wedge (y \sim z)$

1. **Input**: graph G

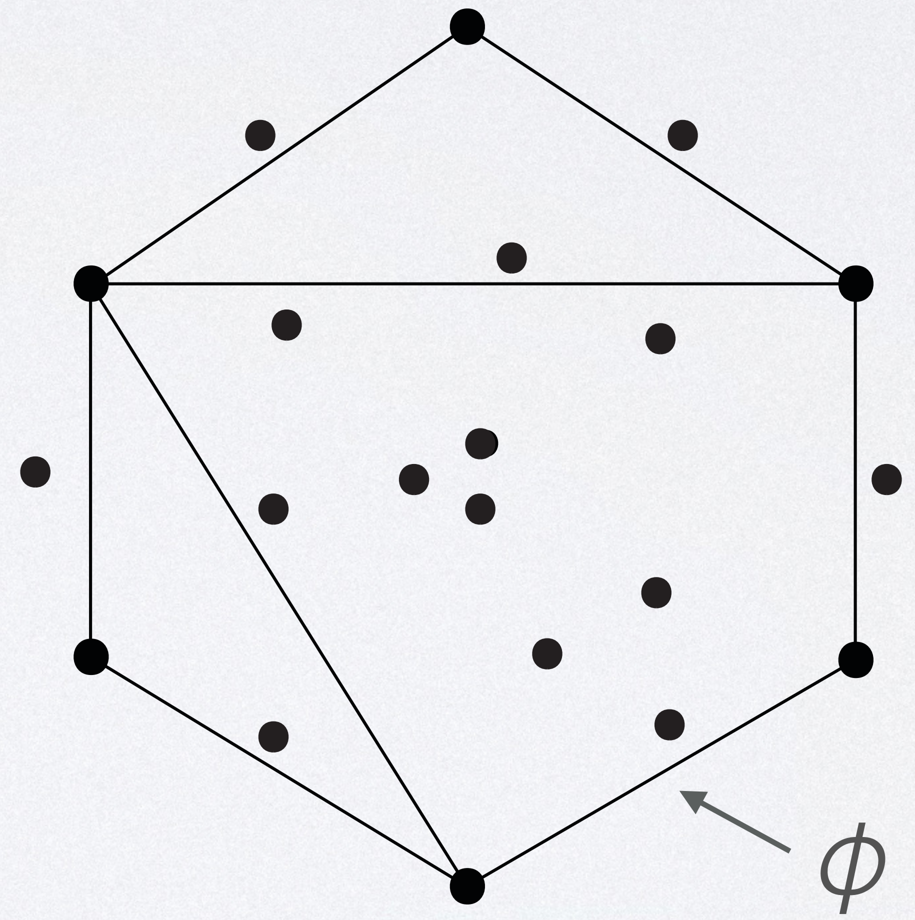


1-subdivided 6-clique

2. Color G with k colors



3. Define new edges using $\phi(x,y)$



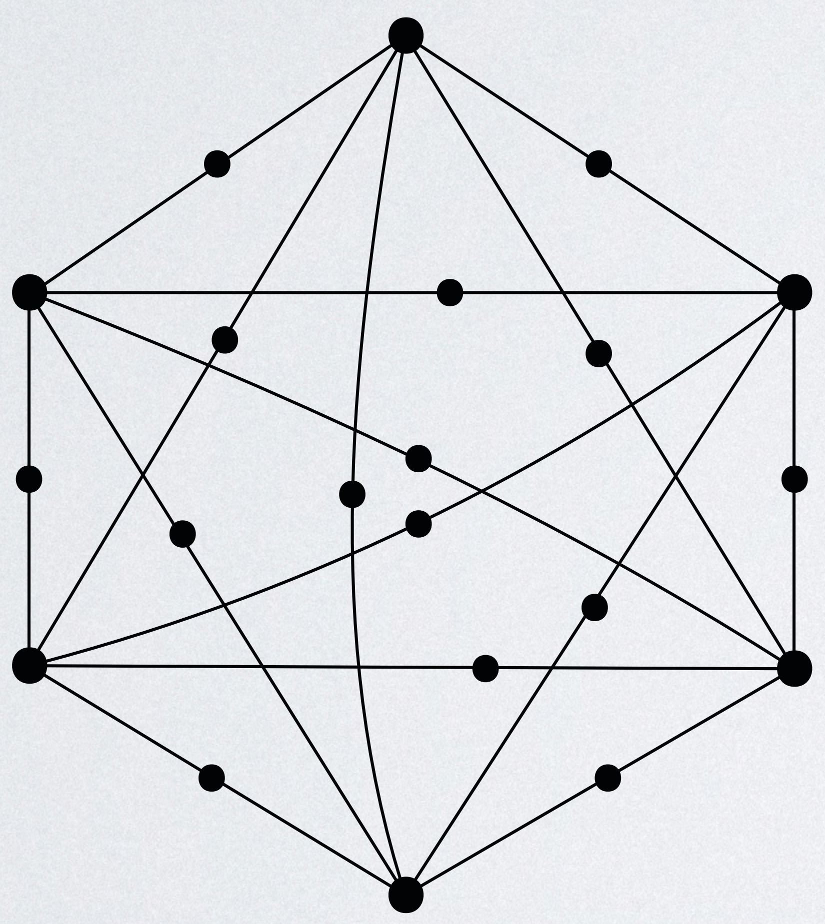
4. **Output:** any induced subgraph H

TRANSDUCTION

specified by a formula $\phi(x,y)$ with k color predicates

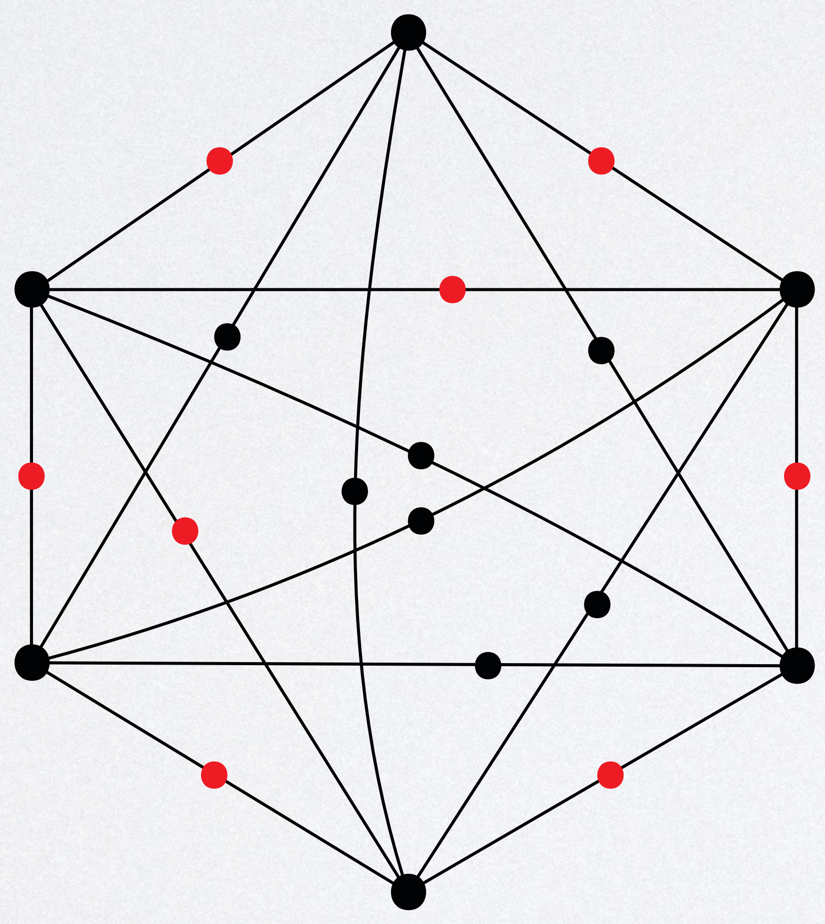
e.g. $\phi(x,y) = \exists z.\ \textcolor{red}{\mathrm{Red}}(z) \wedge (x \sim z) \wedge (y \sim z)$

1. **Input**: graph G

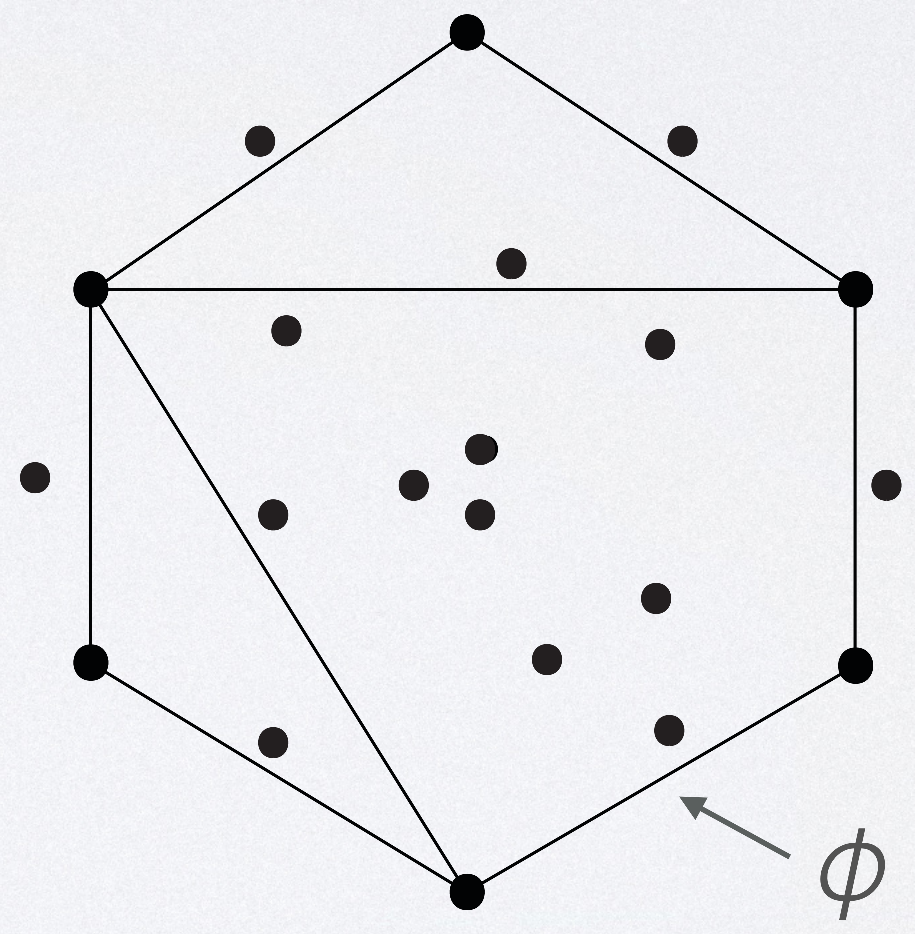


1-subdivided 6-clique

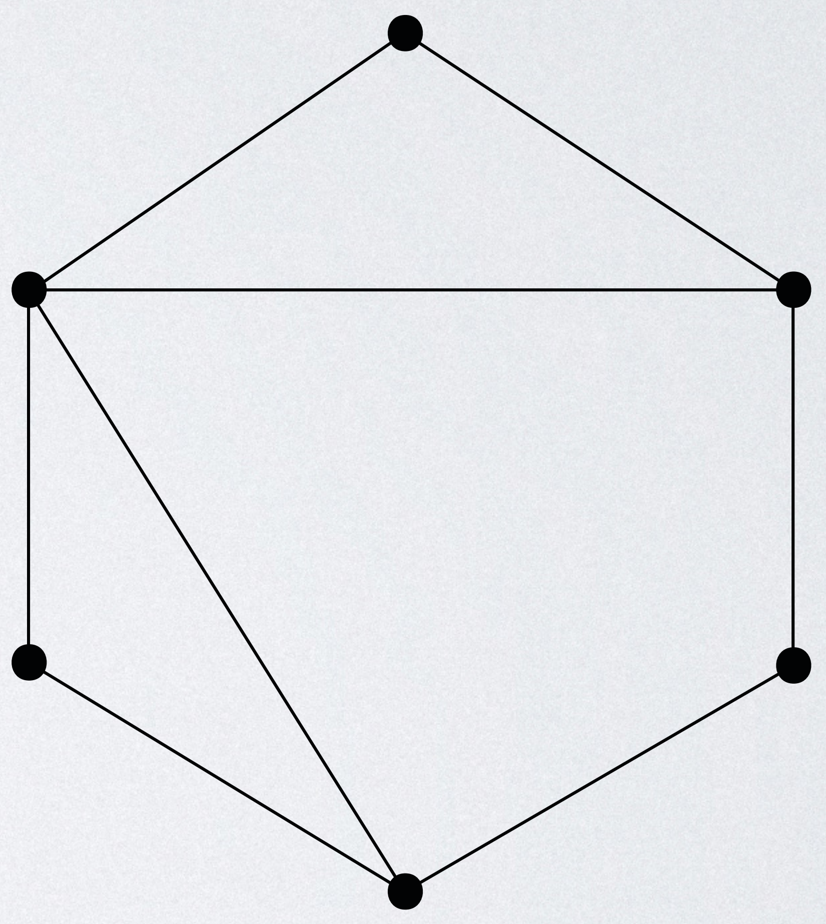
2. Color G with k colors



3. Define new edges using $\phi(x,y)$



4. **Output:** any induced subgraph H

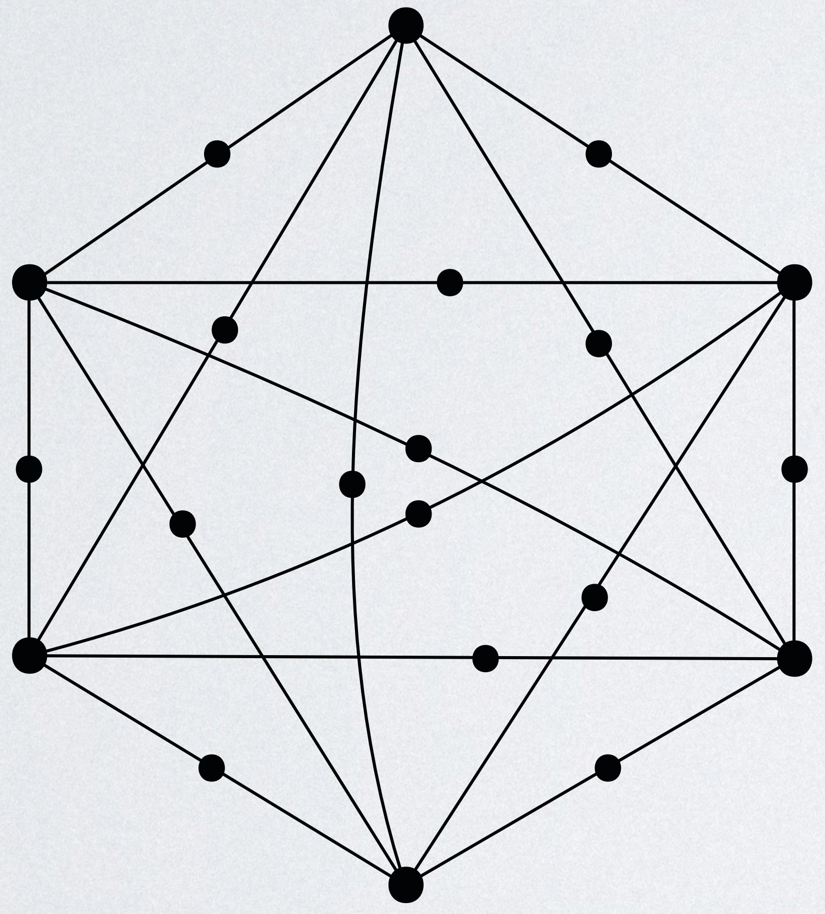


TRANSDUCTION

specified by a formula $\phi(x,y)$ with k color predicates

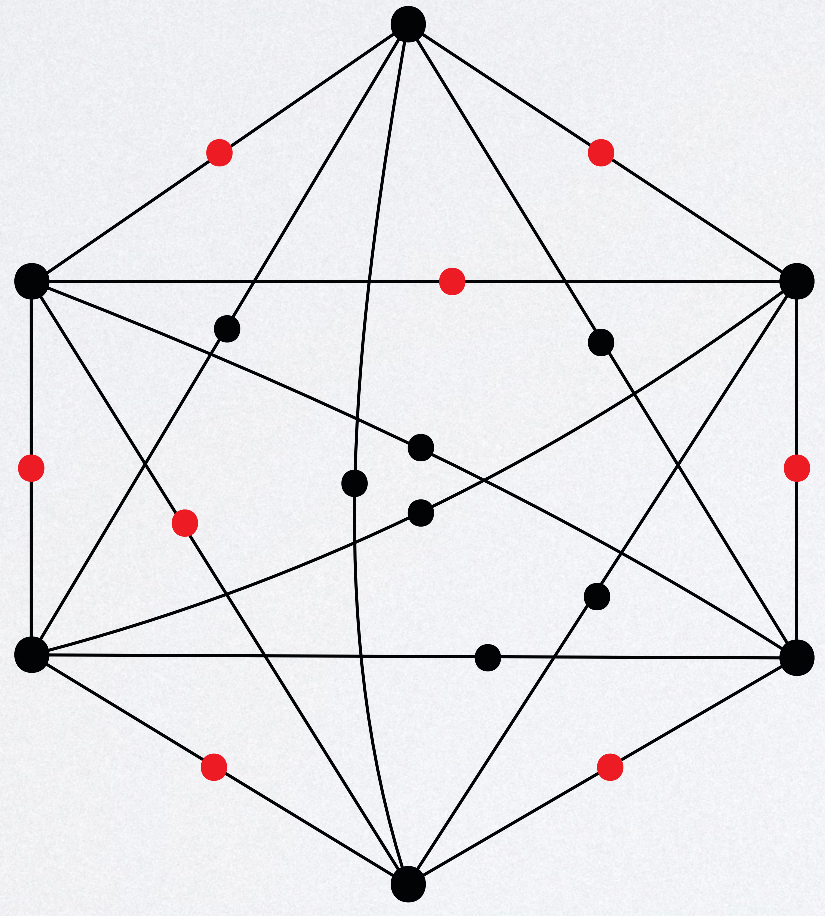
e.g. $\phi(x,y) = \exists z.\, \mathrm{Red}(z) \wedge (x \sim z) \wedge (y \sim z)$

