Logic through the lens of structural graph theory

Michał Pilipczuk

University of Warsaw

Finite and Algorithmic Model Theory

Les Houches, France May 26th, 2025

Supported by ERC project BOBR, ref. no. 948057.

(Ob)structural graph theory

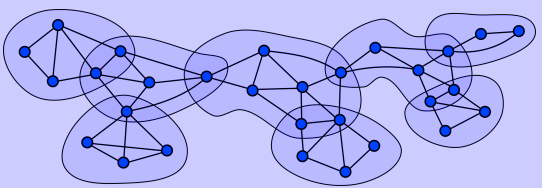
(Ob)structural graph theory

Structure

Decompositions exposing various properties:

— **trees** or **covers**,

— usable by **algorithms**.



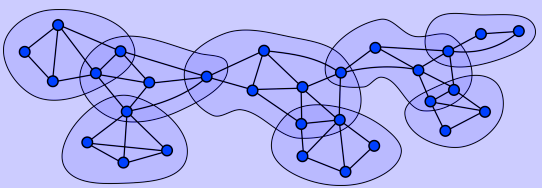
(Ob)structural graph theory

Structure

Decompositions exposing various properties:

— **trees** or **covers**,

— usable by **algorithms**.

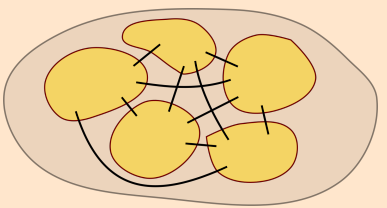


Obstruction

Object embedded in the considered graph:

— no **decomposition**,

— gives **hardness**.



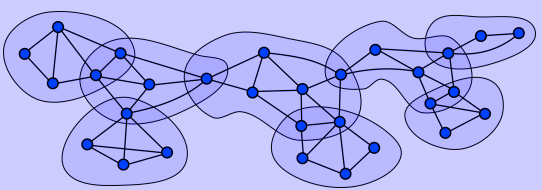
(Ob)structural graph theory

Structure

Decompositions exposing various properties:

— **trees** or **covers**,

— usable by **algorithms**.

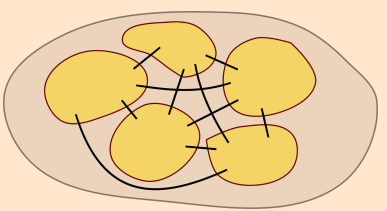


Obstruction

Object embedded in the considered graph:

— no **decomposition**,

— gives **hardness**.



Theorem Template

If G contains no **obstruction** $\mathcal{O}$, then G has a **decomposition** $\mathcal{D}$.

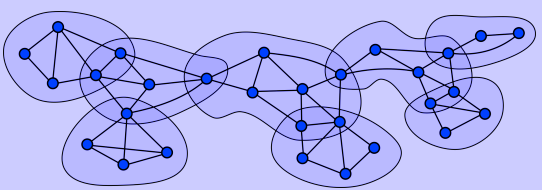
(Ob)structural graph theory

Structure

Decompositions exposing various properties:

— **trees** or **covers**,

— usable by **algorithms**.

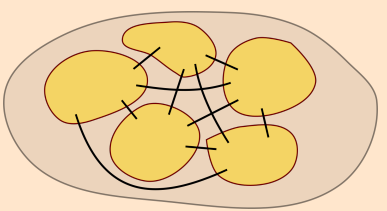


Obstruction

Object embedded in the considered graph:

— no **decomposition**,

— gives **hardness**.



Theorem Template

If G contains no **obstruction** $\mathcal{O}$, then G has a **decomposition** $\mathcal{D}$.

Key: notion of **embedding**.

Embeddings

Embedding	View	Theory

Embeddings

Embedding	View	Theory
(top) minors	topological space	graph minors theory

Embeddings

Embedding	View	Theory
(top) minors	topological space	graph minors theory
induced subgraphs	combinatorial structure	mess

Embeddings

Embedding	View	Theory
(top) minors	topological space	graph minors theory
induced subgraphs	combinatorial structure	mess
induced minors fat minors	metric space	coarse theory

Embeddings

Embedding	View	Theory
(top) minors	topological space	graph minors theory
induced subgraphs	combinatorial structure	mess
induced minors fat minors	metric space	coarse theory
vertex-minors pivot-minors	matrix over $\mathbb{F}_2$	vertex-minors theory

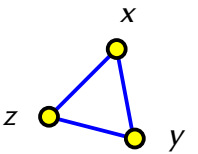
Embeddings

Embedding	View	Theory
(top) minors	topological space	graph minors theory
induced subgraphs	combinatorial structure	mess
induced minors fat minors	metric space	coarse theory
vertex-minors pivot-minors	matrix over $\mathbb{F}_2$	vertex-minors theory
???	logical structure	???

FO and MSO on graphs

FO and MSO on graphs

x, y, z form a triangle; the graph is triangle-free:

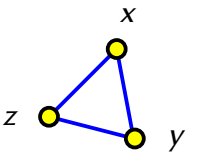


$$\text{triangle}(x, y, z) = \text{adj}(x, y) \wedge \text{adj}(y, z) \wedge \text{adj}(z, x).$$

$$\text{triangleFree} = \forall x.\forall y.\forall z.\ \neg\text{triangle}(x, y, z).$$

FO and MSO on graphs

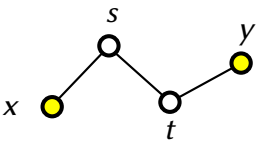
x, y, z form a triangle; the graph is triangle-free:



$$\text{triangle}(x, y, z) = \text{adj}(x, y) \wedge \text{adj}(y, z) \wedge \text{adj}(z, x).$$

$$\text{triangleFree} = \forall x.\forall y.\forall z.\ \neg\text{triangle}(x, y, z).$$

Vertices x and y are at distance $\leqslant 3$:

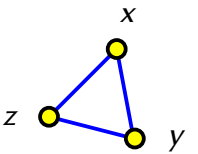


$$\text{dist}_{\leqslant 1}(x, y) = (x = y) \vee \text{adj}(x, y)$$

$$\text{dist}_{\leqslant 3}(x, y) = \exists s.\exists t.\ \text{dist}_{\leqslant 1}(x, s) \wedge \text{dist}_{\leqslant 1}(s, t) \wedge \text{dist}_{\leqslant 1}(t, y).$$

FO and MSO on graphs

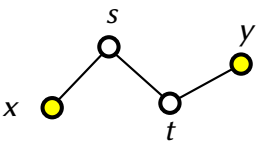
x, y, z form a triangle; the graph is triangle-free:



$$\mathrm{triangle}(x, y, z) = \mathrm{adj}(x, y) \wedge \mathrm{adj}(y, z) \wedge \mathrm{adj}(z, x).$$

$$\mathrm{triangleFree} = \forall x. \forall y. \forall z.\ \neg \mathrm{triangle}(x, y, z).$$

Vertices x and y are at distance $\leqslant 3$:



$$\mathrm{dist}_{\leqslant 1}(x, y) = (x = y) \vee \mathrm{adj}(x, y)$$

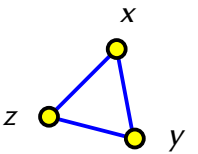
$$\mathrm{dist}_{\leqslant 3}(x, y) = \exists s. \exists t.\ \mathrm{dist}_{\leqslant 1}(x, s) \wedge \mathrm{dist}_{\leqslant 1}(s, t) \wedge \mathrm{dist}_{\leqslant 1}(t, y).$$

Every red vertex is at distance $\leqslant 3$ from a blue vertex:

$$\mathrm{blueDominatesRed} = \forall x.\ [\mathrm{red}(x) \Rightarrow \exists y.\ \mathrm{blue}(y) \wedge \mathrm{dist}_{\leqslant 3}(x, y)]\,.$$

FO and MSO on graphs

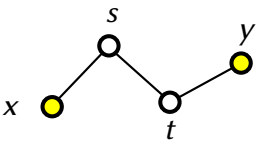
x, y, z form a triangle; the graph is triangle-free:



$$\text{triangle}(x, y, z) = \text{adj}(x, y) \wedge \text{adj}(y, z) \wedge \text{adj}(z, x).$$

$$\text{triangleFree} = \forall x. \forall y. \forall z.\ \neg\text{triangle}(x, y, z).$$

Vertices x and y are at distance $\leqslant 3$:



$$\text{dist}_{\leqslant 1}(x, y) = (x = y) \vee \text{adj}(x, y)$$

$$\text{dist}_{\leqslant 3}(x, y) = \exists s. \exists t.\ \text{dist}_{\leqslant 1}(x, s) \wedge \text{dist}_{\leqslant 1}(s, t) \wedge \text{dist}_{\leqslant 1}(t, y).$$

Every red vertex is at distance $\leqslant 3$ from a blue vertex:

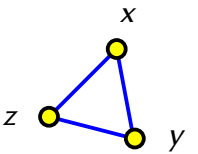
$$\text{blueDominatesRed} = \forall x.\ [\text{red}(x) \Rightarrow \exists y.\ \text{blue}(y) \wedge \text{dist}_{\leqslant 3}(x, y)]\,.$$

MSO_1: the graph is 3-colorable.

$$\text{3Col} = \exists A. \exists B. \exists C.\ \text{ind}(A) \wedge \text{ind}(B) \wedge \text{ind}(C) \wedge \forall x.\ [x \in A \vee x \in B \vee x \in C].$$

FO and MSO on graphs

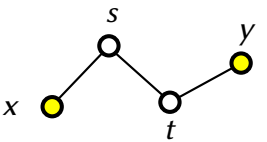
x, y, z form a triangle; the graph is triangle-free:



$$\text{triangle}(x, y, z) = \text{adj}(x, y) \wedge \text{adj}(y, z) \wedge \text{adj}(z, x).$$

$$\text{triangleFree} = \forall x. \forall y. \forall z.\ \neg\text{triangle}(x, y, z).$$