1. **Input:** graph G

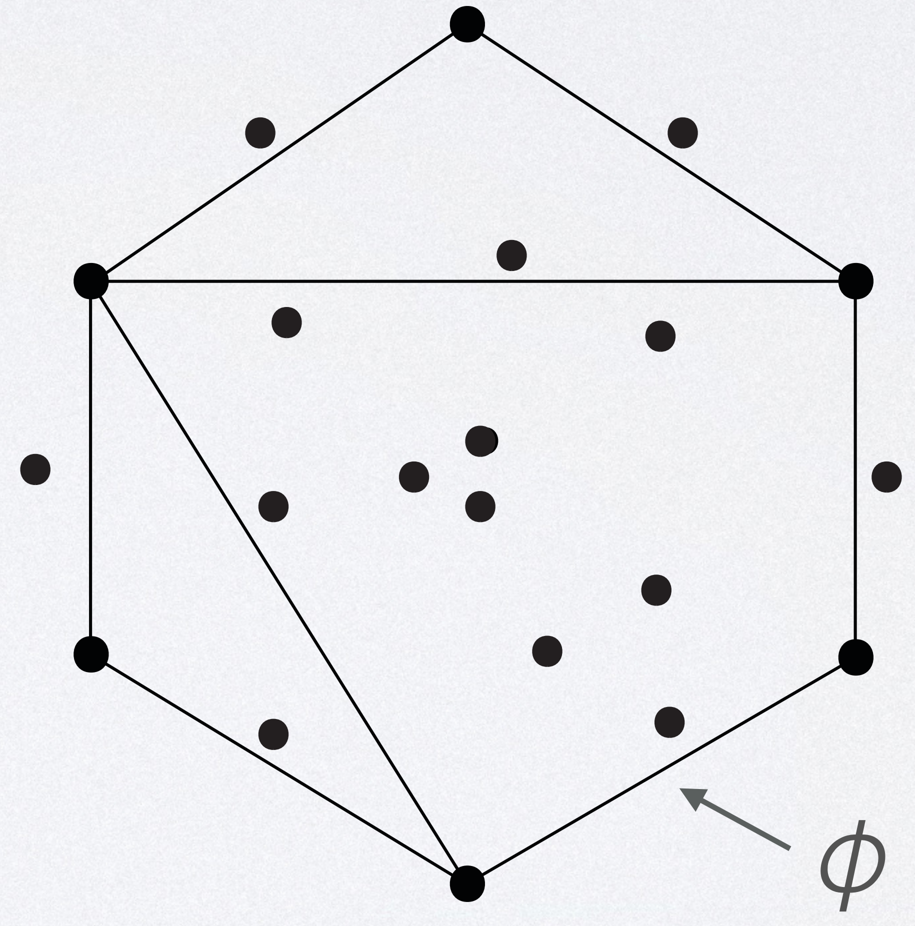


1-subdivided 6-clique

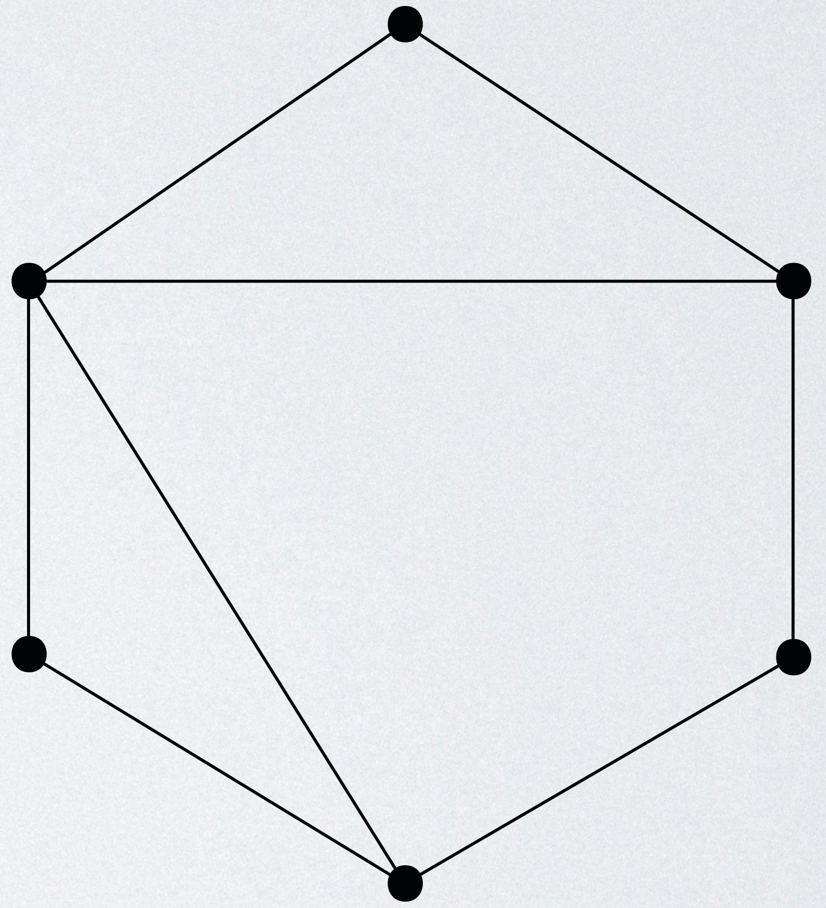
2. Color G with k colors



3. Define new edges using $\phi(x,y)$



4. **Output:** any induced subgraph H



Write: $G \rightsquigarrow_{\phi} H$

TRANSDUCTION QUASI-ORDER

TRANSDUCTION QUASI-ORDER

graph classes

C transduces D if for some FO formula $\phi(x,y)$

TRANSDUCTION QUASI-ORDER

graph classes

C *transduces* D if for some FO formula $\phi(x,y)$

$$\forall H \in D.\ \ \exists G \in C.\ \ G \rightsquigarrow_{\phi} H$$

TRANSDUCTION QUASI-ORDER

graph classes

C *transduces* D if for some FO formula $\phi(x,y)$

$$\forall H{\in}D.\ \ \exists G{\in}C.\ \ G \rightsquigarrow_{\phi} H$$

Write: $C \geq_{FO} D$

TRANSDUCTION QUASI-ORDER

graph classes

C *transduces* D if for some FO formula $\phi(x,y)$

$$\forall H \in D.\ \ \exists G \in C.\ \ G \rightsquigarrow_{\phi} H$$

Write: $C \geq_{FO} D$ **Fact:** $\geq_{FO}$ is transitive

TRANSDUCTION QUASI-ORDER

graph classes

C transduces D if for some FO formula $\phi(x,y)$

$$\forall H \in D.\ \ \exists G \in C.\ \ G \rightsquigarrow_{\phi} H$$

Write: $C \geq_{FO} D$ **Fact:** $\geq_{FO}$ is transitive

{1-subdivided cliques} $\geq_{FO}$ {all graphs} $\geq_{FO}$ any class

MONADIC DEPENDENCE

MONADIC DEPENDENCE

Definition (Shelah, 1986) A graph class C is *monadically dependent* if C does not transduce the class of all graphs:

MONADIC DEPENDENCE

Definition (Shelah, 1986) A graph class C is *monadically dependent* if C does not transduce the class of all graphs:

$$C \not<_{FO} \{\text{all graphs}\}$$

monotone	*hereditary*
bounded treewidth	bounded cliquewidth
excluding a minor	. . .
. . .	bounded twin-width
nowhere dense	**monadically dependent**

Theorem [Podewski-Ziegler '78, Adler-Adler'10, Grohe-Kreutzer-Siebertz '14]
For ***monotone*** graph classes:

monadically dependent $\Leftrightarrow$ nowhere dense $^{*}\Leftrightarrow$ tractable

Theorem [Podewski-Ziegler '78, Adler-Adler'10, Grohe-Kreutzer-Siebertz '14]
For ***monotone*** graph classes:

monadically dependent $\Leftrightarrow$ nowhere dense $^*\Leftrightarrow$ tractable

Theorem [Bonnet, O. de Mendez, Thomassé, Simon, **T.** '22]
For hereditary classes of ***ordered*** graphs:

monadically dependent $\Leftrightarrow$ bounded twin-width $^*\Leftrightarrow$ tractable

Theorem [Podewski-Ziegler '78, Adler-Adler'10, Grohe-Kreutzer-Siebertz '14]
For ***monotone*** graph classes:

monadically dependent $\Leftrightarrow$ nowhere dense $^{*}\!\Leftrightarrow$ tractable

Theorem [Bonnet, O. de Mendez, Thomassé, Simon, **T.** '22]
For hereditary classes of ***ordered*** graphs:

monadically dependent $\Leftrightarrow$ bounded twin-width $^{*}\!\Leftrightarrow$ tractable

Conjecture [Pilipczuk, Siebertz, **T.** '16]
For all hereditary graph classes:

monadically dependent $\Leftrightarrow$ tractable

OUTLINE

1. The model checking problem
2. Sparsity: monotone case
3. Twin-width: ordered case
4. Monadic dependence
5. **Flip-breakability**
6. Stability: orderless case

	monotone: deletion	*hereditary:* flip
radius ∞ MSO	bounded treewidth $\Leftrightarrow$ ∞-**deletion**-breakable	bounded cliquewidth $\Leftrightarrow$ ∞-**flip**-breakable
finite radii FO	nowhere dense $\Leftrightarrow$ **deletion**-breakable	monadically dependent $\Leftrightarrow$ **flip**-breakable

∞-DELETION-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is ∞-*deletion-breakable* if

∞-DELETION-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is ∞-*deletion-breakable* if

$\exists k \geq 1.\ \exists U : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

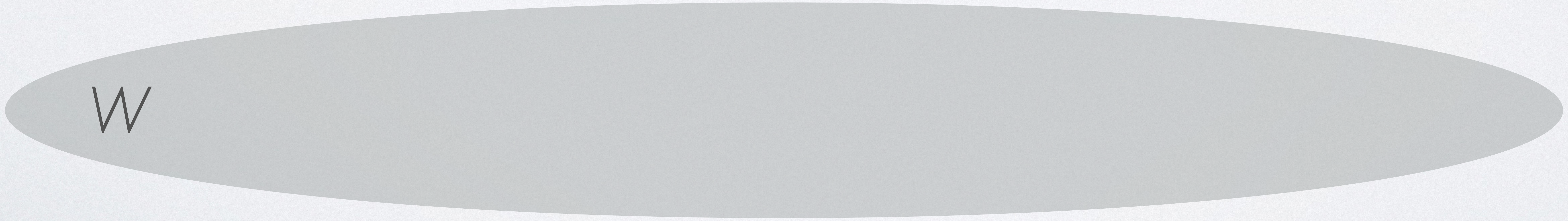
∞-DELETION-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is ∞-*deletion-breakable* if

$\exists k \geq 1.\ \exists U : \mathbf{N} \rightarrow \mathbf{N}$ – unbounded function s.t.

$\forall G \in C \quad \forall W \subseteq V(G)$



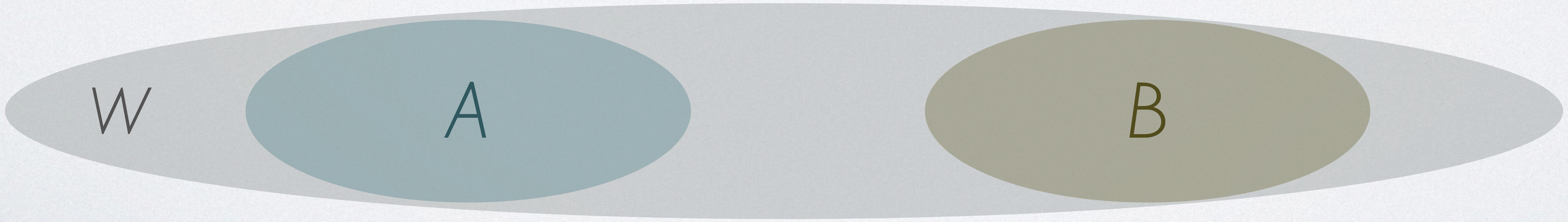
∞-DELETION-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is *∞-deletion-breakable* if

$\exists k \geq 1.\ \exists U : \mathbf{N} \rightarrow \mathbf{N}$ – unbounded function s.t.

$\forall G \in C \quad \forall W \subseteq V(G) \quad \exists A, B \subseteq W,\ |A| = |B| \geq U(|W|)$



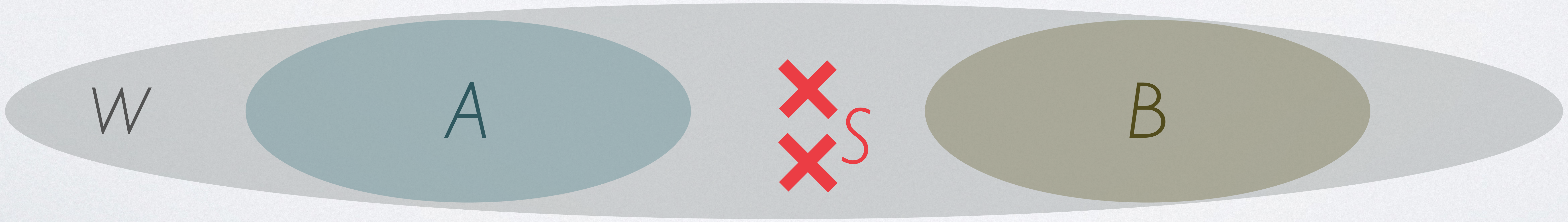
∞-DELETION-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is *∞-deletion-breakable* if

$\exists k \geq 1.\ \exists U : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\forall G \in C \quad \forall W \subseteq V(G) \quad \exists A, B \subseteq W,\ |A| = |B| \geq U(|W|) \quad \exists S \subseteq V(G),\ |S| \leq k$



∞-DELETION-BREAKABILITY

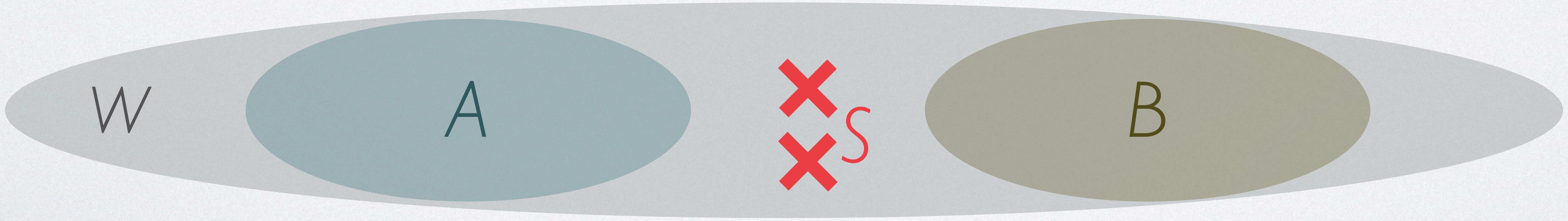
[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is *∞-deletion-breakable* if

$\exists k \geq 1.\ \exists U : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\forall G \in C \quad \forall W \subseteq V(G) \quad \exists A, B \subseteq W,\ |A| = |B| \geq U(|W|) \quad \exists S \subseteq V(G),\ |S| \leq k$

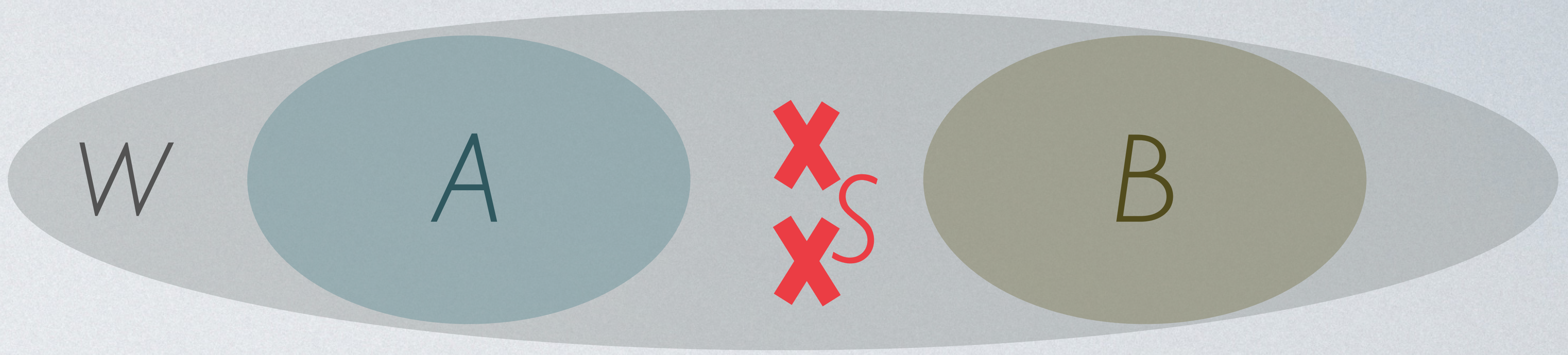
$\text{dist}(A,B) = \infty$ in $G - S$



$\exists k \geq 1.\ \exists U : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\exists A, B \subseteq W,\ |A| = |B| \geq U(|W|)$ $\exists S \subseteq V(G),\ |S| \leq k$

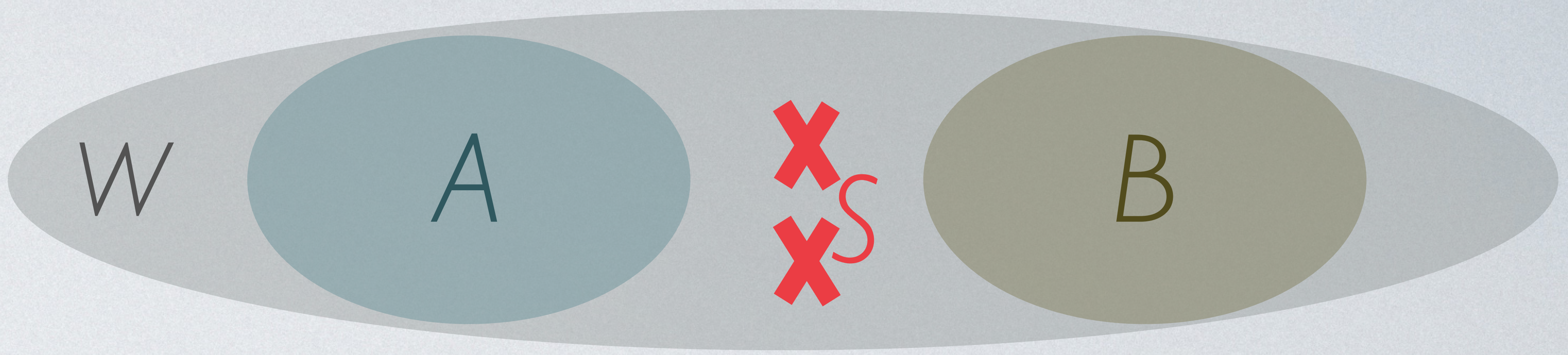
$\mathrm{dist}(A,B) = \infty$ in $G - S$



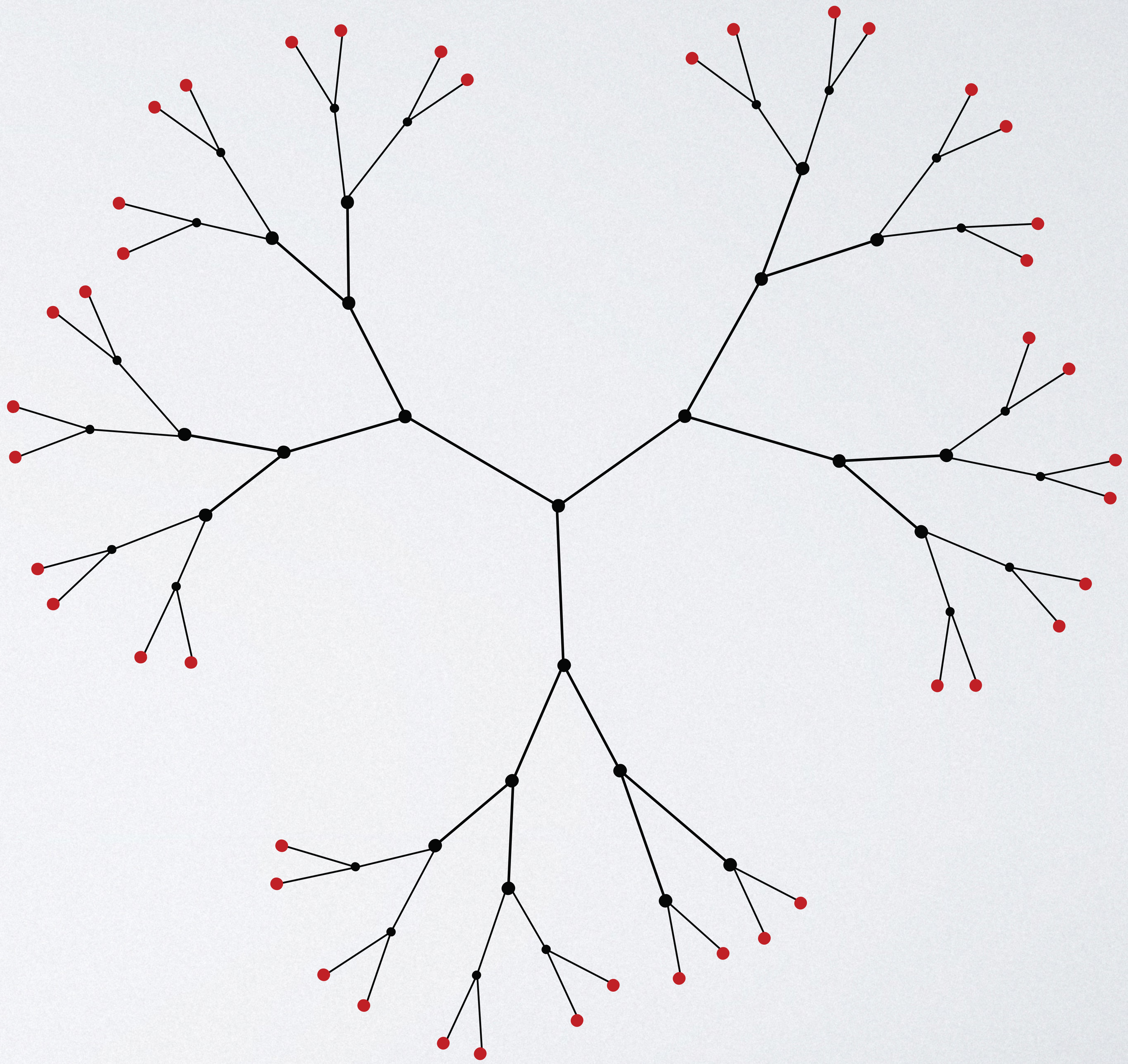
$\exists k \geq 1.\ \exists \mathrm{U} : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\exists A, B \subseteq W,\ |A| = |B| \geq \mathrm{U}(|W|) \quad \exists S \subseteq V(G),\ |S| \leq k$

$\mathrm{dist}(A,B) = \infty$ in $G - S$



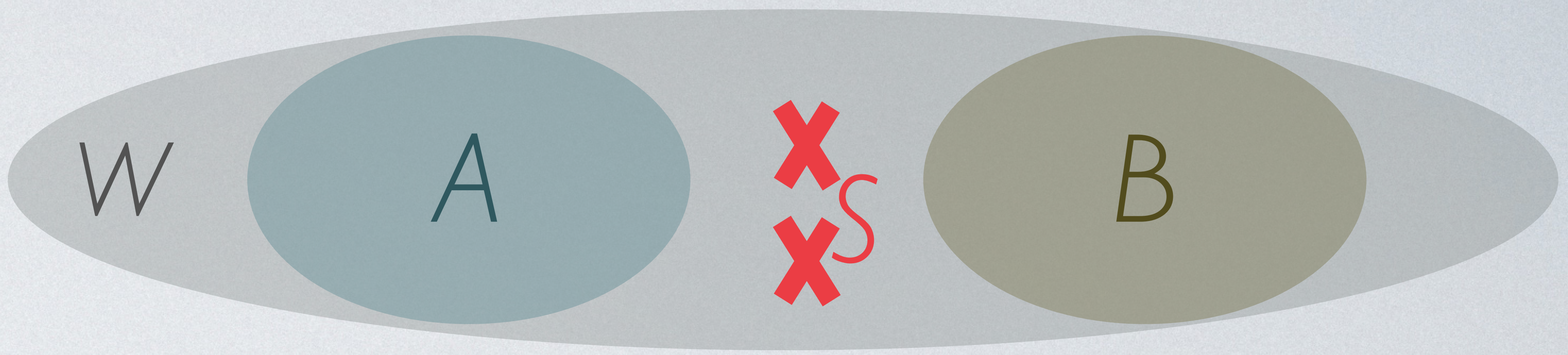
Example. Class of trees: $k := 1,\ \mathrm{U}(n) := n/3$