Vertices x and y are at distance $\leqslant 3$:



$$\text{dist}_{\leqslant 1}(x, y) = (x = y) \vee \text{adj}(x, y)$$

$$\text{dist}_{\leqslant 3}(x, y) = \exists s. \exists t.\ \text{dist}_{\leqslant 1}(x, s) \wedge \text{dist}_{\leqslant 1}(s, t) \wedge \text{dist}_{\leqslant 1}(t, y).$$

Every red vertex is at distance $\leqslant 3$ from a blue vertex:

$$\text{blueDominatesRed} = \forall x.\ [\text{red}(x) \Rightarrow \exists y.\ \text{blue}(y) \wedge \text{dist}_{\leqslant 3}(x, y)]\,.$$

MSO_1: the graph is 3-colorable.

$$\text{3Col} = \exists A.\ \exists B.\ \exists C.\ \text{ind}(A) \wedge \text{ind}(B) \wedge \text{ind}(C) \wedge \forall x.\ [x \in A \vee x \in B \vee x \in C].$$

MSO_2: the graph has a Hamiltonian cycle.

$$\text{Ham} = \exists F.\ [\text{conn}(F) \wedge \forall x.\ \exists^{!2} e.\ [e \in F \wedge \text{inc}(x, e)]].$$

Logic and model-checking

Logic and model-checking

First-Order logic (FO) on graphs:

Logic and model-checking

First-Order logic (FO) on graphs:

– We work with **vertex-colored** graphs (graphs with unary predicates).

Logic and model-checking

First-Order logic (FO) on graphs:

- We work with **vertex-colored** graphs (graphs with unary predicates).
- **Atomic formulas:** $x = y$, $\mathrm{adj}(x, y)$, $\mathrm{red}(x)$

Logic and model-checking

First-Order logic (FO) on graphs:

- We work with **vertex-colored** graphs (graphs with unary predicates).
- **Atomic formulas:** $x = y$, $\mathrm{adj}(x, y)$, $\mathrm{red}(x)$
- We can use **boolean connectives** and **quantifiers**.

Logic and model-checking

First-Order logic (FO) on graphs:

- We work with **vertex-colored** graphs (graphs with unary predicates).
- **Atomic formulas:** $x = y$, $\mathrm{adj}(x, y)$, $\mathrm{red}(x)$
- We can use **boolean connectives** and **quantifiers**.

Examples: there is a **clique** of size k; there is a **dominating set** of size k.

Logic and model-checking

First-Order logic (FO) on graphs:

- We work with **vertex-colored** graphs (graphs with unary predicates).
- **Atomic formulas:** $x = y$, $\mathrm{adj}(x, y)$, $\mathrm{red}(x)$
- We can use **boolean connectives** and **quantifiers**.

Examples: there is a **clique** of size k; there is a **dominating set** of size k.

Extensions:

Logic and model-checking

First-Order logic (FO) on graphs:

- We work with **vertex-colored** graphs (graphs with unary predicates).
- **Atomic formulas:** $x = y$, $\mathrm{adj}(x, y)$, $\mathrm{red}(x)$
- We can use **boolean connectives** and **quantifiers**.

Examples: there is a **clique** of size k; there is a **dominating set** of size k.

Extensions:

- MSO_1: Also quantification over **subsets** of **vertices**.

Logic and model-checking

First-Order logic (FO) on graphs:

- We work with **vertex-colored** graphs (graphs with unary predicates).
- **Atomic formulas:** $x = y$, $\mathrm{adj}(x, y)$, $\mathrm{red}(x)$
- We can use **boolean connectives** and **quantifiers**.

Examples: there is a **clique** of size k; there is a **dominating set** of size k.

Extensions:

- MSO_1: Also quantification over **subsets** of **vertices**.
- MSO_2: Also quantification over **subsets** of **vertices** and of **edges**.

Logic and model-checking

First-Order logic (FO) on graphs:

- We work with **vertex-colored** graphs (graphs with unary predicates).
- **Atomic formulas:** $x = y$, $\mathrm{adj}(x, y)$, $\mathrm{red}(x)$
- We can use **boolean connectives** and **quantifiers**.

Examples: there is a **clique** of size k; there is a **dominating set** of size k.

Extensions:

- MSO_1: Also quantification over **subsets** of **vertices**.
- MSO_2: Also quantification over **subsets** of **vertices** and of **edges**.
- **Note:** Just MSO over $(V, \mathrm{adj}(\cdot,\cdot))$ and $(V \uplus E, \mathrm{inc}(\cdot,\cdot))$ encodings.

Logic and model-checking

First-Order logic (FO) on graphs:

- We work with **vertex-colored** graphs (graphs with unary predicates).
- **Atomic formulas:** $x = y$, $\mathrm{adj}(x, y)$, $\mathrm{red}(x)$
- We can use **boolean connectives** and **quantifiers**.

Examples: there is a **clique** of size k; there is a **dominating set** of size k.

Extensions:

- MSO_1: Also quantification over **subsets** of **vertices**.
- MSO_2: Also quantification over **subsets** of **vertices** and of **edges**.
- **Note:** Just MSO over $(V, \mathrm{adj}(\cdot,\cdot))$ and $(V \uplus E, \mathrm{inc}(\cdot,\cdot))$ encodings.