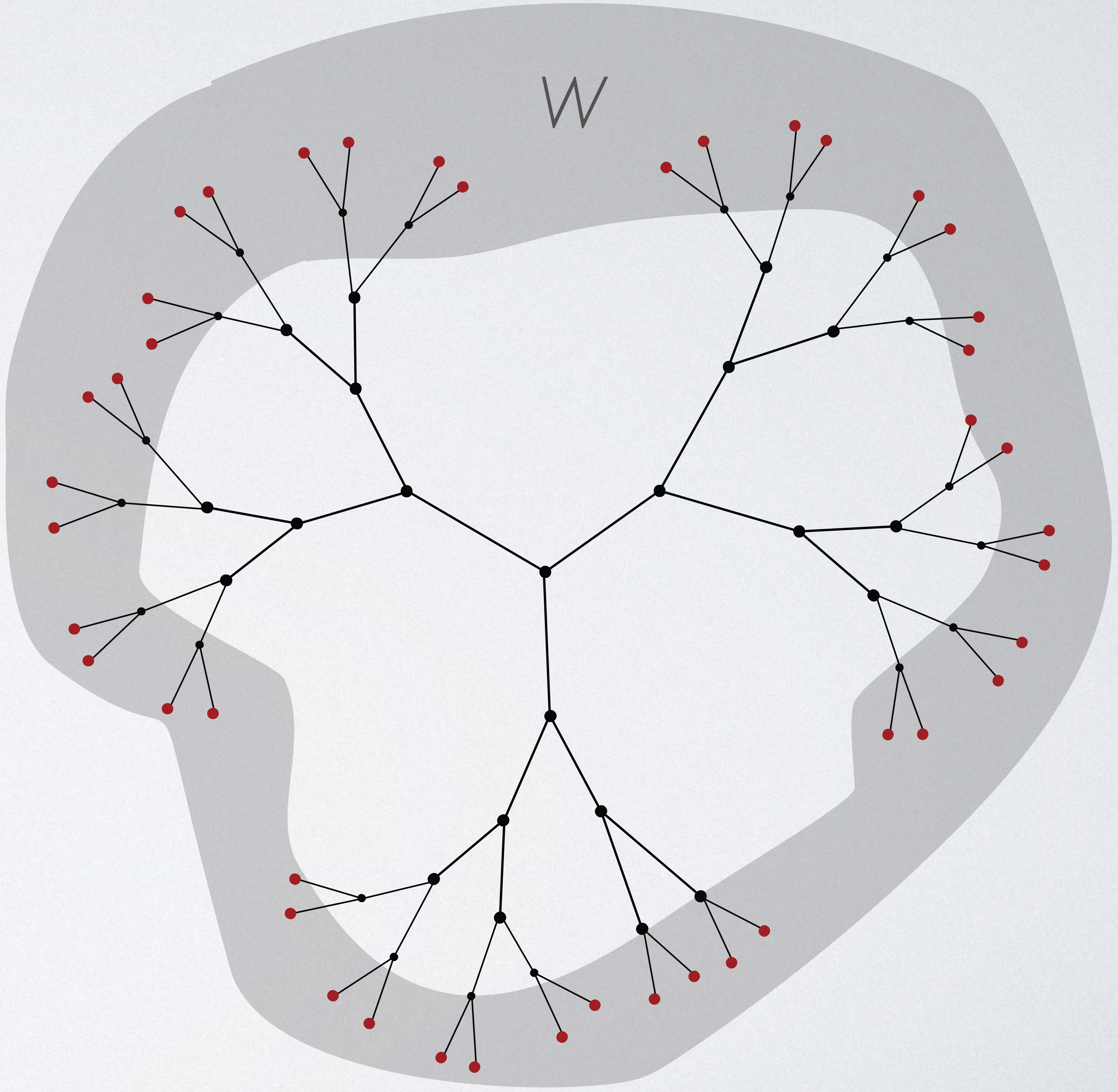
$\exists k \geq 1.\ \exists \mathrm{U} : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\exists A, B \subseteq W,\ |A| = |B| \geq \mathrm{U}(|W|)$ $\quad$ $\exists S \subseteq V(G),\ |S| \leq k$

$\mathrm{dist}(A,B) = \infty$ in $G - S$



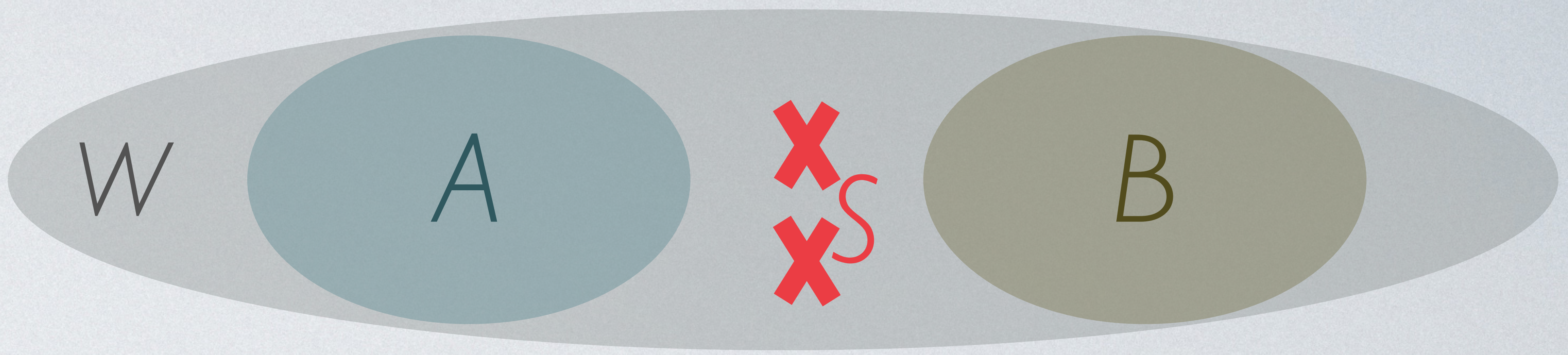
Example. Class of trees: $k := 1,\ \mathrm{U}(n) := n/3$



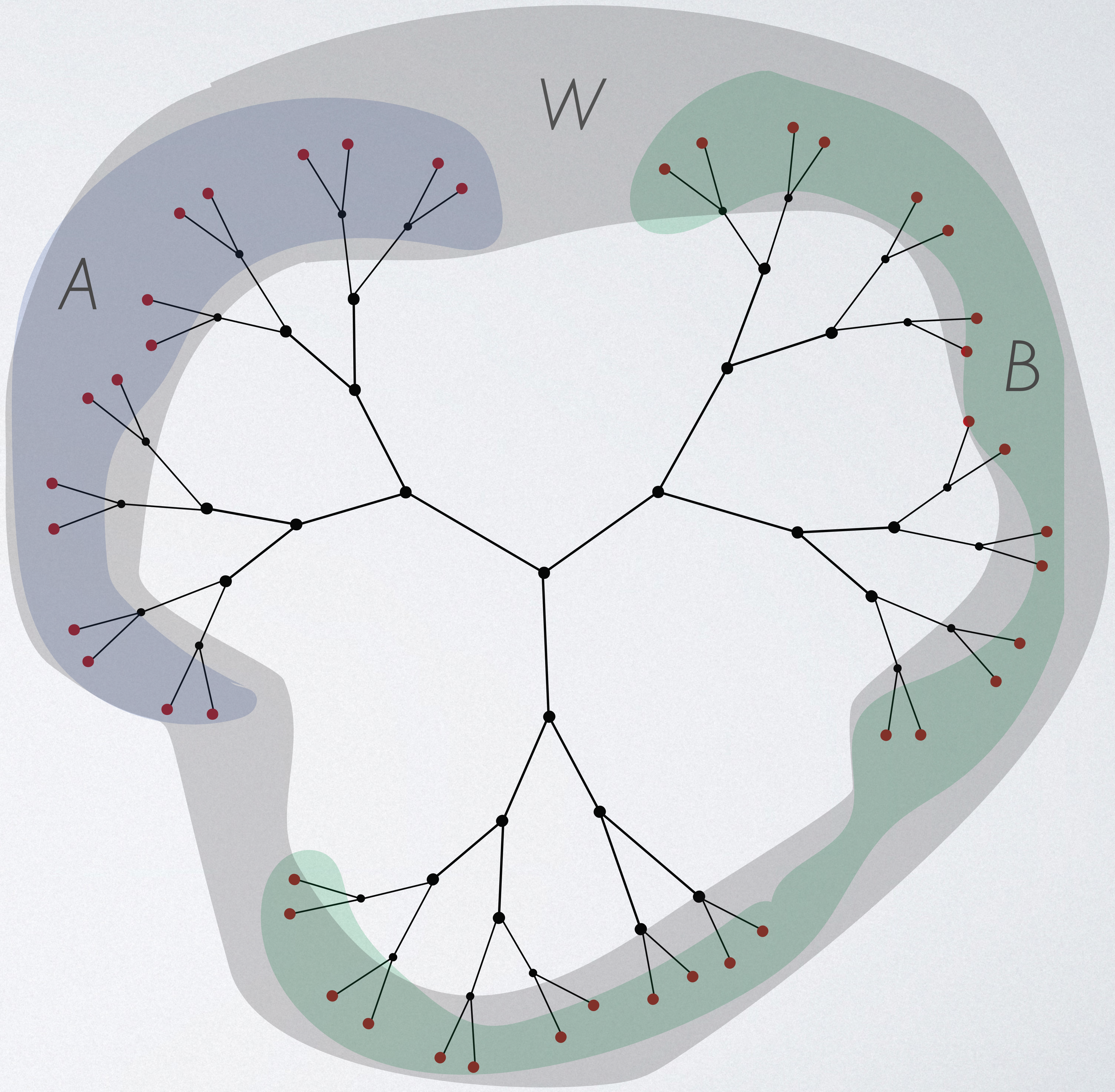
$\exists k \geq 1.\ \exists U : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\exists A, B \subseteq W,\ |A| = |B| \geq U(|W|)$ $\exists S \subseteq V(G),\ |S| \leq k$

$\mathrm{dist}(A,B) = \infty$ in $G - S$



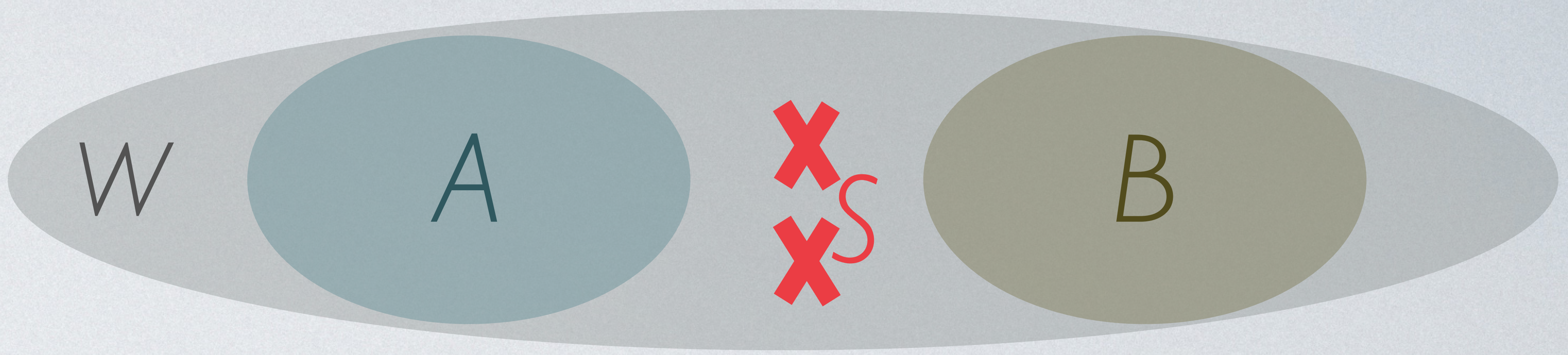
Example. Class of trees: $k := 1,\ U(n) := n/3$



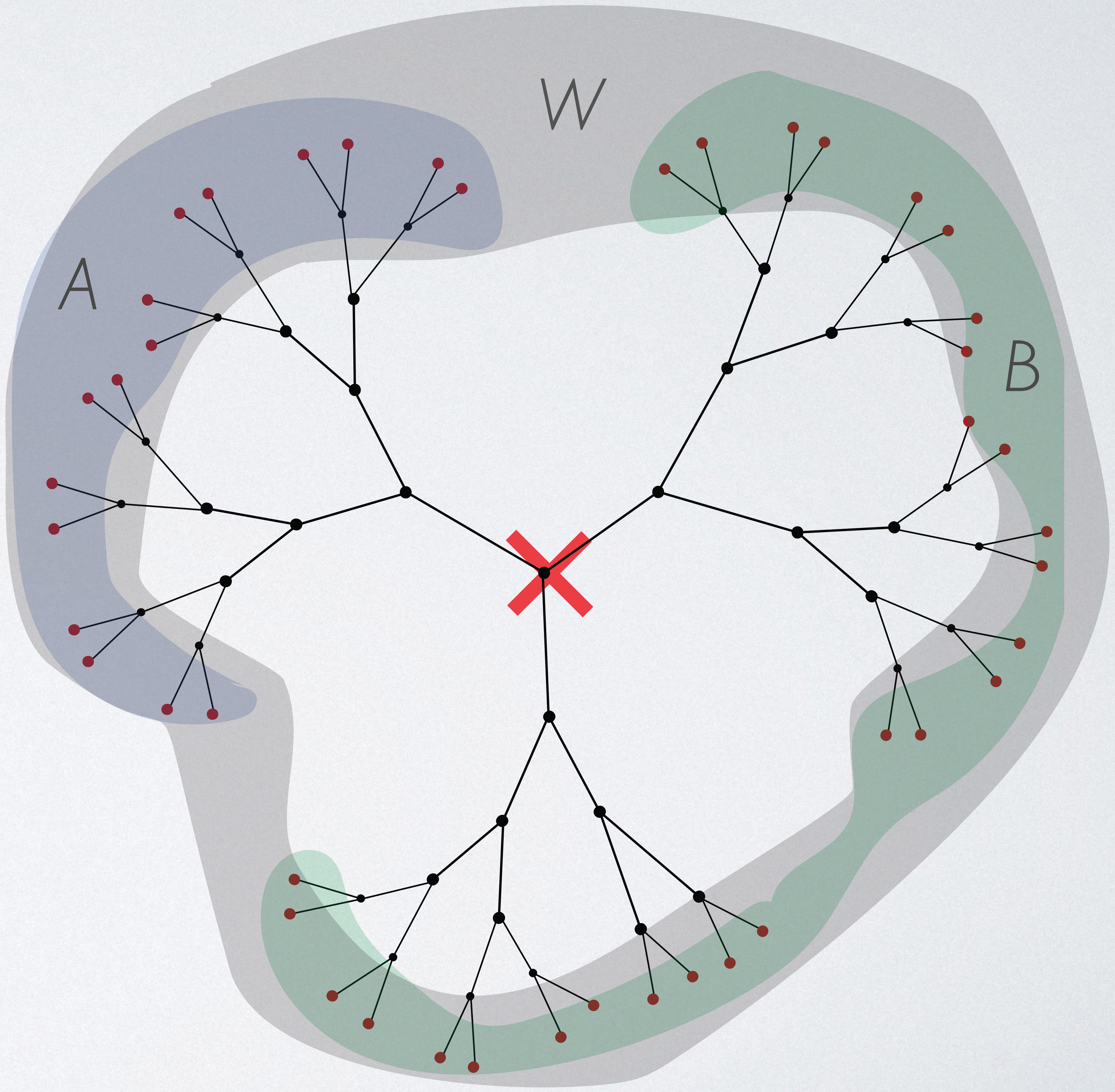
$\exists k \geq 1.\ \exists \mathrm{U} : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\exists A, B \subseteq W,\ |A| = |B| \geq \mathrm{U}(|W|) \qquad \exists S \subseteq V(G),\ |S| \leq k$

$\mathrm{dist}(A,B) = \infty$ in $G - S$



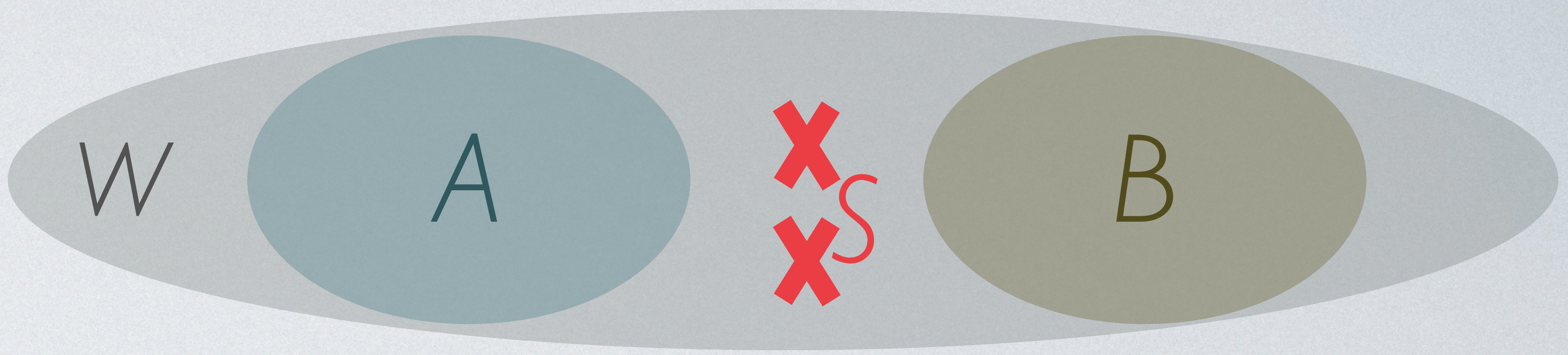
Example. Class of trees: $k := 1$, $\mathrm{U}(n) := n/3$



$\exists k \geq 1.\ \exists U : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

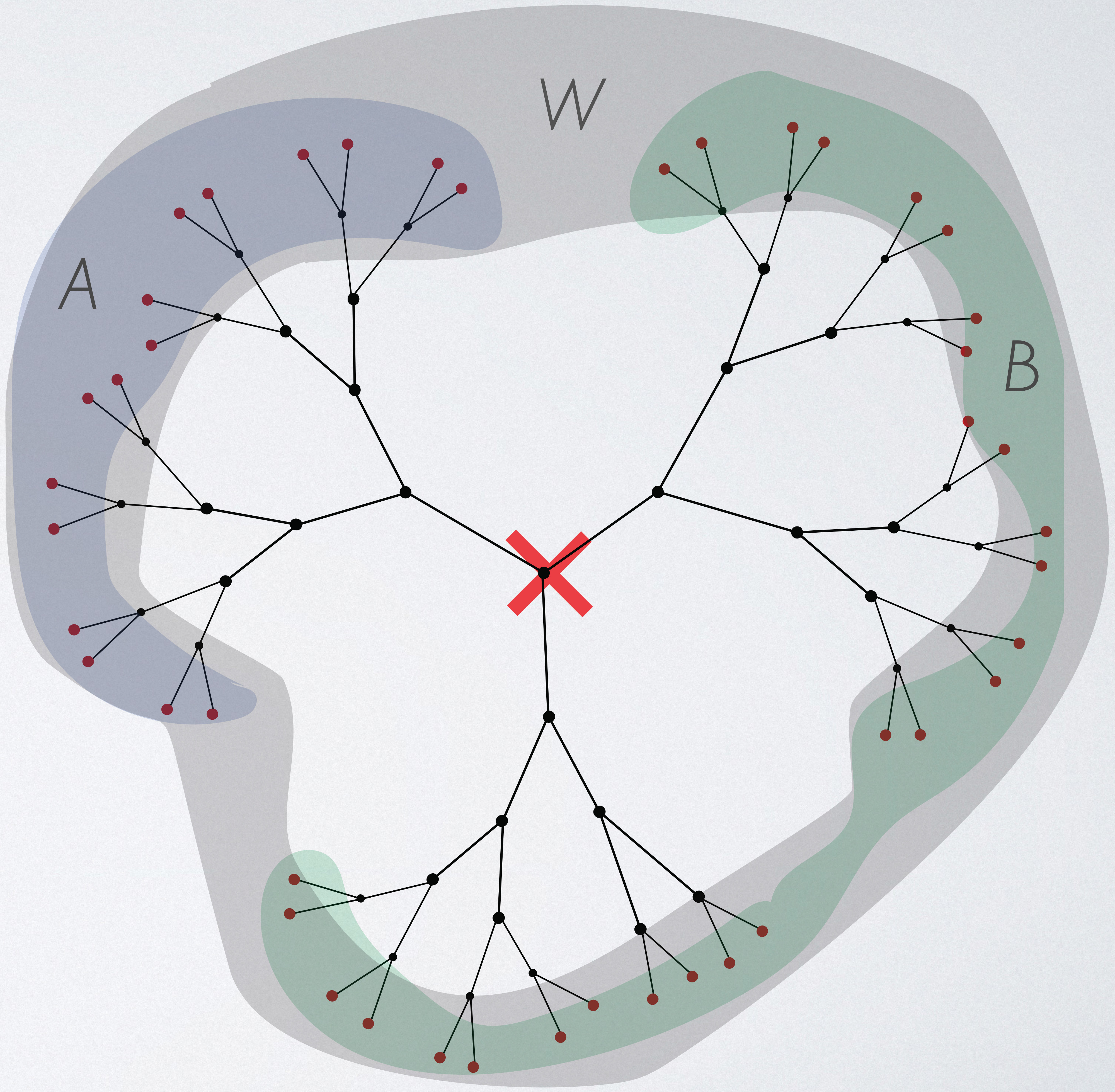
$\exists A, B \subseteq W,\ |A| = |B| \geq U(|W|) \quad \exists S \subseteq V(G),\ |S| \leq k$

$\mathrm{dist}(A,B) = \infty$ in $G - S$



Example. Class of trees: $k := 1$, $U(n) := n/3$

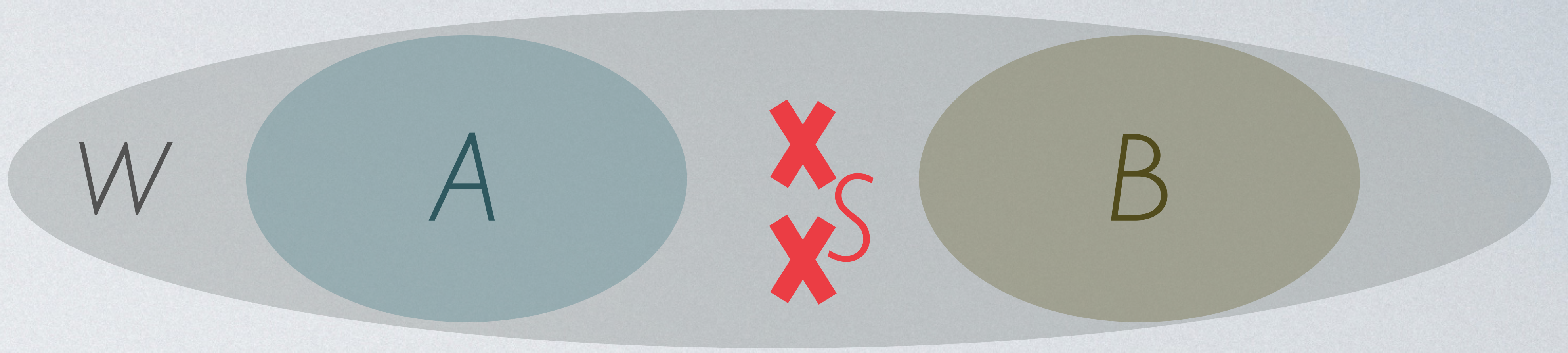
Example. Class of graphs of treewidth $\leq t$:
$k := t+1$, $U(n) := n/3$.