Model-checking problem for logic $\mathcal{L}$

Given a graph G and a sentence $\varphi \in \mathcal{L}$, is φ true in G?

Model-checking: complexity

Model-checking: complexity

Model checking MSO_1 and MSO_2

Model-checking: complexity

Model checking MSO_1 and MSO_2

– NP-hard already for fixed sentences of MSO_1 on planar graphs.

Model-checking: complexity

Model checking MSO_1 and MSO_2

– NP-hard already for fixed sentences of MSO_1 on planar graphs.

– In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]

Model-checking: complexity

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]

Model-checking: complexity

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]
- **Bnd rankwidth** serves a similar role for MSO_1.

Model-checking: complexity

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]
- **Bnd rankwidth** serves a similar role for MSO_1.

Model checking FO

Model-checking: complexity

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]
- **Bnd rankwidth** serves a similar role for MSO_1.

Model checking FO

- **Brute-force:** $n^{\mathcal{O}(\|\varphi\|)}$.

Model-checking: complexity

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]
- **Bnd rankwidth** serves a similar role for MSO_1.

Model checking FO

- **Brute-force:** $n^{\mathcal{O}(\|\varphi\|)}$.
- In general AW[$\star$]-hard, so no **fpt** running time $\mathcal{O}_\varphi(n^c)$ expected.

Model-checking: complexity

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]
- **Bnd rankwidth** serves a similar role for MSO_1.

Model checking FO

- **Brute-force:** $n^{\mathcal{O}(\|\varphi\|)}$.
- In general AW[$\star$]-hard, so no **fpt** running time $\mathcal{O}_\varphi(n^c)$ expected.
- In time $\mathcal{O}_{\varphi,\Delta}(n)$ on graphs of **maximum degree** $\leqslant \Delta$. [Seese; '95]

Model-checking: complexity

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]
- **Bnd rankwidth** serves a similar role for MSO_1.

Model checking FO

- **Brute-force:** $n^{\mathcal{O}(\|\varphi\|)}$.
- In general AW[$\star$]-hard, so no **fpt** running time $\mathcal{O}_\varphi(n^c)$ expected.
- In time $\mathcal{O}_{\varphi,\Delta}(n)$ on graphs of **maximum degree** $\leqslant \Delta$. [Seese; '95]
- In time $\mathcal{O}_{\varphi,H}(n)$ on H**-minor-free** graphs. [Flum, Grohe; '01]

Model-checking: complexity

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]
- **Bnd rankwidth** serves a similar role for MSO_1.

Model checking FO

- **Brute-force:** $n^{\mathcal{O}(\|\varphi\|)}$.
- In general AW[$\star$]-hard, so no **fpt** running time $\mathcal{O}_\varphi(n^c)$ expected.

locality

- In time $\mathcal{O}_{\varphi,\Delta}(n)$ on graphs of **maximum degree** $\leqslant \Delta$. [Seese; '95]
- In time $\mathcal{O}_{\varphi,H}(n)$ on H-**minor-free** graphs. [Flum, Grohe; '01]

Model-checking: complexity

compositionality

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]
- **Bnd rankwidth** serves a similar role for MSO_1.

Model checking FO

- **Brute-force:** $n^{\mathcal{O}(\|\varphi\|)}$.
- In general AW[$\star$]-hard, so no **fpt** running time $\mathcal{O}_\varphi(n^c)$ expected.

locality

- In time $\mathcal{O}_{\varphi,\Delta}(n)$ on graphs of **maximum degree** $\leqslant \Delta$. [Seese; '95]
- In time $\mathcal{O}_{\varphi,H}(n)$ on H-**minor-free** graphs. [Flum, Grohe; '01]

Model-checking: complexity

compositionality

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]
- **Bnd rankwidth** serves a similar role for MSO_1.

Model checking FO

- **Brute-force:** $n^{\mathcal{O}(\|\varphi\|)}$.
- In general AW[$\star$]-hard, so no **fpt** running time $\mathcal{O}_\varphi(n^c)$ expected.

locality

- In time $\mathcal{O}_{\varphi,\Delta}(n)$ on graphs of **maximum degree** $\leqslant \Delta$. [Seese; '95]
- In time $\mathcal{O}_{\varphi,H}(n)$ on H-**minor-free** graphs. [Flum, Grohe; '01]

Goal: Understand graph classes with tractable FO model-checking.

Model-checking: complexity

compositionality

locality

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]
- **Bnd rankwidth** serves a similar role for MSO_1.

Model checking FO

- **Brute-force:** $n^{\mathcal{O}(\|\varphi\|)}$.
- In general AW[$\star$]-hard, so no **fpt** running time $\mathcal{O}_\varphi(n^c)$ expected.
- In time $\mathcal{O}_{\varphi,\Delta}(n)$ on graphs of **maximum degree** $\leqslant \Delta$. [Seese; '95]
- In time $\mathcal{O}_{\varphi,H}(n)$ on H-**minor-free** graphs. [Flum, Grohe; '01]

Goal: Understand graph classes with tractable FO model-checking.

Precisely: For what classes $\mathscr{C}$, FO-MC can be solved in fpt time $\mathcal{O}_\varphi(n^c)$?

Model-checking: complexity

compositionality

Model checking MSO_1 and MSO_2

- NP-hard already for fixed sentences of MSO_1 on planar graphs.
- In **fpt** time $f(\varphi, t) \cdot n = \mathcal{O}_{\varphi,t}(n)$ on graphs of **treewidth** t. [Courcelle; '90]
- **Bnd treewidth** is the limit for such statements. [Kreutzer, Tazari; '10]
- **Bnd rankwidth** serves a similar role for MSO_1.

Model checking FO

- **Brute-force:** $n^{\mathcal{O}(\|\varphi\|)}$.
- In general AW[⋆]-hard, so no **fpt** running time $\mathcal{O}_\varphi(n^c)$ expected.

locality

- In time $\mathcal{O}_{\varphi,\Delta}(n)$ on graphs of **maximum degree** $\leqslant \Delta$. [Seese; '95]
- In time $\mathcal{O}_{\varphi,H}(n)$ on H-**minor-free** graphs. [Flum, Grohe; '01]

Goal: Understand graph classes with tractable FO model-checking.

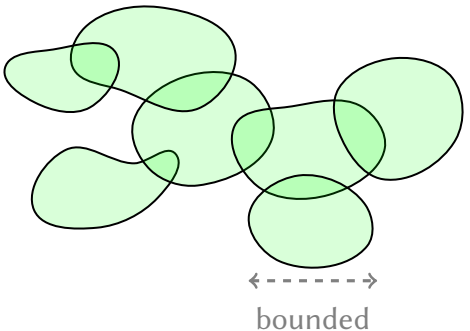
Precisely: For what classes $\mathscr{C}$, FO-MC can be solved in fpt time $\mathcal{O}_\varphi(n^c)$?

This is an **excuse** to understand those classes.

Treewidth, rankwidth, and compositionality

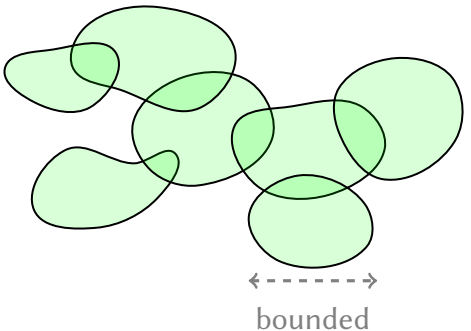
Treewidth, rankwidth, and compositionality

Treewidth:

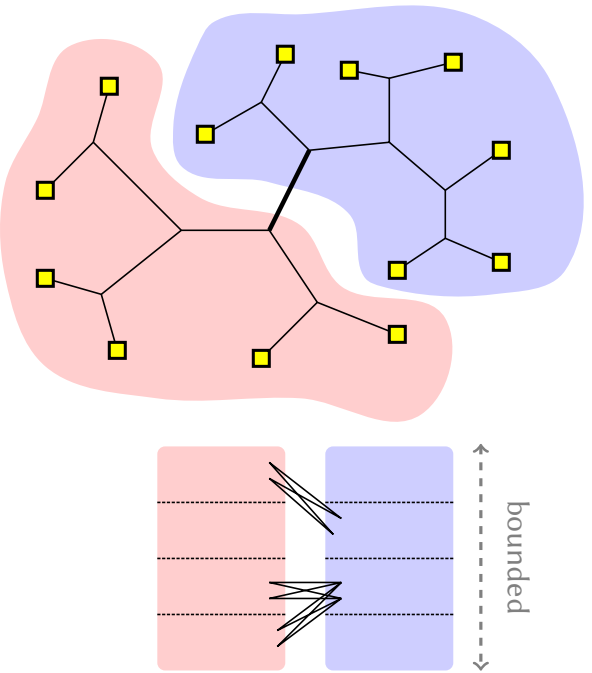


Treewidth, rankwidth, and compositionality

Treewidth:

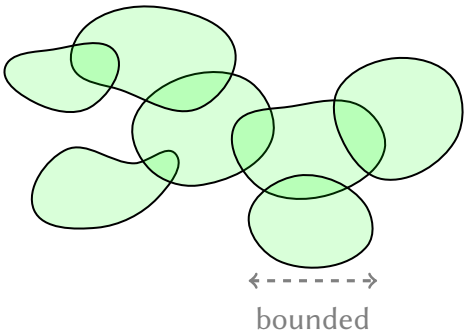


Rankwidth:

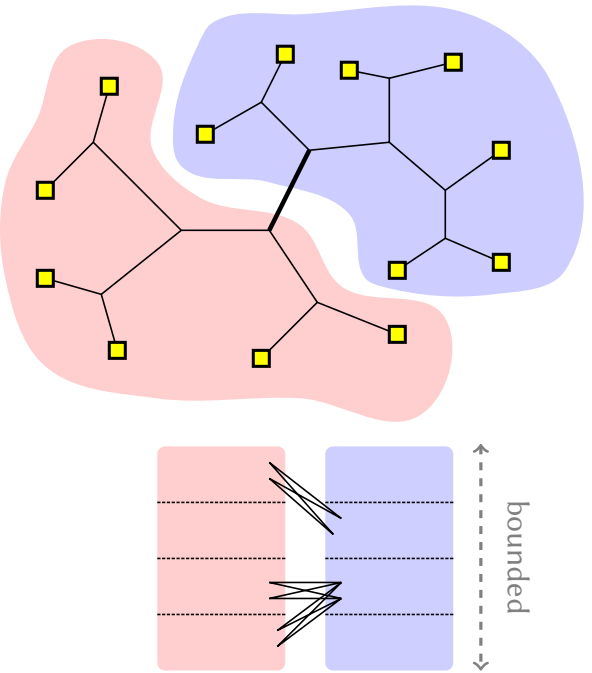


Treewidth, rankwidth, and compositionality

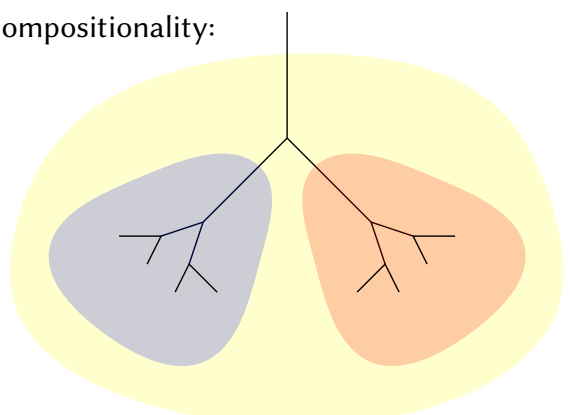
Treewidth:



Rankwidth:



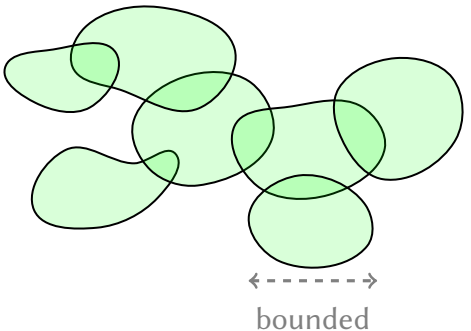
Compositionality:



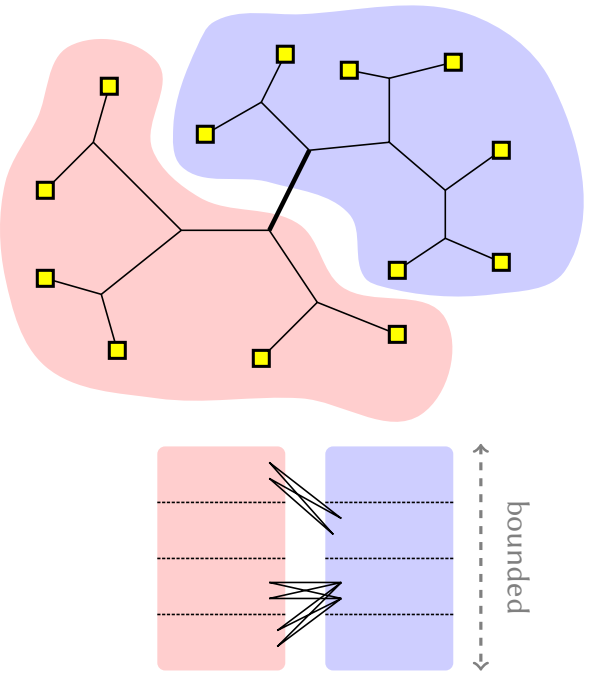
q-type of a tree computable from q-types of subtrees

Treewidth, rankwidth, and compositionality

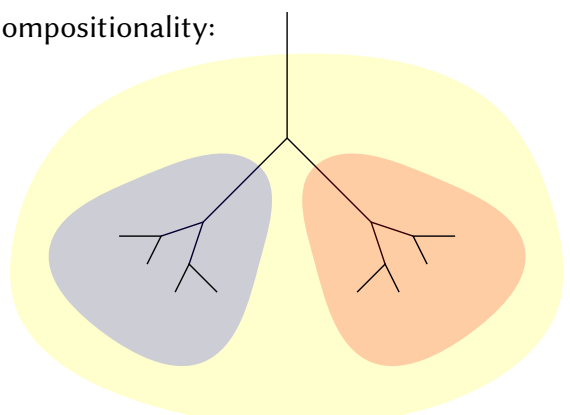
Treewidth:



Rankwidth:



Compositionality:



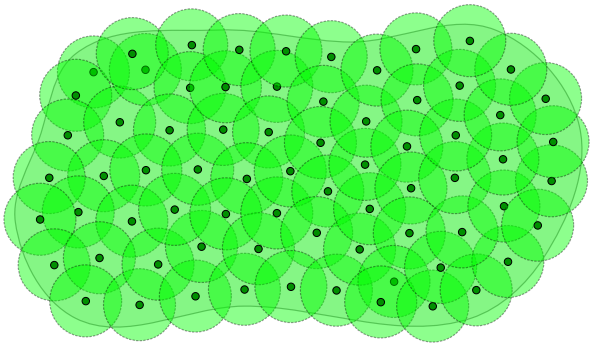
q-type of a tree computable from q-types of subtrees