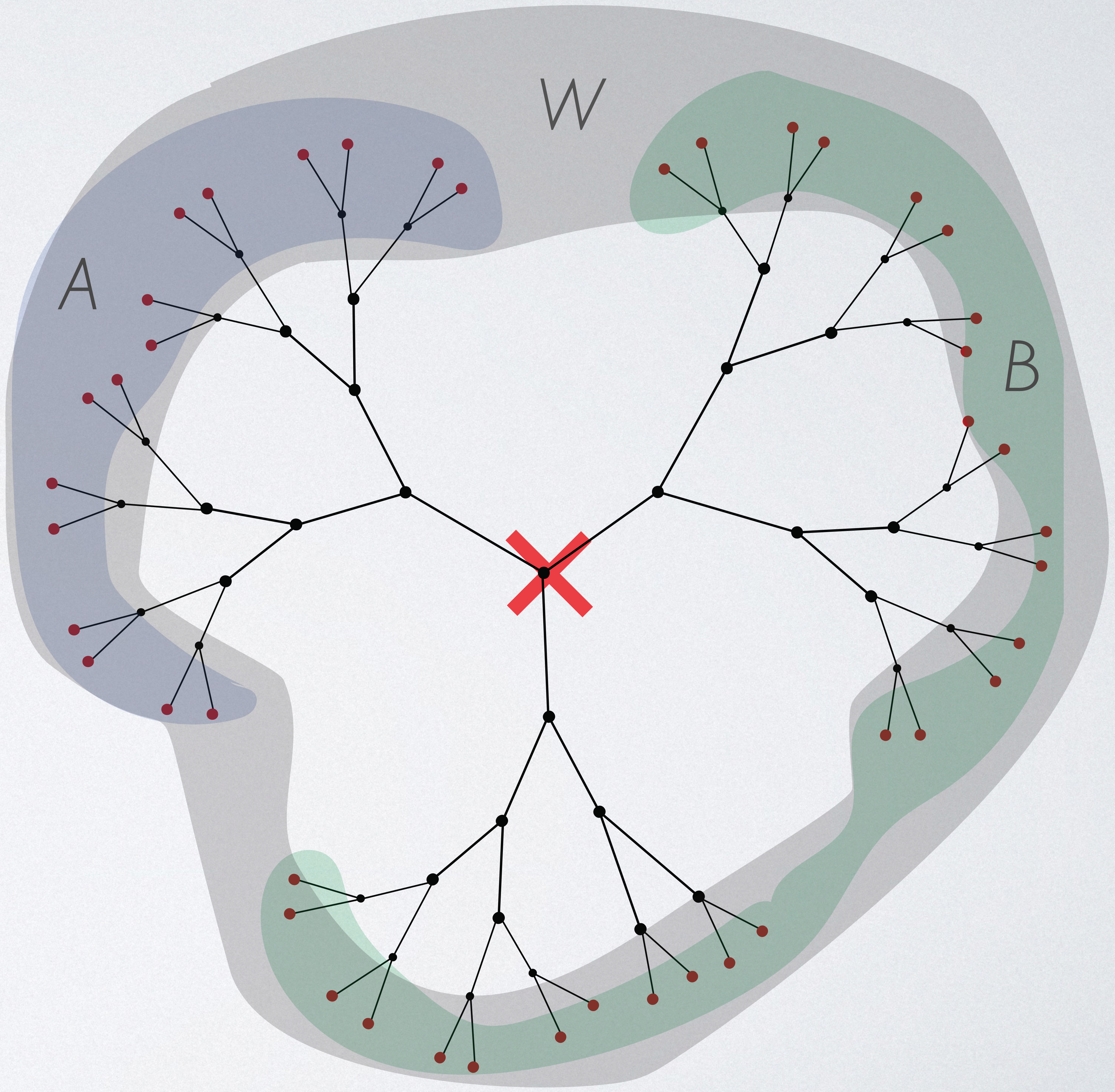


$\exists k \geq 1.\ \exists \mathrm{U} : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\exists A, B \subseteq W,\ |A| = |B| \geq \mathrm{U}(|W|)$ $\quad \exists S \subseteq V(G),\ |S| \leq k$

$\mathrm{dist}(A,B) = \infty$ in $G - S$





Example. Class of trees: $k := 1$, $\mathrm{U}(n) := n/3$

Example. Class of graphs of treewidth $\leq t$: $k := t+1$, $\mathrm{U}(n) := n/3$.

Theorem. C is ∞-deletion-breakable $\Leftrightarrow$

C has bounded tree-width.

DELETION-BREAKABILITY

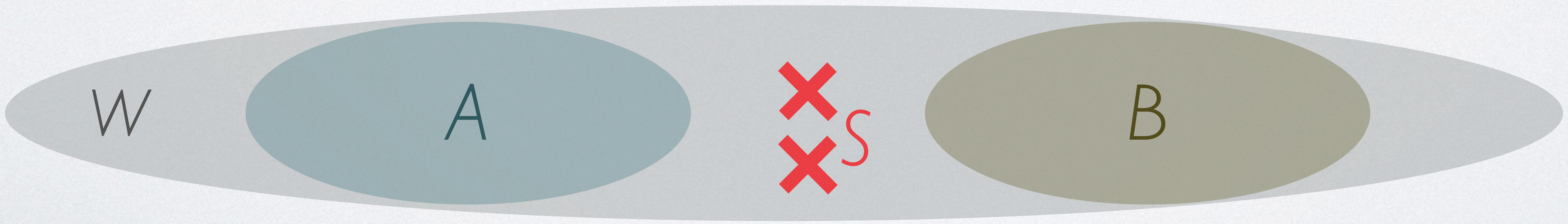
[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is *deletion-breakable* if

$\forall r \geq 1.\ \exists k_r \geq 1.\ \exists U_r : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\forall G \in C \quad \forall W \subseteq V(G) \quad \exists A,B \subseteq W,\ |A|=|B| \geq U_r(|W|), \quad \exists S \subseteq V(G), |S| \leq k_r$

dist(A,B)≥*r* in $G - S$



$\forall r \geq 1.\ \exists k_r \geq 1.\ \exists U_r : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$$\forall G \in C \quad \forall W \subseteq V(G) \quad \exists A, B \subseteq W,\ |A| = |B| \geq U_r(|W|), \quad \exists S \subseteq V(G), |S| \leq k_r$$

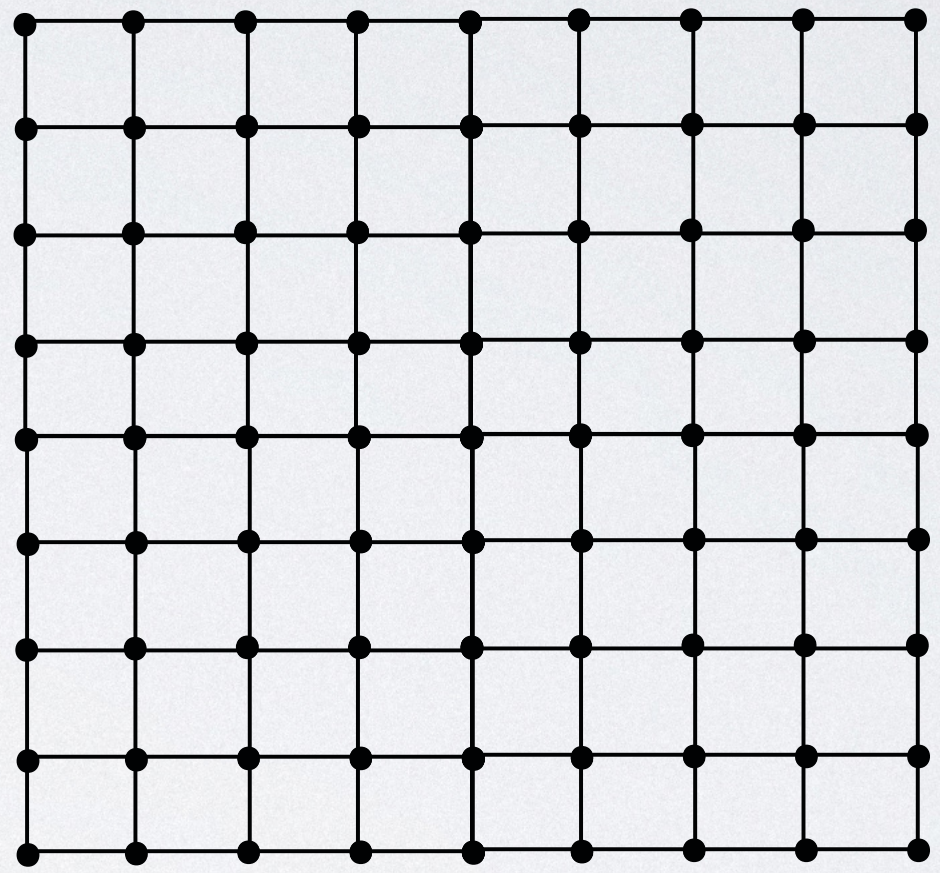
$$\mathrm{dist}(A,B) \geq r \ \text{ in } \ G - S$$

$\forall r \geq 1.\ \exists k_r \geq 1.\ \exists U_r : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$$\forall G \in C \quad \forall W \subseteq V(G) \quad \exists A, B \subseteq W,\ |A| = |B| \geq U_r(|W|), \quad \exists S \subseteq V(G), |S| \leq k_r$$

$$\mathrm{dist}(A,B) \geq r \ \text{ in } \ G - S$$

Example. Class of grids:
$k_r := 0, \quad U_r(n) := \Omega(n/r^2)$



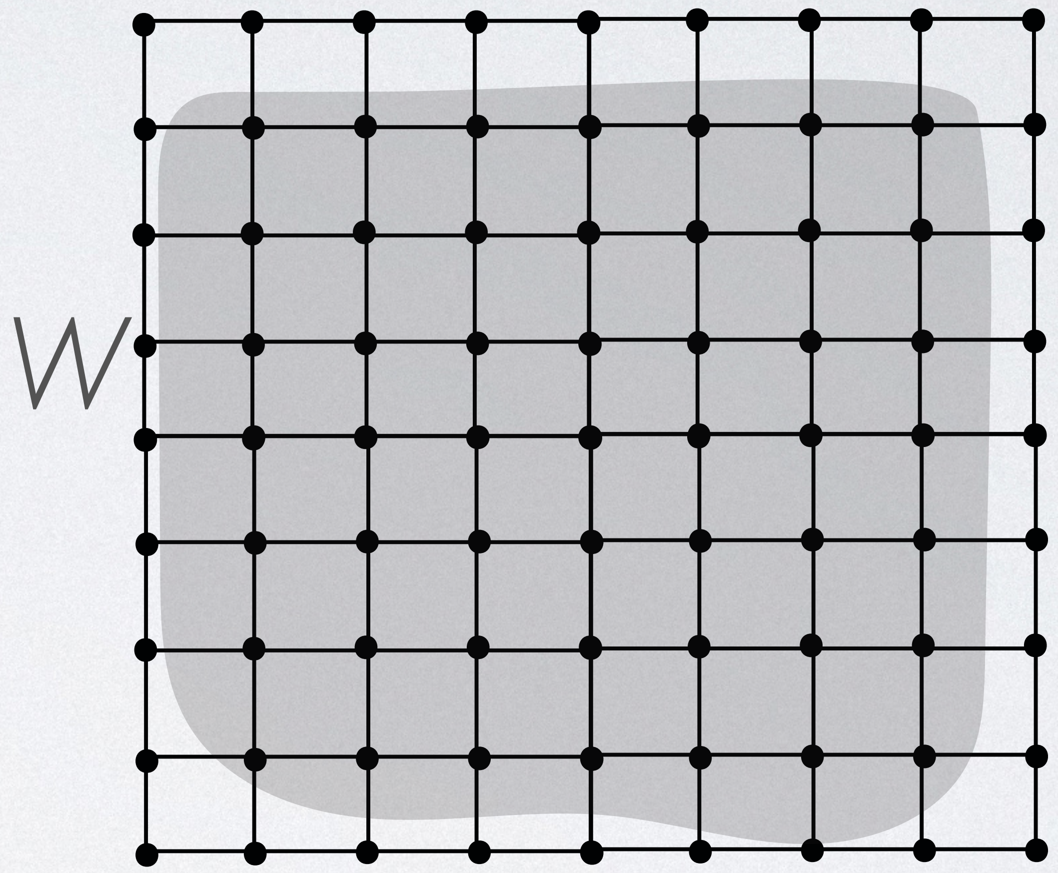
$\forall r \geq 1.\ \exists k_r \geq 1.\ \exists \mathrm{U}_r : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$$\forall G \in C \quad \forall W \subseteq V(G) \quad \exists A, B \subseteq W,\ |A| = |B| \geq \mathrm{U}_r(|W|), \quad \exists S \subseteq V(G),\ |S| \leq k_r$$

$$\mathrm{dist}(A,B) \geq r \text{ in } G - S$$

Example. Class of grids:
$k_r := 0$, $\mathrm{U}_r(n) := \Omega(n/r^2)$

– every set $W \subseteq V(G)$ contains



$\forall r \geq 1.\ \exists k_r \geq 1.\ \exists U_r : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

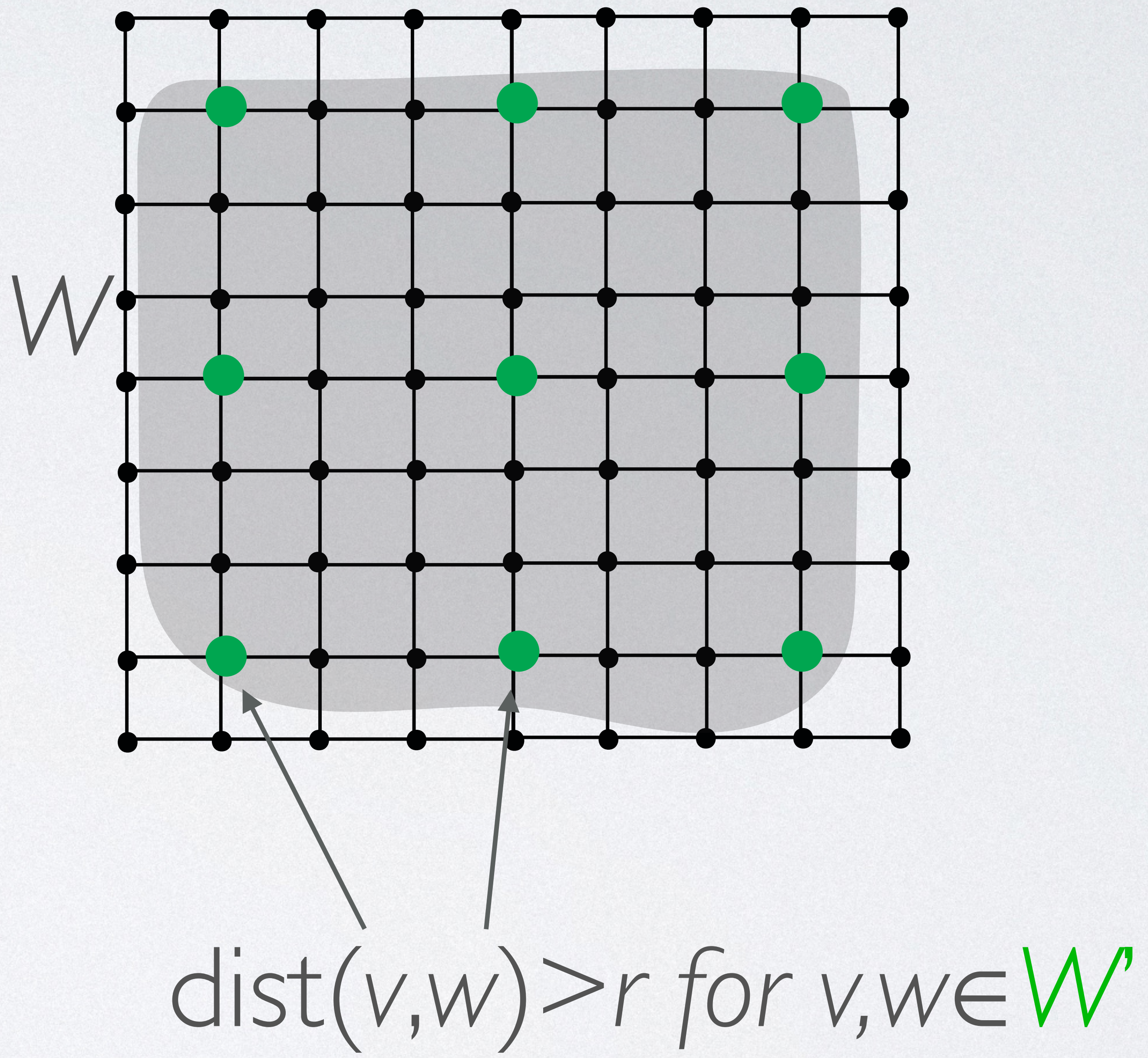
$$\forall G \in C \quad \forall W \subseteq V(G) \quad \exists A,B \subseteq W,\ |A|=|B| \geq U_r(|W|), \quad \exists S \subseteq V(G), |S| \leq k_r$$

$$\text{dist}(A,B) \geq r \ \text{ in } \ G{-}S$$

Example. Class of grids:

$k_r := 0, \quad U_r(n) := \Omega(n/r^2)$

– every set $W \subseteq V(G)$ contains
some r-independent set W' of size $\Omega(n/r^2)$