Model-checking: Compute types bottom-up

Locality

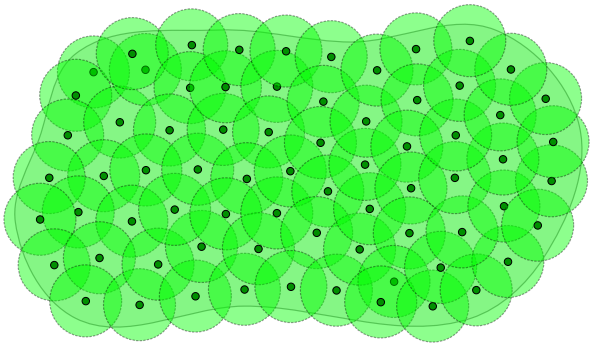
Locality

Idea: To understand the q-type of G, it suffices to understand the q-types of balls of radius $r := 2^{\mathcal{O}(q)}$.



Locality

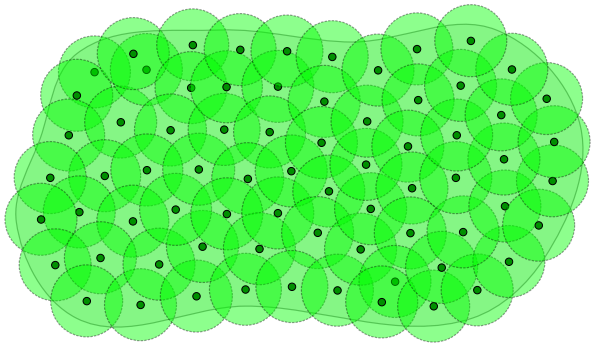
Idea: To understand the q-type of G, it suffices to understand the q-types of balls of radius $r := 2^{\mathcal{O}(q)}$.



Max degree Δ: Balls are of constant size, computing their types is trivial.

Locality

Idea: To understand the q-type of G, it suffices to understand the q-types of balls of radius $r := 2^{\mathcal{O}(q)}$.

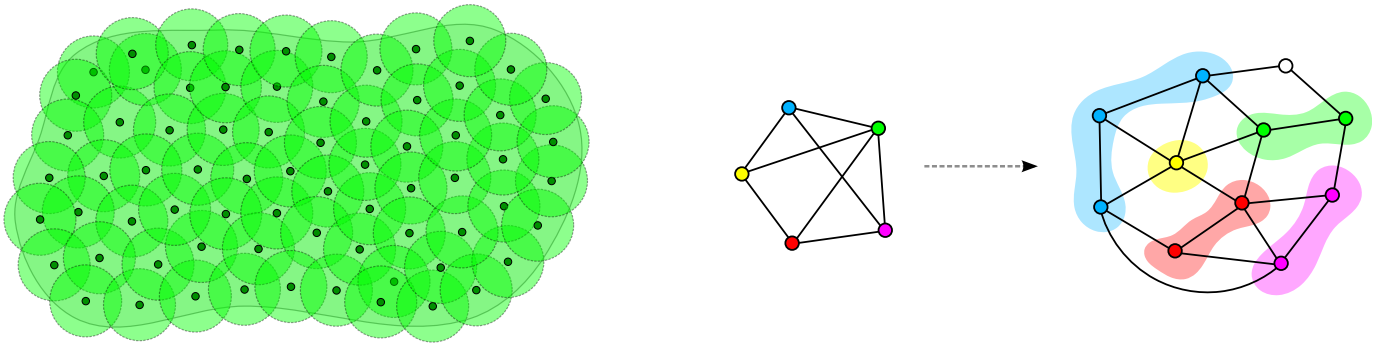


Max degree Δ: Balls are of constant size, computing their types is trivial.

Planar: Ball of radius r has **treewidth** $\leqslant 3r$, use compositionality.

Locality

Idea: To understand the q-type of G, it suffices to understand the q-types of balls of radius $r := 2^{\mathcal{O}(q)}$.



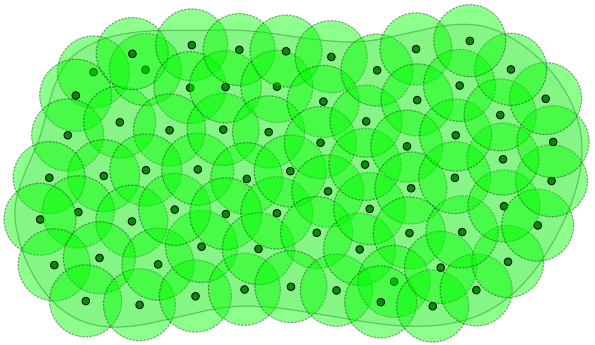
Max degree Δ: Balls are of constant size, computing their types is trivial.

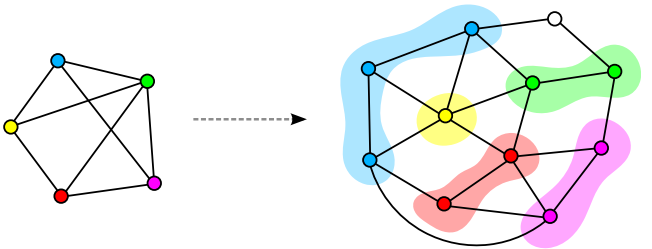
Planar: Ball of radius r has **treewidth** $\leqslant 3r$, use compositionality.