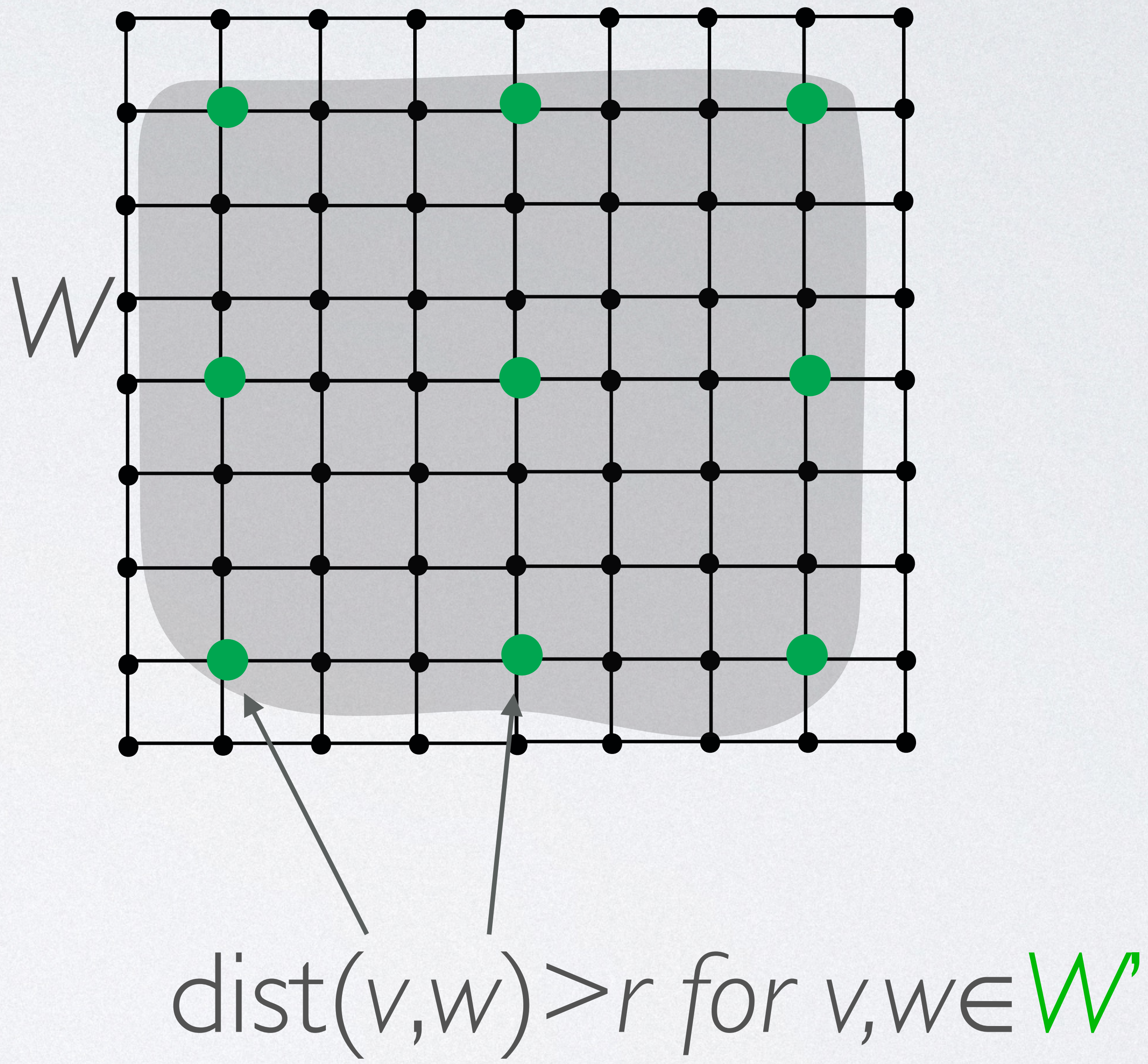
$\forall r \geq 1.\ \exists k_r \geq 1.\ \exists U_r : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$$\forall G \in C \quad \forall W \subseteq V(G) \quad \exists A, B \subseteq W,\ |A| = |B| \geq U_r(|W|), \quad \exists S \subseteq V(G), |S| \leq k_r$$

$$\mathrm{dist}(A,B) \geq r \ \text{ in } \ G - S$$

Example. Class of grids:
$k_r := 0, \quad U_r(n) := \Omega(n/r^2)$

– every set $W \subseteq V(G)$ contains
some r-independent set W' of size $\Omega(n/r^2)$
split W' into two halves A, B



$\forall r\geq 1.\ \exists k_r\geq 1.\ \exists U_r : \mathbf{N}\to\mathbf{N}$ – unbounded function s.t.

$$\forall G\in C \quad \forall W\subseteq V(G) \quad \exists A,B\subseteq W,\ |A|=|B|\geq U_r(|W|), \quad \exists S\subseteq V(G), |S|\leq k_r$$

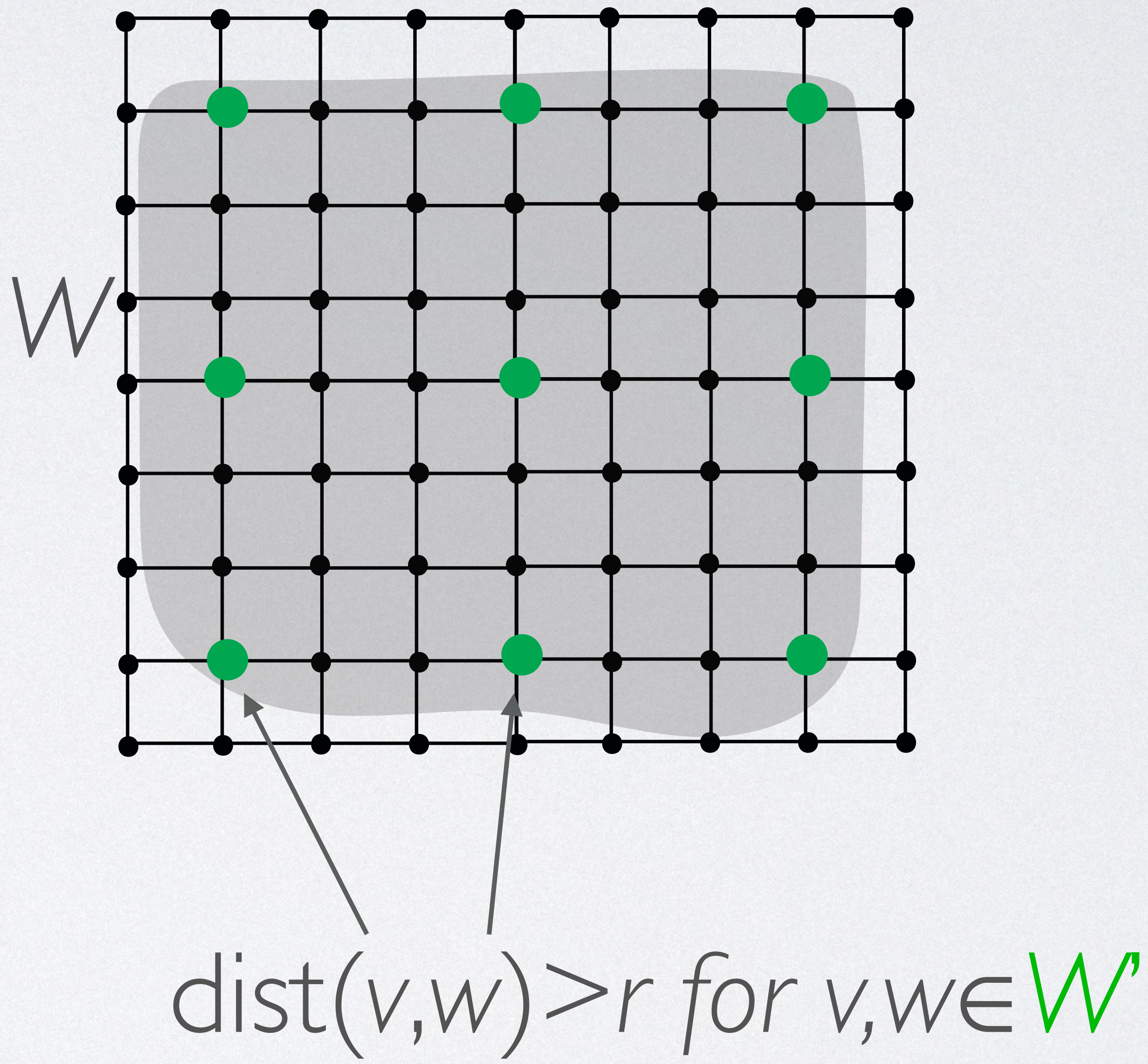
$$\text{dist}(A,B)\geq r \ \text{ in } \ G-S$$

Example. Class of grids:

$k_r := 0, \quad U_r(n):=\Omega(n/r^2)$

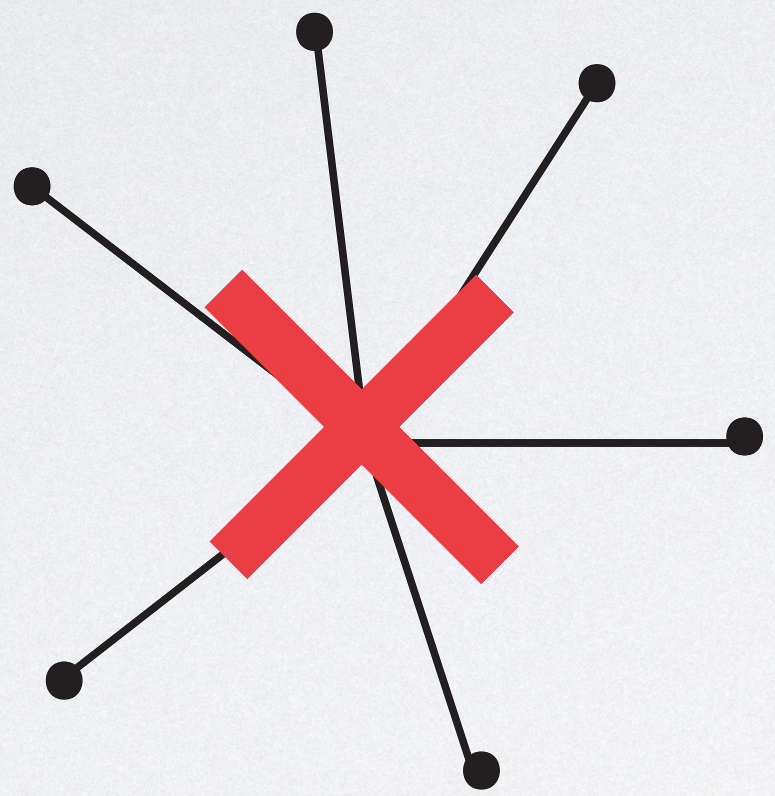
– every set $W\subseteq V(G)$ contains some r-independent set W' of size $\Omega(n/r^2)$

split W' into two halves A, B



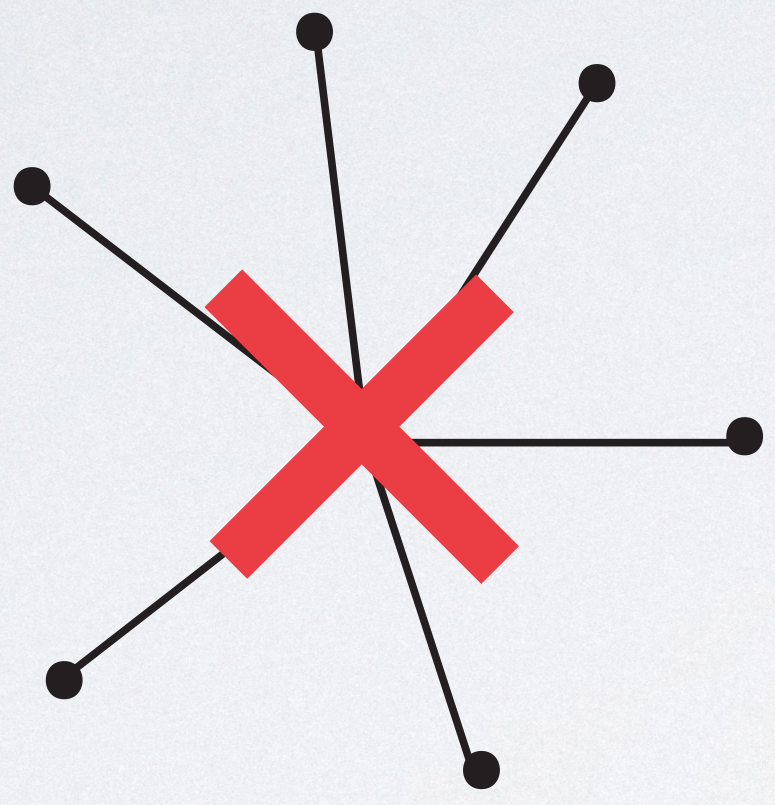
Theorem. C is deletion-breakable $\Leftrightarrow$ C is nowhere dense.

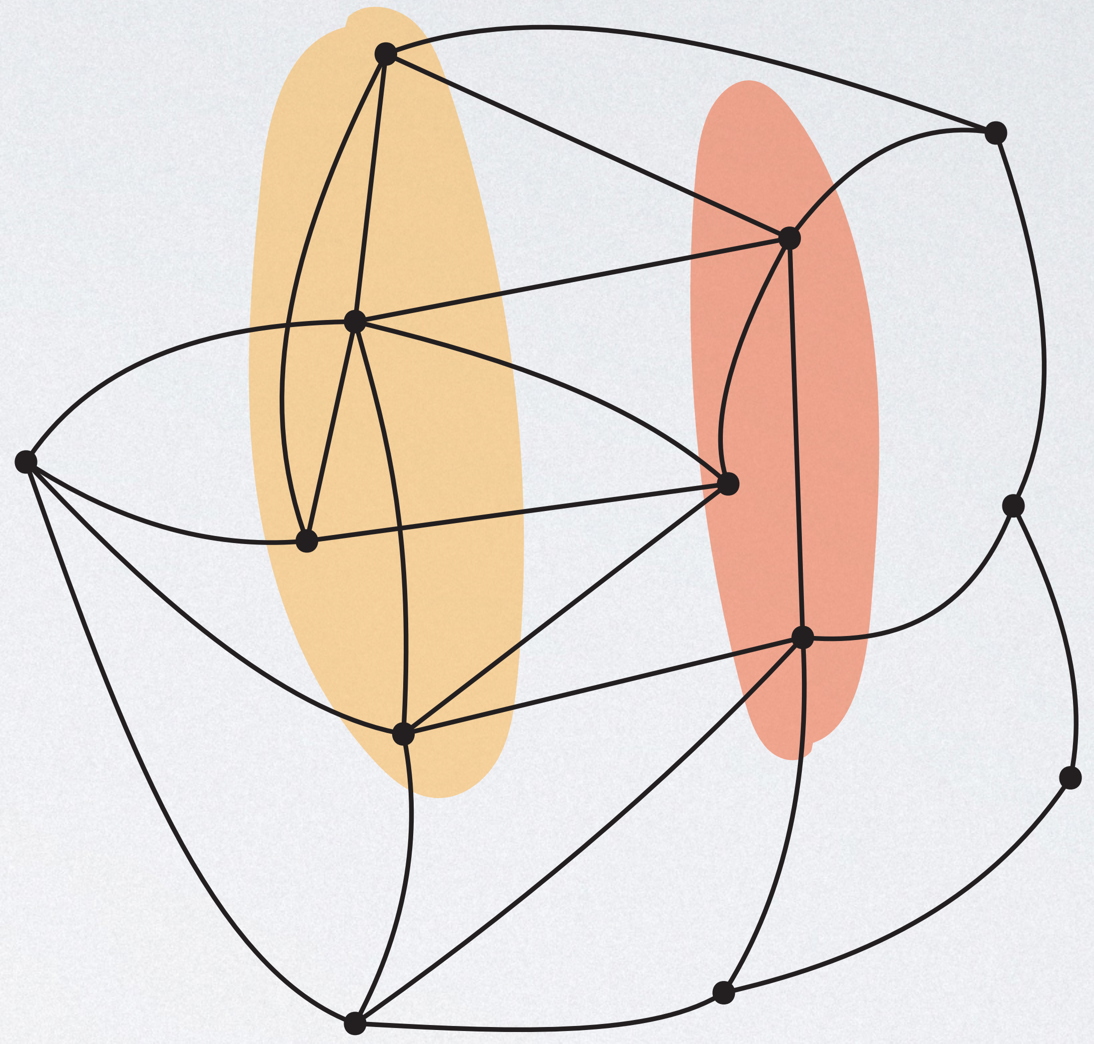
GOING DENSE



deleting a vertex

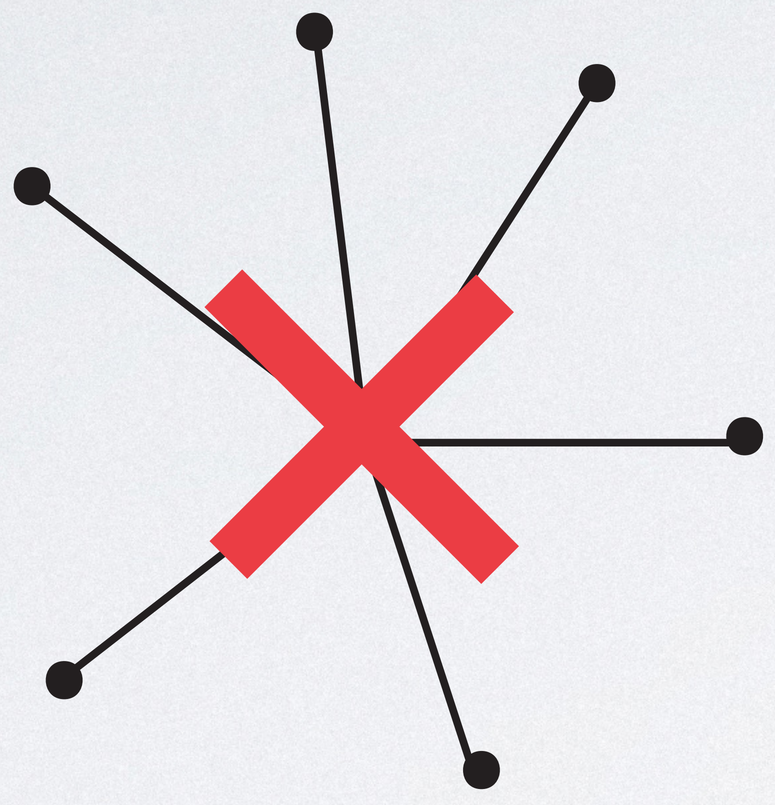
GOING DENSE

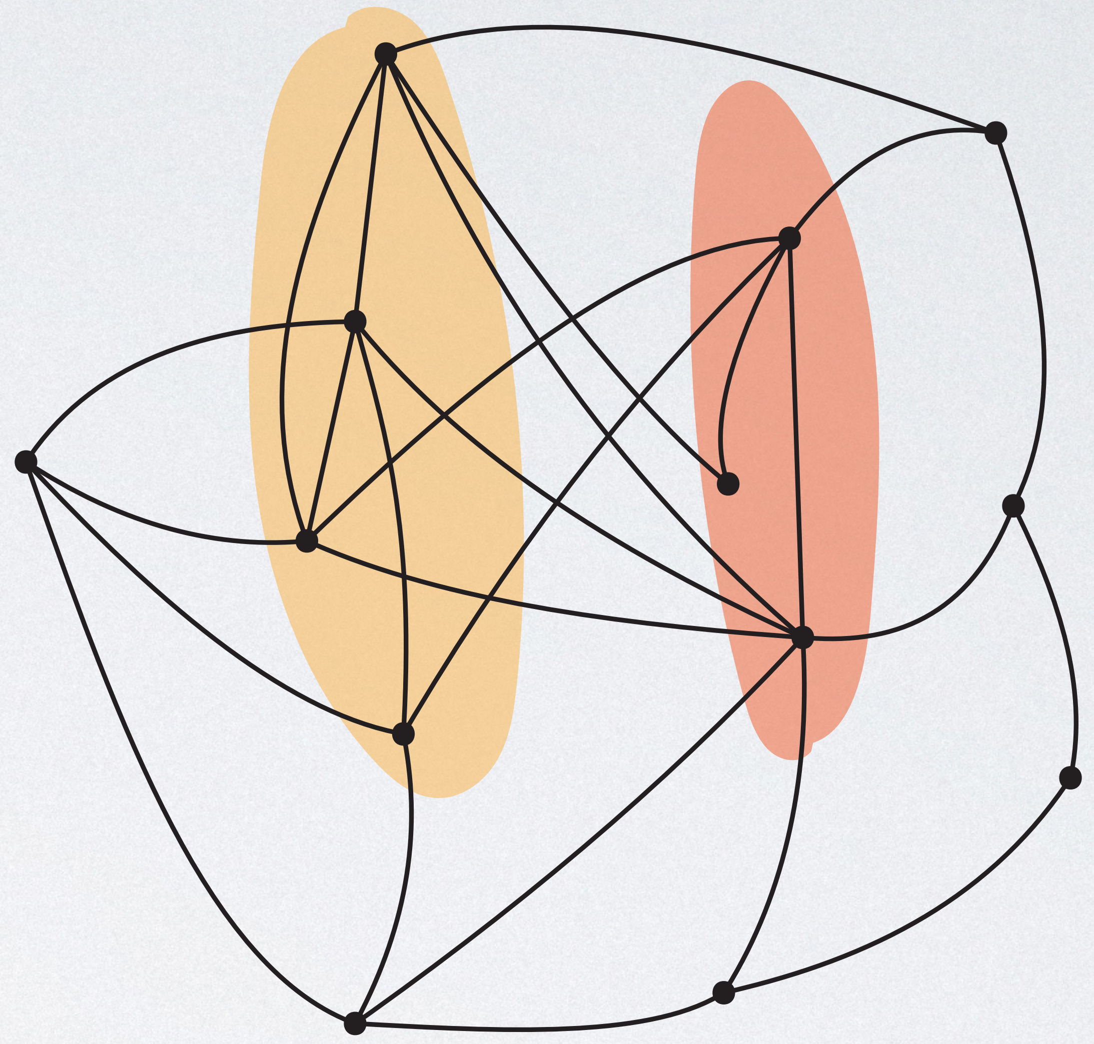




deleting a vertex → **flipping a pair of sets**

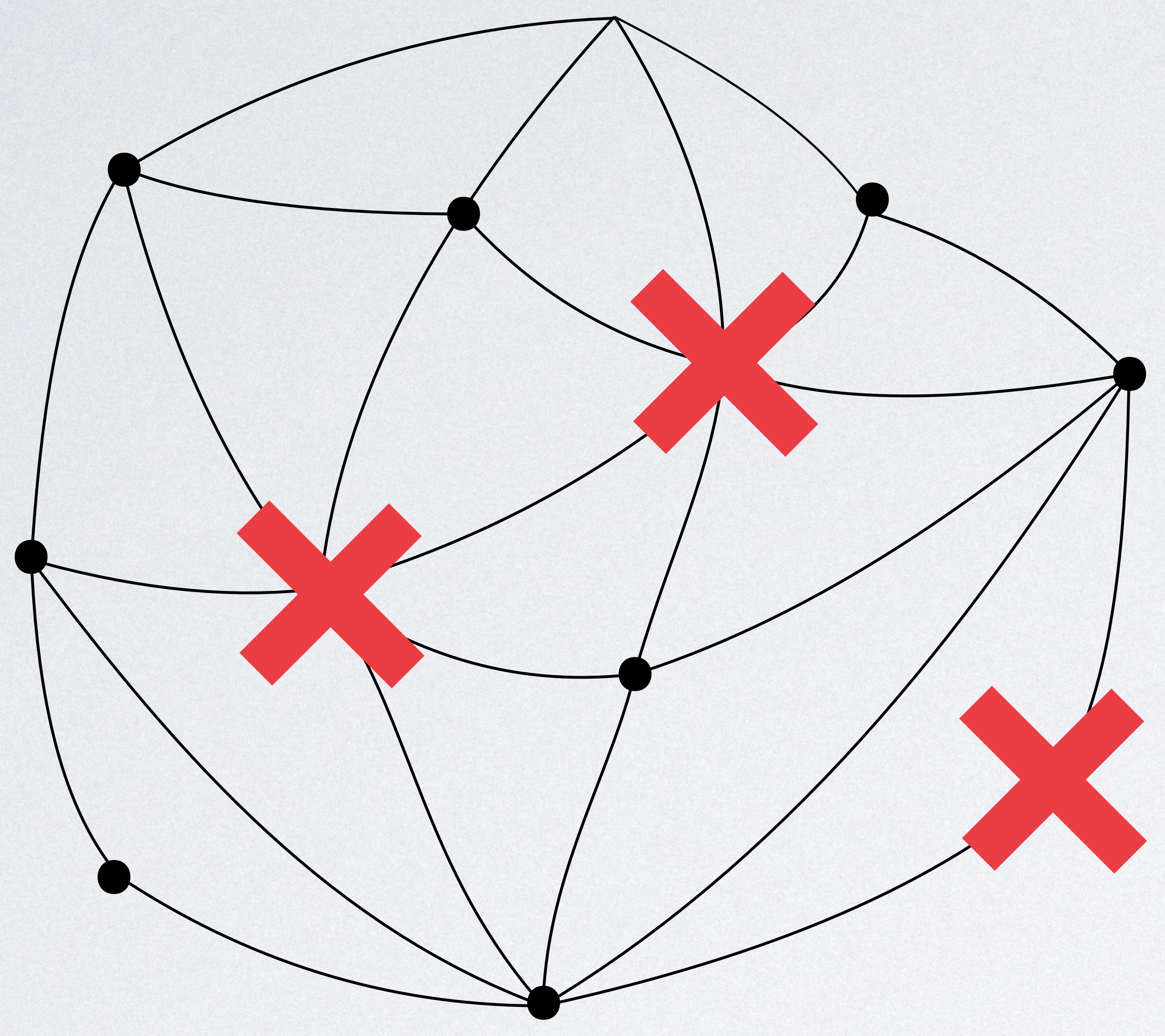
GOING DENSE





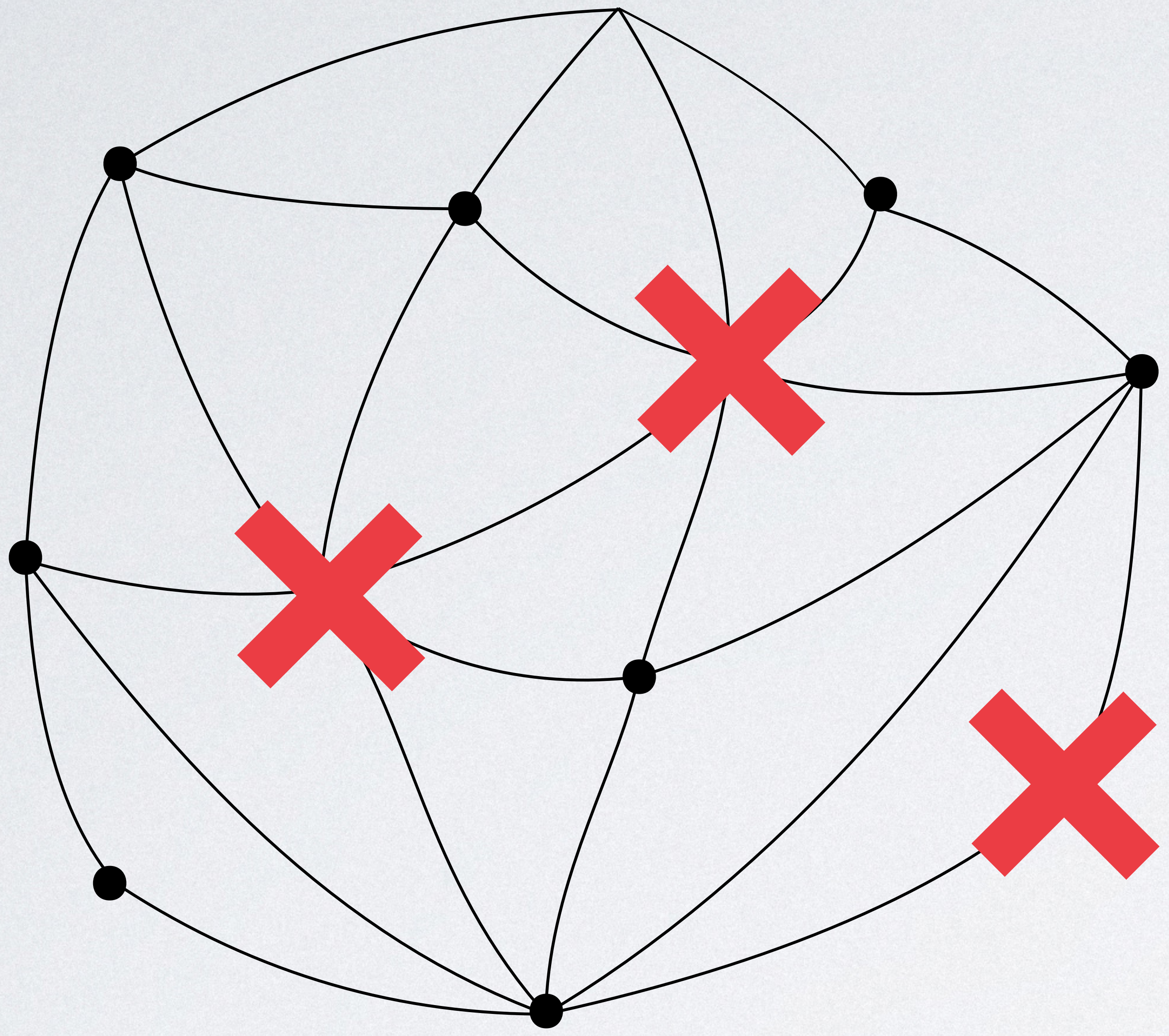
deleting a vertex → **flipping a pair of sets**

GOING DENSE

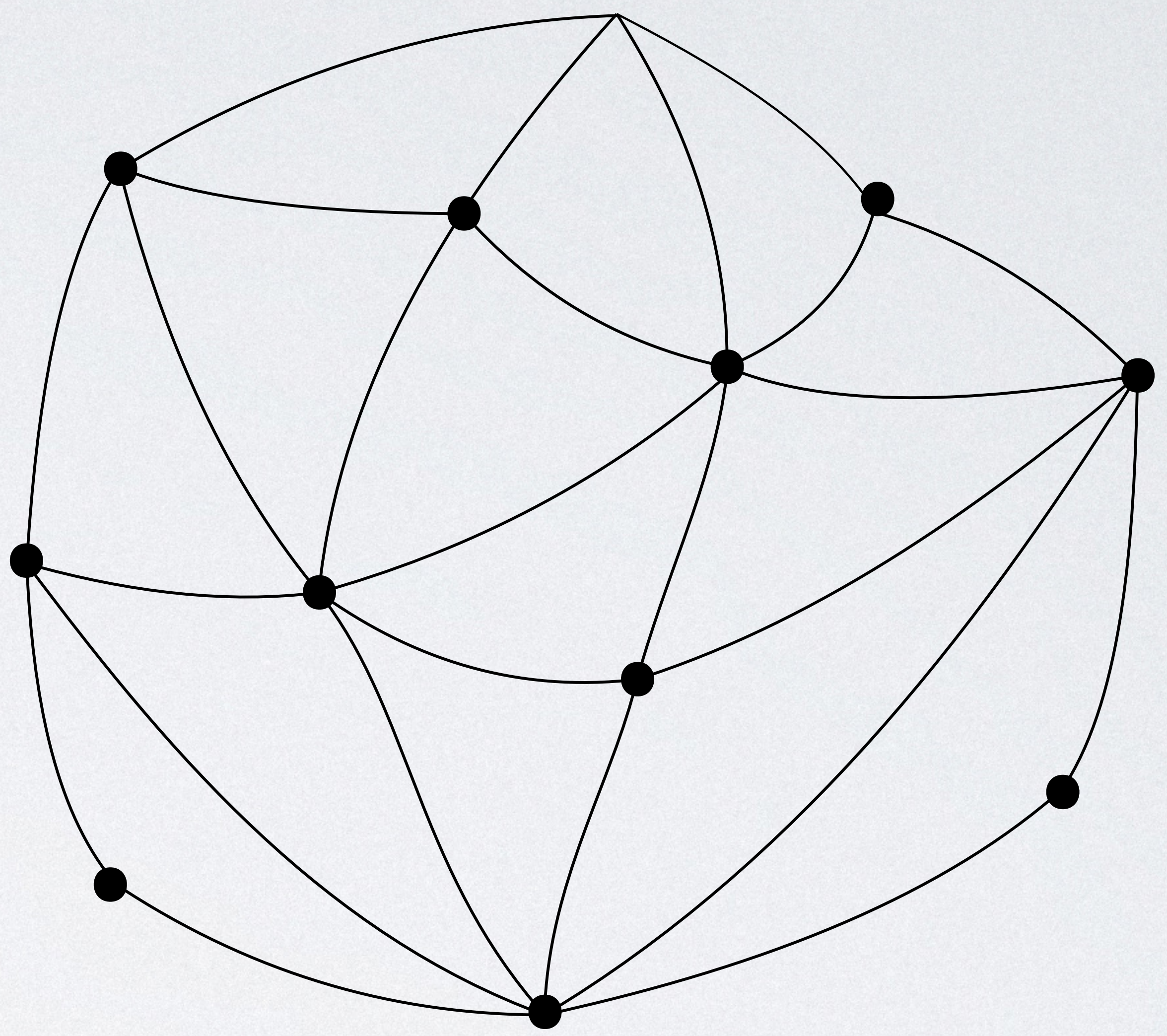


deleting *k* vertices

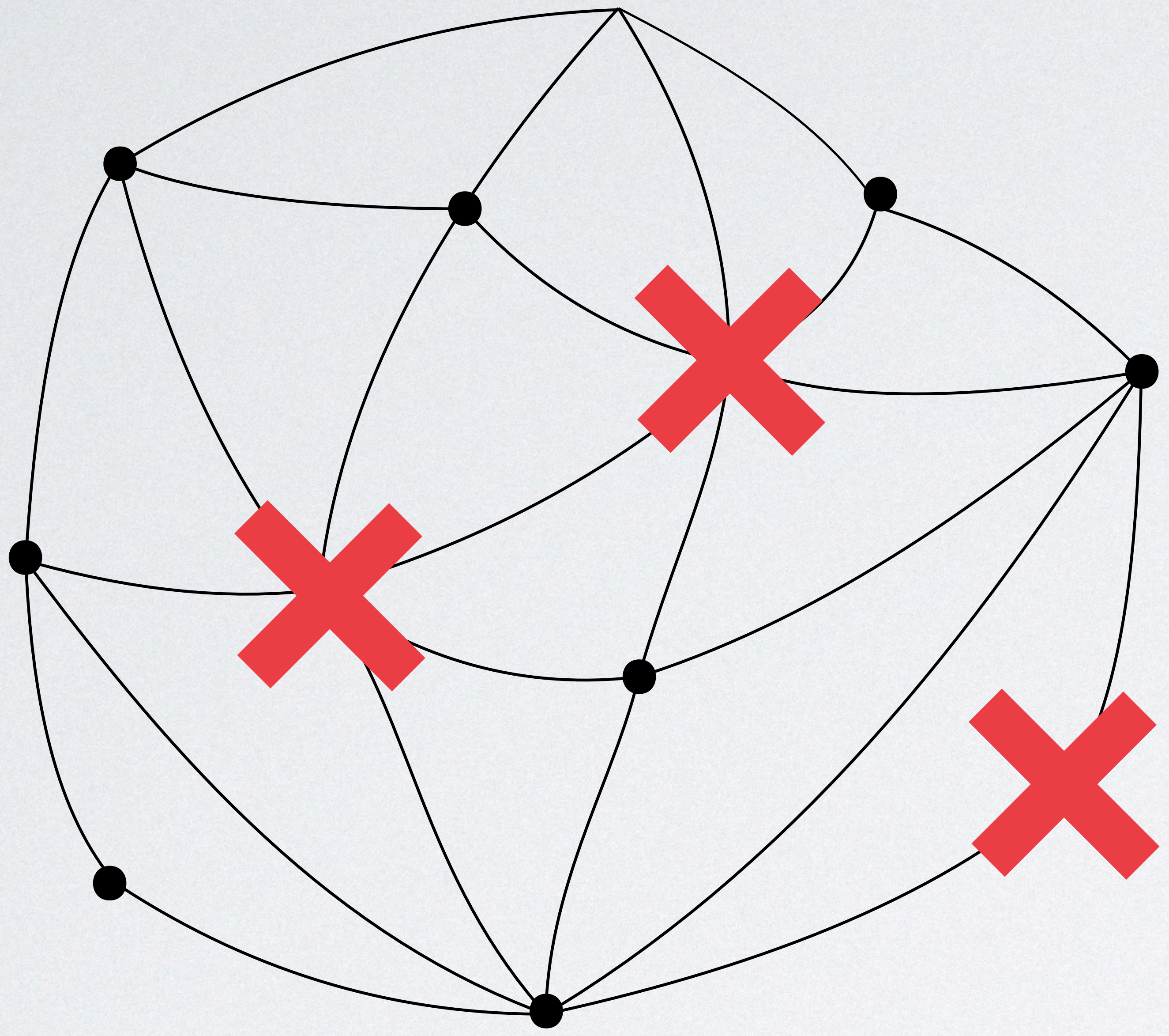
GOING DENSE

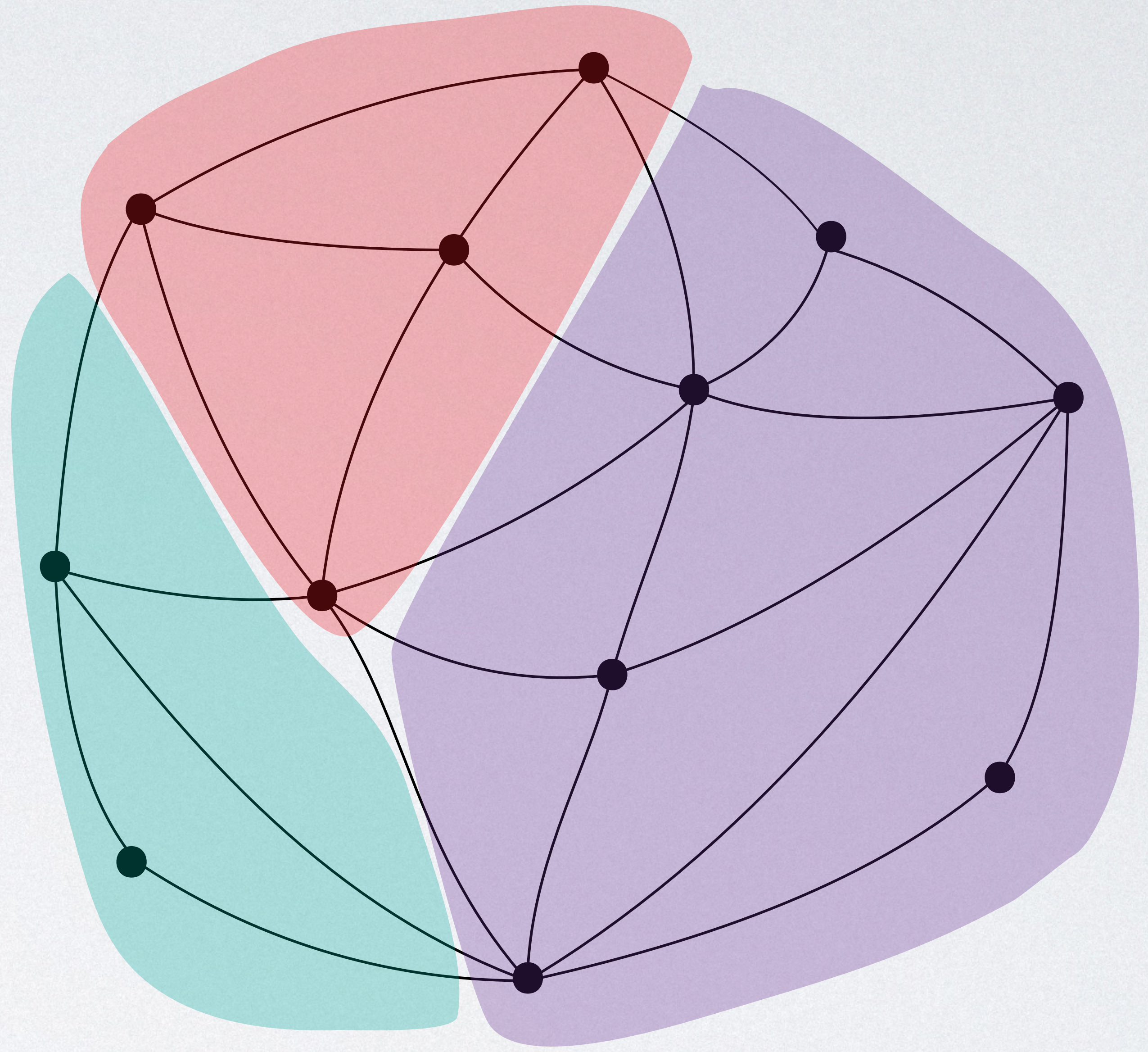


deleting *k* vertices →



GOING DENSE





deleting k vertices →

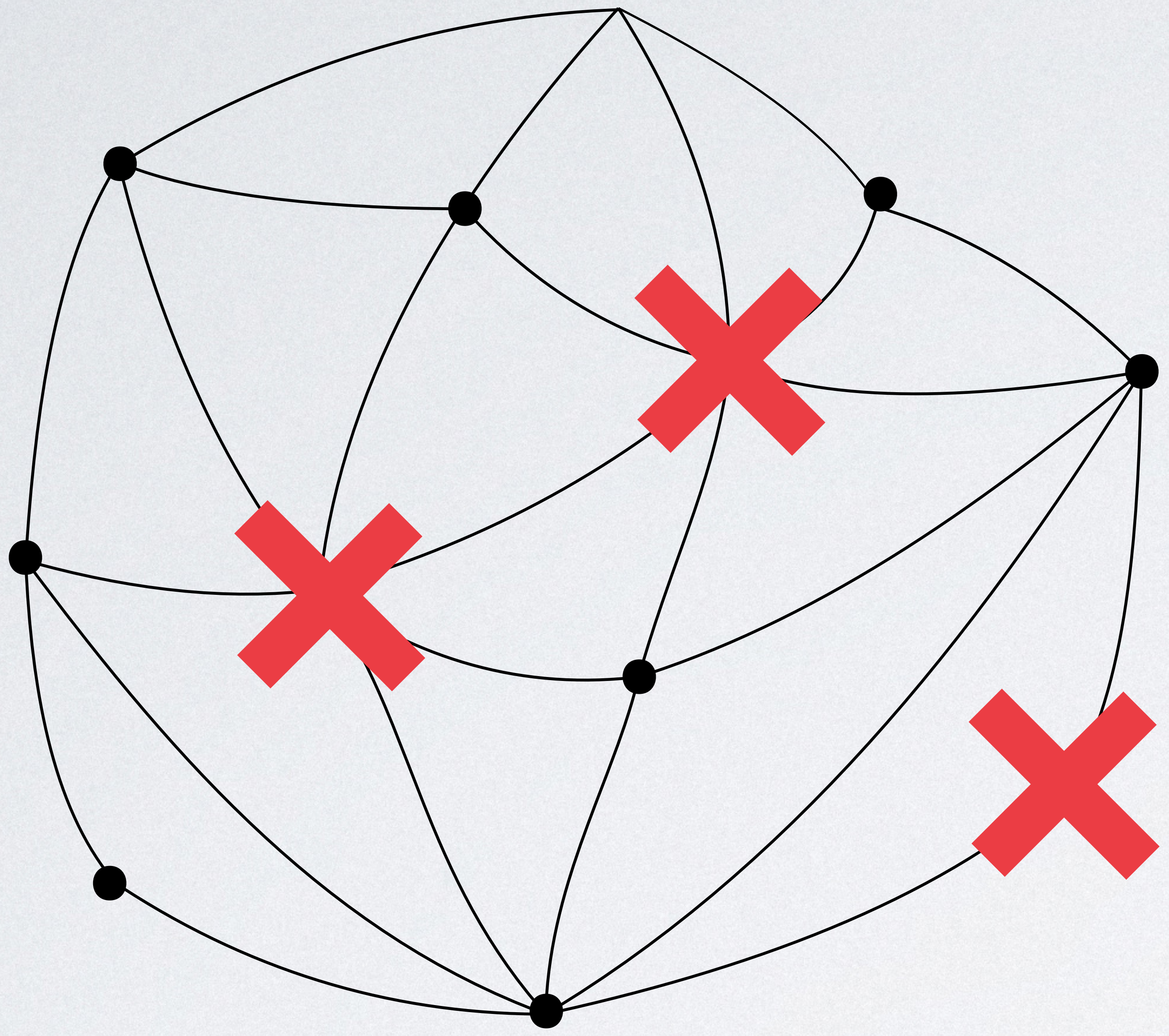
***k-flip* of G:**

partition $V(G)= A_1 \cup \ldots \cup A_k$

For each pair A_iA_j

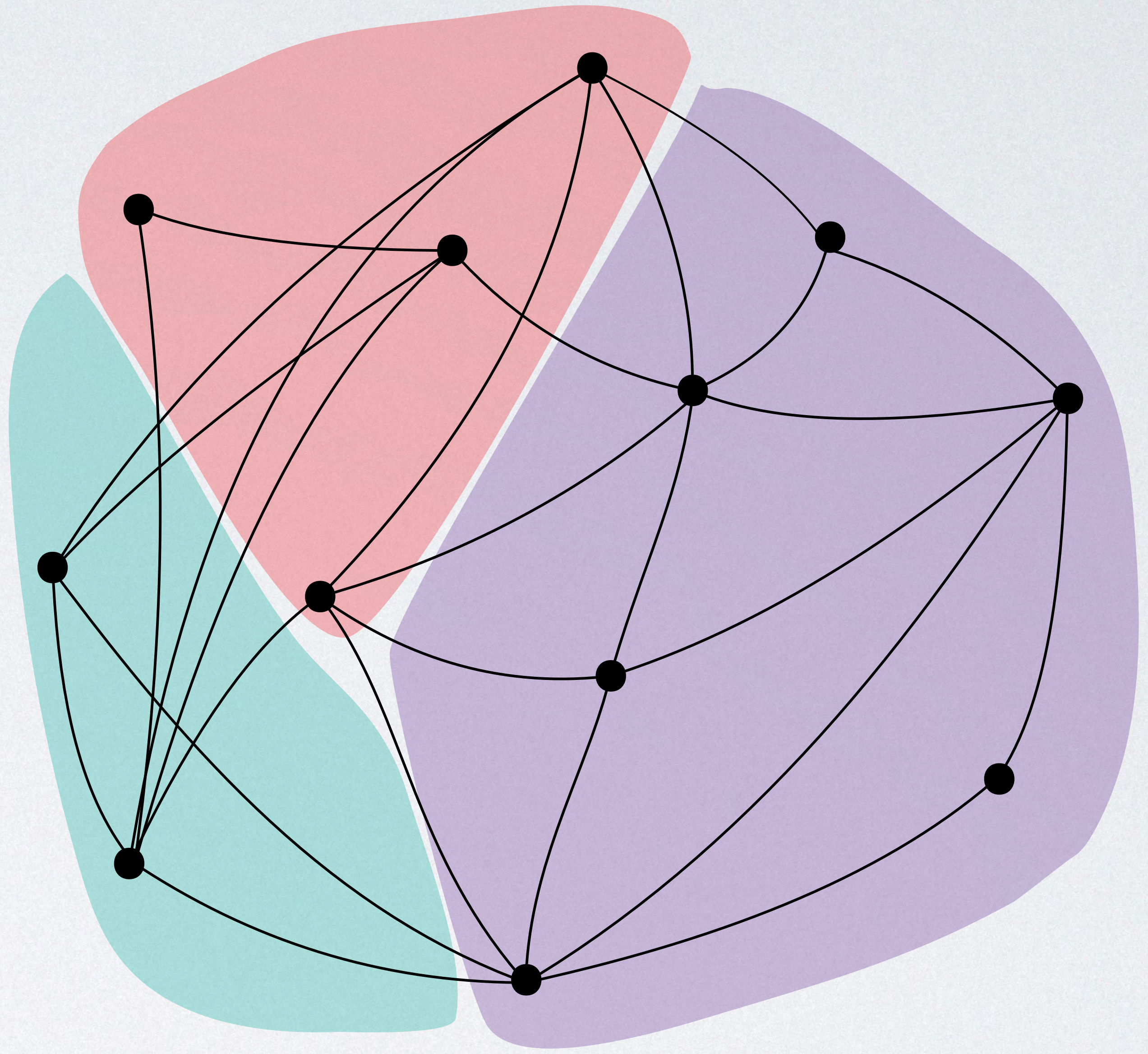
flip or not

GOING DENSE



deleting *k* vertices

→



***k-flip* of G:**

partition $V(G) = A_1 \cup \ldots \cup A_k$

For each pair A_iA_j

flip or not

∞-FLIP-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

∞-FLIP-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is ∞-*flip-breakable* if

∞-FLIP-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is *∞-flip-breakable* if

$\exists k \geq 1.\ \exists U : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

∞-FLIP-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is *∞-flip-breakable* if

$\exists k \geq 1.\ \exists U : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\forall G \in C.\ \forall W \subseteq V(G)\ \exists A,B \subseteq W,\ |A| = |B| \geq U(|W|)$, **$\exists k$-flip G' of G** s.t.

dist(A,B)$=\infty$ **in G'**

∞-FLIP-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is *∞-flip-breakable* if

$\exists k \geq 1.\ \exists U : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\forall G \in C.\ \forall W \subseteq V(G)\ \exists A, B \subseteq W,\ |A| = |B| \geq U(|W|)$, **$\exists k$-flip G' of G** s.t.

dist(A,B)=∞ **in *G'***

Theorem. C is ∞-flip-breakable ⇔ C has bounded cliquewidth.

FLIP-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC ’24]

FLIP-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is *flip-breakable* if

FLIP-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is *flip-breakable* if

$\forall r \geq 1$. $\exists k_r \geq 1$. $\exists U_r : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

FLIP-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is *flip-breakable* if

$\forall r \geq 1.\ \exists k_r \geq 1.\ \exists U_r : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\forall G \in C \quad \forall W \subseteq V(G) \quad \exists A,B \subseteq W,\ |A|=|B| \geq U_r(|W|), \quad \exists k_r$-flip G' of G:

dist(A,B)≥*r* in G'

FLIP-BREAKABILITY

[Dreier, Mählmann, **T.**, STOC '24]

Definition. A class C of graphs is *flip-breakable* if

$\forall r \geq 1.\ \exists k_r \geq 1.\ \exists U_r : \mathbf{N} \to \mathbf{N}$ – unbounded function s.t.

$\forall G \in C \quad \forall W \subseteq V(G) \quad \exists A,B \subseteq W,\ |A|=|B| \geq U_r(|W|), \quad \exists k_r$-flip G' of G:

dist(A,B)≥*r* in G'

Theorem. C is flip-breakable ⇔ C is monadically dependent.

	monotone: delete	*hereditary:* flip
radius ∞ MSO	bounded treewidth $\Leftrightarrow$ ∞-**deletion**-breakable	bounded cliquewidth $\Leftrightarrow$ ∞-**flip**-breakable
finite radii FO	nowhere dense $\Leftrightarrow$ **deletion**-breakable	monadically dependent $\Leftrightarrow$ **flip**-breakable

FORBIDDEN PATTERNS

[Dreier, Mählmann, **T.**, STOC '24]

FORBIDDEN PATTERNS

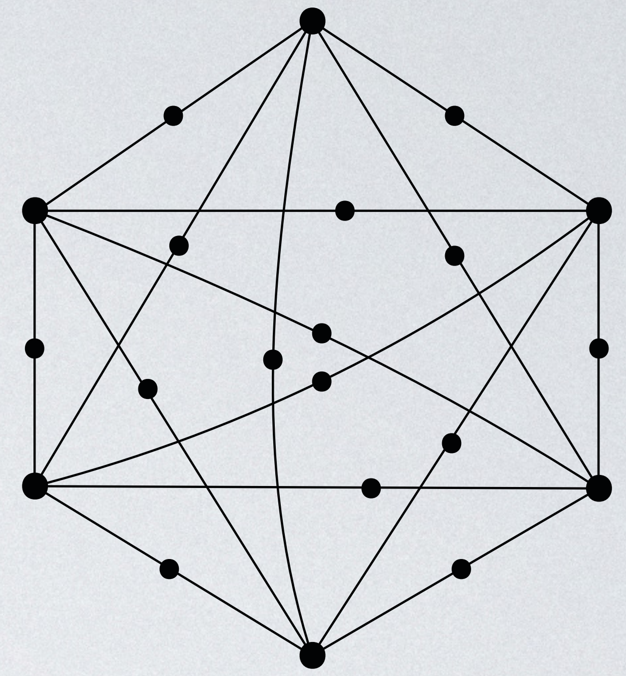
[Dreier, Mählmann, **T.**, STOC '24]

Theorem. C is not monadically dependent $\Leftrightarrow$

C contains arbitrarily large "bad patterns"

FORBIDDEN PATTERNS

[Dreier, Mählmann, **T.**, STOC '24]

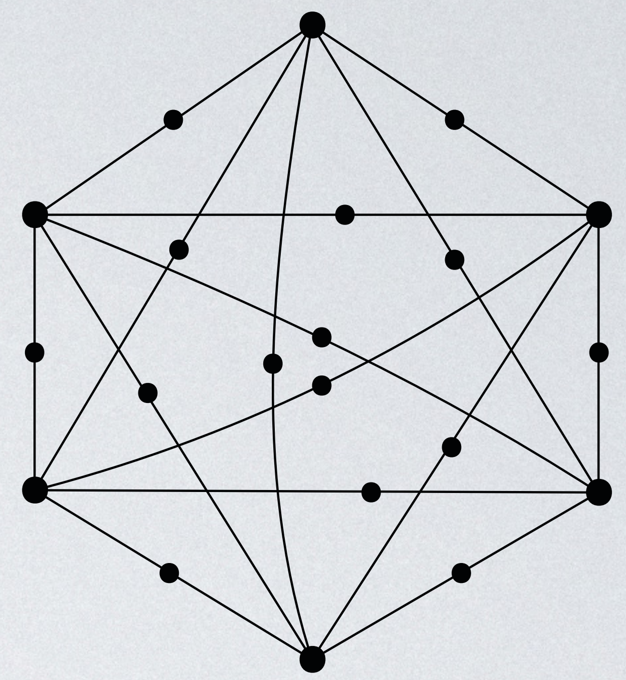


Theorem. C is not monadically dependent $\Leftrightarrow$

C contains arbitrarily large "bad patterns" $\approx$ r-subdivided cliques, up to flips

FORBIDDEN PATTERNS

[Dreier, Mählmann, **T.**, STOC '24]



Theorem. C is not monadically dependent ⇔

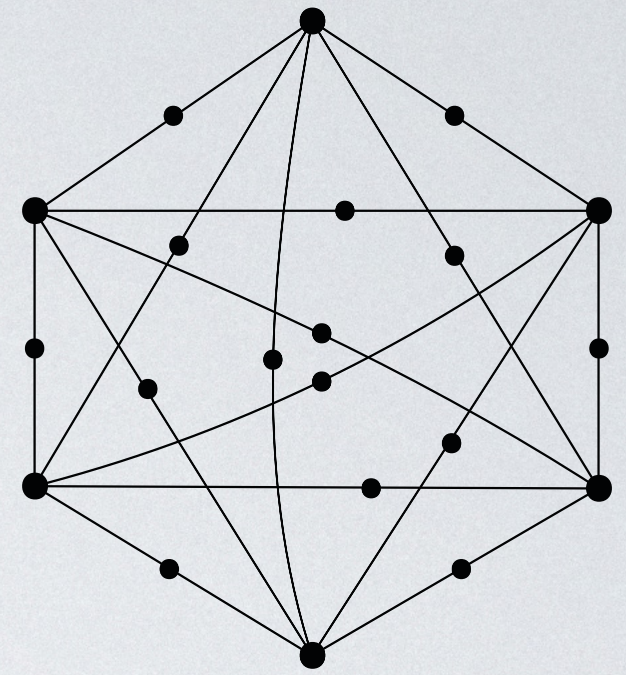
C contains arbitrarily large "bad patterns" ≈ r-subdivided cliques, up to flips

Corollary. For hereditary graph classes,
not monadically dependent ⇒* FO model checking is not fpt.

*assuming FPT≠AW[$*$]

FORBIDDEN PATTERNS

[Dreier, Mählmann, **T.**, STOC '24]



Theorem. C is not monadically dependent ⇔

C contains arbitrarily large "bad patterns" ≈ r-subdivided cliques, up to flips

Corollary. For hereditary graph classes,
not monadically dependent ⇒* FO model checking is not fpt.

*assuming FPT≠AW[$*$]

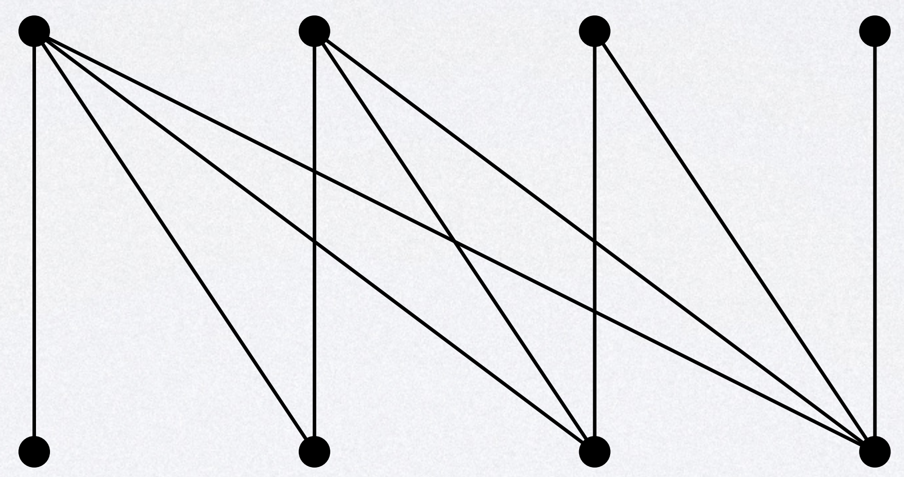
'⇒' implication in *"tractable* ⇔ *monadically dependent"* conjecture.

OUTLINE

1. The model checking problem
2. Sparsity: monotone case
3. Twin-width: ordered case
4. Monadic dependence
5. Flip-breakability
6. **Stability: orderless case**

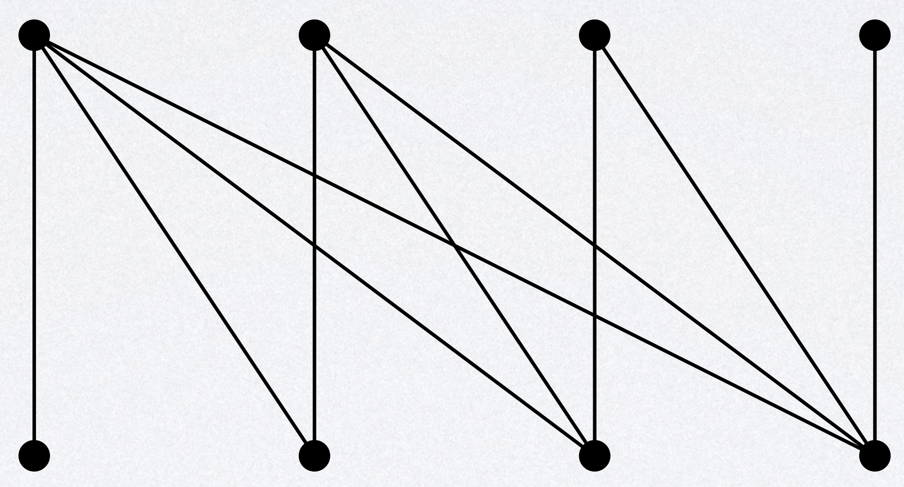
ORDERLESS CLASSES

Definition
A graph class is *orderless* if it avoids some half-graph as a semi-induced subgraph.



ORDERLESS CLASSES

Definition
A graph class is *orderless* if it avoids some half-graph as a semi-induced subgraph.



Example:

- all nowhere dense classes C, and
- all classes that can be transduced from C

ORDERLESS CLASSES

Theorem
For a hereditary, orderless class of graphs C:

C is monadically dependent $\Leftrightarrow$ C is *stable* $^{*}\Leftrightarrow$ C is tractable

ORDERLESS CLASSES

Theorem

For a hereditary, orderless class of graphs C:

C is monadically dependent $\Leftrightarrow$ C is *stable* $^{*}\Leftrightarrow$ C is tractable

key notion in model theory

ORDERLESS CLASSES

Theorem
For a hereditary, orderless class of graphs *C*:

C is monadically dependent $\Leftrightarrow$ *C* is *stable* $^*\Leftrightarrow$ *C* is tractable

key notion in model theory

Ingredients:

ORDERLESS CLASSES

Theorem

For a hereditary, orderless class of graphs C:

C is monadically dependent $\Leftrightarrow$ C is *stable* $^{*}\Leftrightarrow$ C is tractable

key notion in model theory

Ingredients:

1. existence of a treelike decomposition [Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski, **T.**, 23]

ORDERLESS CLASSES

Theorem
For a hereditary, orderless class of graphs C:

C is monadically dependent $\Leftrightarrow$ C is *stable* $^*\Leftrightarrow$ C is tractable

key notion in model theory

Ingredients:

1. existence of a treelike decomposition [Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski, **T.**, 23]
2. efficient computation of decomposition [Dreier, Eleftheriadis, Mählmann, McCarty, Pilipczuk, **T**. '24+]

ORDERLESS CLASSES

Theorem
For a hereditary, orderless class of graphs C:

C is monadically dependent $\Leftrightarrow$ C is *stable* $^{*}\Leftrightarrow$ C is tractable

key notion in model theory

Ingredients:

1. existence of a treelike decomposition [Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski, **T.**, 23]
2. efficient computation of decomposition [Dreier, Eleftheriadis, Mählmann, McCarty, Pilipczuk, **T**. '24+]
3. dynamic algorithm [Dreier, Mählmann, Siebertz '23]

FLIPPER GAME

a treelike decomposition

[Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski, **T.**, 23]

FLIPPER GAME

a treelike decomposition

[Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski, **T.**, 23]

Flipper Game of radius $r \geq 1$ between two players – Flipper and Keeper

FLIPPER GAME

a treelike decomposition

[Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski, **T.**, 23]

Flipper Game of radius $r \geq 1$ between two players – Flipper and Keeper

In each round:

FLIPPER GAME

a treelike decomposition

[Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski, **T.**, 23]

Flipper Game of radius $r \geq 1$ between two players – Flipper and Keeper

In each round:

1. Flipper flips a pair of sets in the current graph

FLIPPER GAME

a treelike decomposition

[Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski, **T.**, 23]

Flipper Game of radius $r \geq 1$ between two players – Flipper and Keeper

In each round:

1. Flipper flips a pair of sets in the current graph
2. Keeper restricts the current graph to some ball of radius *r*

FLIPPER GAME

a treelike decomposition

[Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski, **T.**, 23]

Flipper Game of radius $r \geq 1$ between two players – Flipper and Keeper

In each round:

1. Flipper flips a pair of sets in the current graph
2. Keeper restricts the current graph to some ball of radius *r*

Flipper wins when one vertex remains

FLIPPER GAME

a treelike decomposition

[Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski, **T.**, 23]

Flipper Game of radius $r \geq 1$ between two players – Flipper and Keeper

In each round:

1. Flipper flips a pair of sets in the current graph
2. Keeper restricts the current graph to some ball of radius r

Flipper wins when one vertex remains

Theorem. A class C is monadically dependent and orderless ⇔

$\forall r \geq 1 \;\; \exists t \geq 1$: Flipper wins in $\leq t$ rounds in every graph $G \in C$.

Theorem. A class C is monadically dependent and orderless $\Leftrightarrow$

$\forall r \geq 1 \;\; \exists t \geq 1$: Flipper wins in $\leq t$ rounds on every graph $G \in C$.

Theorem. A class C is monadically dependent and orderless ⇔

$\forall r \geq 1 \quad \exists t \geq 1$: Flipper wins in $\leq t$ rounds on every graph $G \in C$.

A winning strategy of Flipper in G is a tree T of depth t and branching $n := V(G)$
(branching corresponds to balls of radius r)
→ T has size $\leq n^t$

Theorem. A class C is monadically dependent and orderless $\Leftrightarrow$

$\forall r \geq 1 \quad \exists t \geq 1$: Flipper wins in $\leq t$ rounds on every graph $G \in C$.

A winning strategy of Flipper in G is a tree T of depth t and branching $n := V(G)$
(branching corresponds to balls of radius r)
→ T has size $\leq n^t$

[Dreier, Eleftheriadis, Mählmann, McCarty, Pilipczuk, **T**. '24+]:

Efficient "compression" to tree of size $\leq O_{\varepsilon,C}(n^{1+\varepsilon})$, for any fixed $\varepsilon > 0$.

Theorem. A class C is monadically dependent and orderless $\Leftrightarrow$

$\forall r \geq 1 \quad \exists t \geq 1$: Flipper wins in $\leq t$ rounds on every graph $G \in C$.

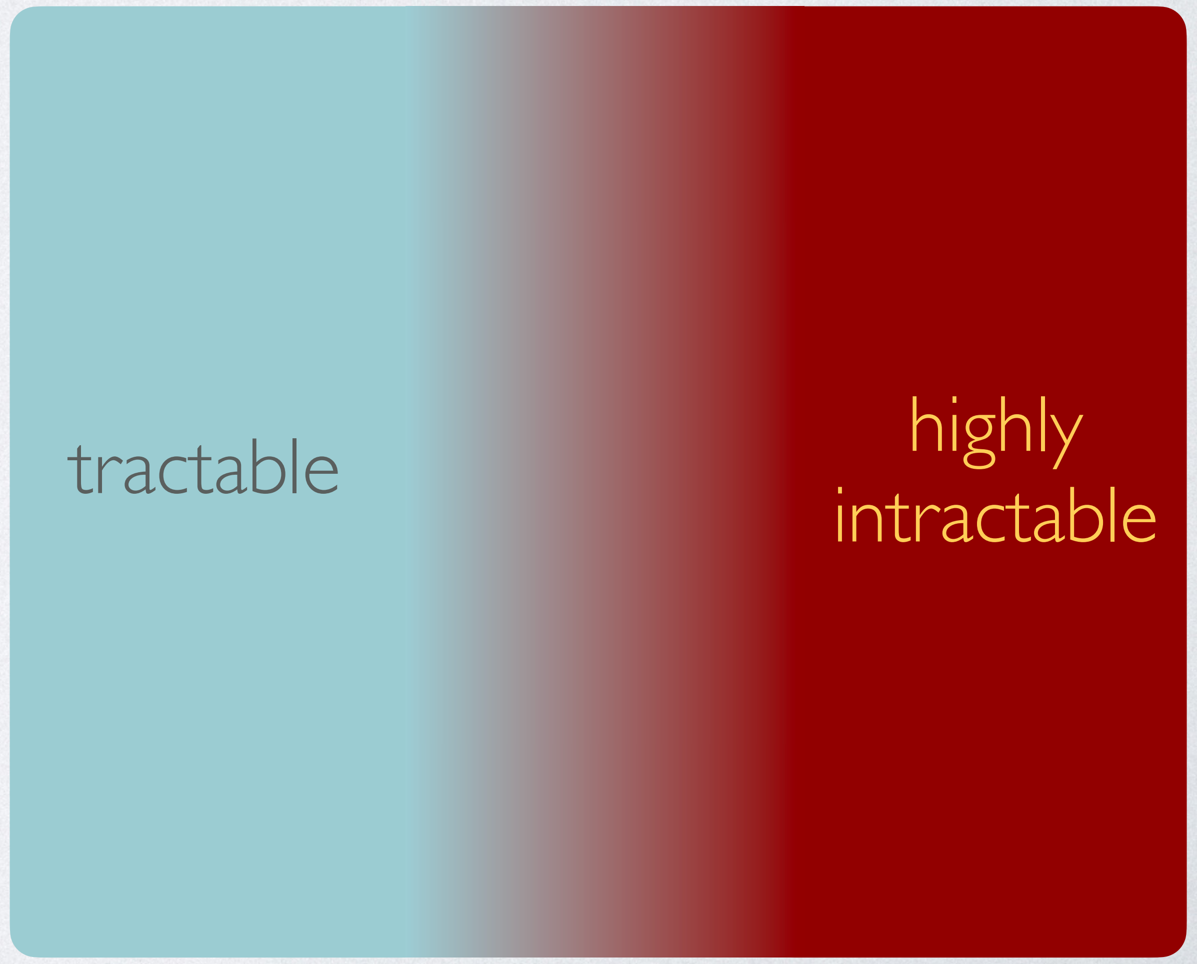
A winning strategy of Flipper in G is a tree T of depth t and branching $n := V(G)$
(branching corresponds to balls of radius r)
$\rightarrow$ T has size $\leq n^t$

[Dreier, Eleftheriadis, Mählmann, McCarty, Pilipczuk, **T**. '24+]:

Efficient "compression" to tree of size $\leq O_{\varepsilon,C}(n^{1+\varepsilon})$, for any fixed $\varepsilon > 0$.

Ingredients: stability theory, sparsity theory, VC theory, geometric range queries

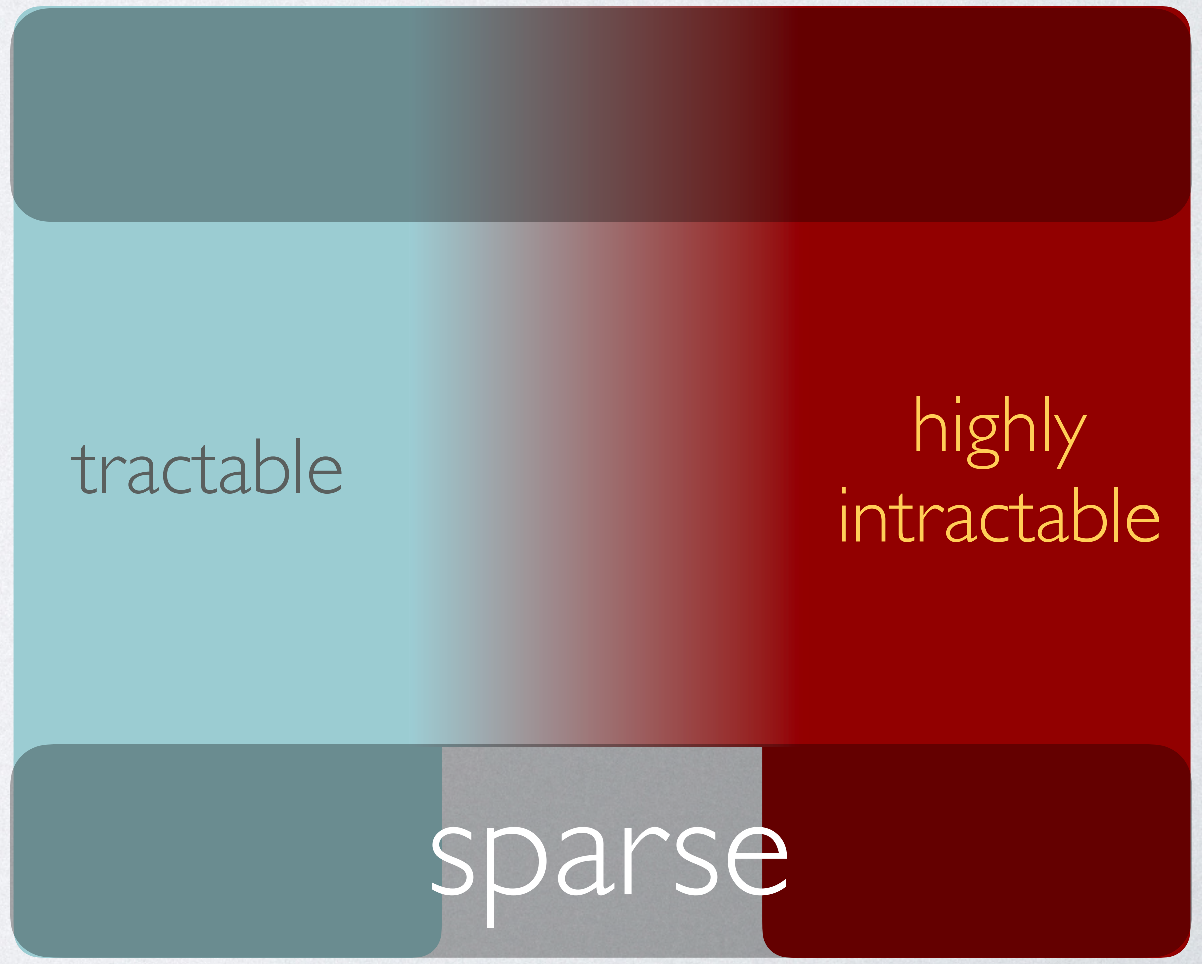
SUMMARY



SUMMARY

For **monotone** graph classes:

monadically dependent ⇔ nowhere dense *⇔ tractable



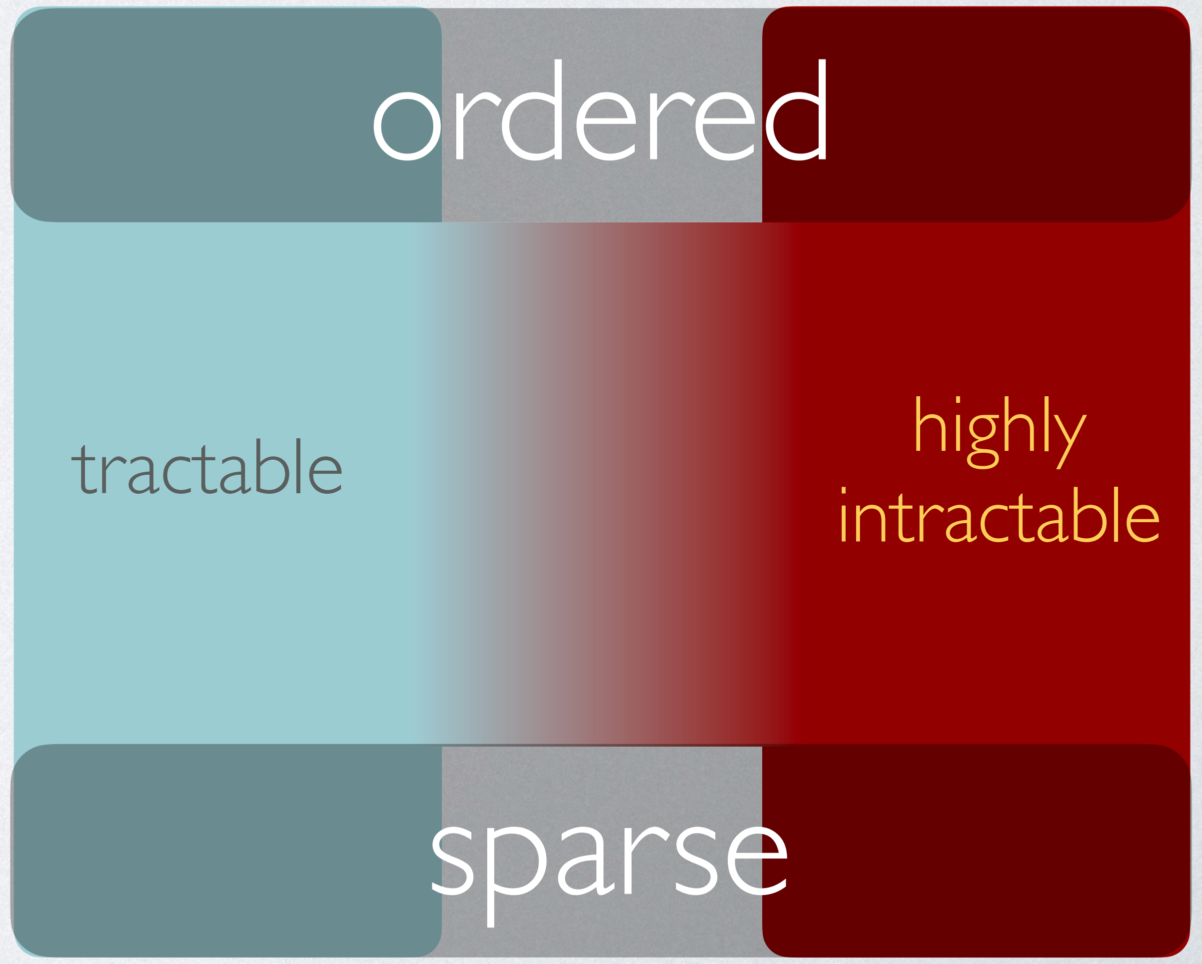
SUMMARY

For **monotone** graph classes:

monadically dependent $\Leftrightarrow$ nowhere dense $^{*}\Leftrightarrow$ tractable

For hereditary classes of **ordered graphs**:

monadically dependent $\Leftrightarrow$ bounded twin-width $^{*}\Leftrightarrow$ tractable



SUMMARY

For **monotone** graph classes:

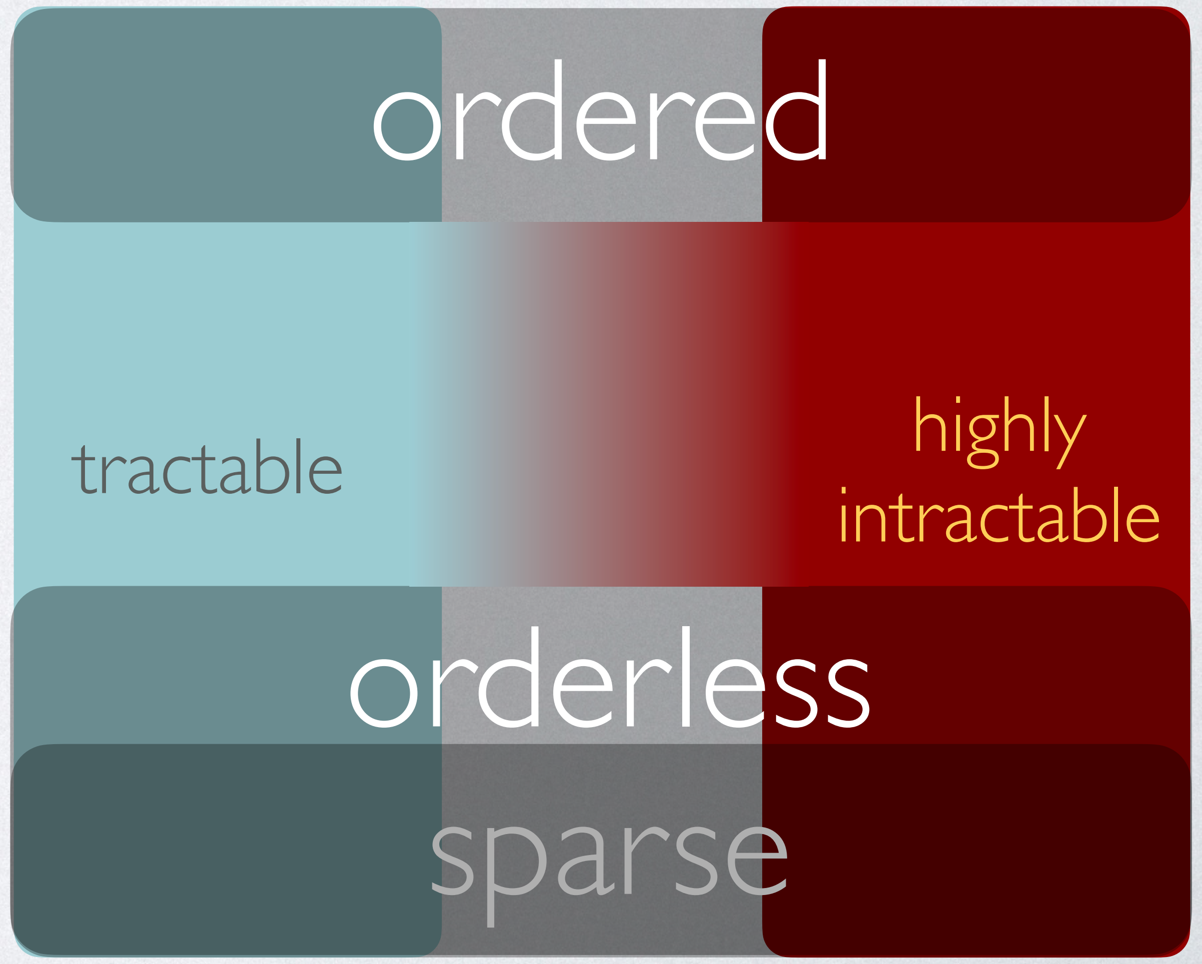
monadically dependent ⇔ nowhere dense *⇔ tractable

For hereditary classes of **ordered graphs**:

monadically dependent ⇔ bounded twin-width *⇔ tractable

For hereditary, **orderless** graph classes:

monadically dependent ⇔ stable *⇔ tractable



SUMMARY

For **monotone** graph classes:

monadically dependent $\Leftrightarrow$ nowhere dense $^*\Leftrightarrow$ tractable

For hereditary classes of **ordered graphs**:

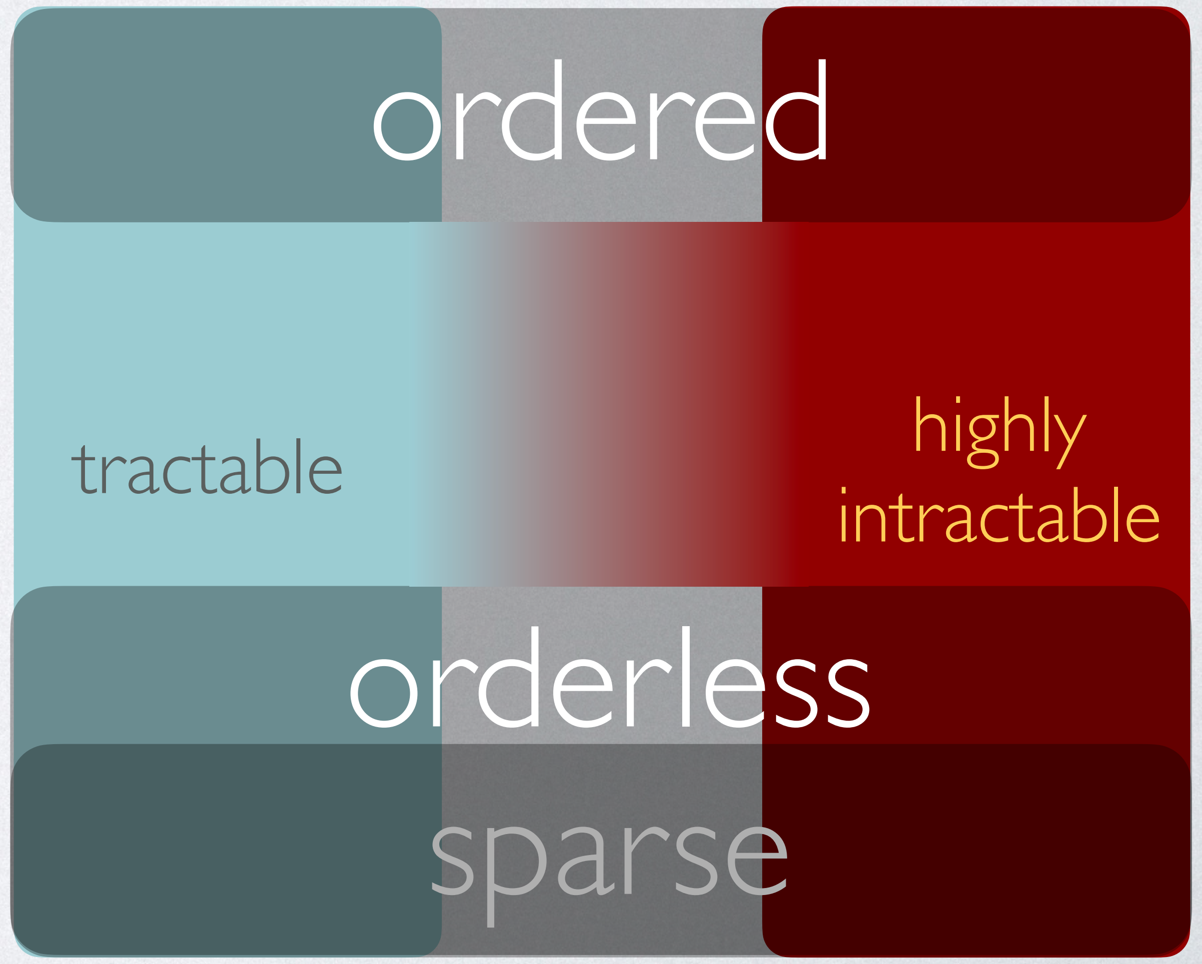
monadically dependent $\Leftrightarrow$ bounded twin-width $^*\Leftrightarrow$ tractable

For hereditary, **orderless** graph classes:

monadically dependent $\Leftrightarrow$ stable $^*\Leftrightarrow$ tractable

For **all** hereditary classes:

monadically dependent $\Leftrightarrow$ flip-breakable $^*\Leftrightarrow$? tractable



SUMMARY

For **monotone** graph classes:

monadically dependent ⇔ nowhere dense *⇔ tractable

For hereditary classes of **ordered graphs**:

monadically dependent ⇔ bounded twin-width *⇔ tractable

For hereditary, **orderless** graph classes:

monadically dependent ⇔ stable *⇔ tractable

For **all** hereditary classes:

monadically dependent ⇔ flip-breakable *⇔ ? tractable