H-minor-free: Exclude H as a minor. (Planar $= \{K_5, K_{3,3}\}$-minor-free.)

Locality

Idea: To understand the q-type of G, it suffices to understand the q-types of balls of radius $r := 2^{\mathcal{O}(q)}$.





Max degree Δ: Balls are of constant size, computing their types is trivial.

Planar: Ball of radius r has **treewidth** $\leqslant 3r$, use compositionality.

H-minor-free: Exclude H as a minor. (Planar $= \{K_5, K_{3,3}\}$-minor-free.)

Thm: H-minor-free graphs admit tree decompositions into almost embeddable parts. [Robertson, Seymour]

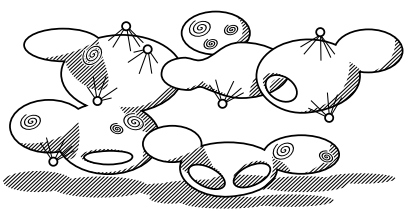


fig. by Felix Reidl

Shallow minors

Shallow minors

So far: Results of graph theory $\rightarrow$ Tractable model-checking

Shallow minors

So far: Results of graph theory $\rightarrow$ Tractable model-checking

Idea: FO is **local**... so let's exclude **local minors**.

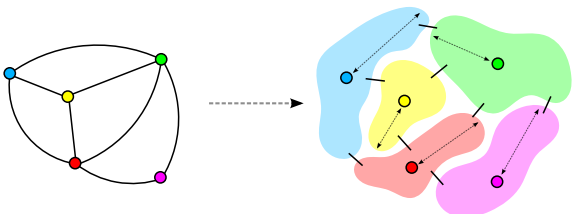
Shallow minors

So far: Results of graph theory $\rightarrow$ Tractable model-checking

Idea: FO is **local**... so let's exclude **local minors**.

H is a **depth-d** minor of G $\Leftrightarrow$

$\exists$ model of H in G with branch sets of radius $\leqslant d$



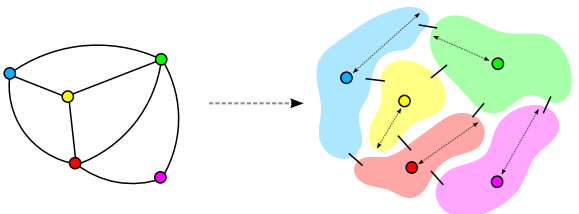
Shallow minors

So far: Results of graph theory $\rightarrow$ Tractable model-checking

Idea: FO is **local**... so let's exclude **local minors**.

H is a **depth-d** minor of G $\Leftrightarrow$

$\exists$ model of H in G with branch sets of radius $\leqslant d$



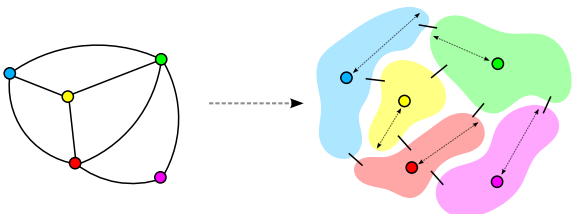
Shallow minors

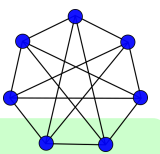
So far: Results of graph theory $\to$ Tractable model-checking

Idea: FO is **local**... so let's exclude **local minors**.

H is a **depth-d** minor of G $\Leftrightarrow$

$\exists$ model of H in G with branch sets of radius $\leqslant d$



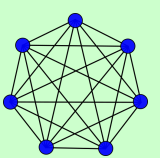


Definition

A graph class $\mathscr{C}$ has **bounded expansion** if for each $d \in \mathbb{N}$, there is $c(d)$ s.t. all depth-d minors of graphs from $\mathscr{C}$ have **avg degree** at most $c(d)$.

Definition

A graph class $\mathscr{C}$ is **nowhere dense** if for each $d \in \mathbb{N}$, there is $t(d)$ s.t. no graph $G \in \mathscr{C}$ contains the **clique** $K_{t(d)}$ as a depth-d minor.



Sparsity

Sparsity

Intuition: Sparsity **gradated** by the depth.

Sparsity

Intuition: Sparsity **gradated** by the depth.

– The larger the depth, the more complicated structures are allowed.

Sparsity

Intuition: Sparsity **gradated** by the depth.

– The larger the depth, the more complicated structures are allowed.

$\mathscr{C}$ has **bounded degree** $\Rightarrow$ $\mathscr{C}$ has **bounded expansion**

Sparsity

Intuition: Sparsity **gradated** by the depth.

– The larger the depth, the more complicated structures are allowed.

$\mathscr{C}$ has **bounded degree** $\Rightarrow$ $\mathscr{C}$ has **bounded expansion**

$\mathscr{C}$ is **minor-free** $\Rightarrow$ $\mathscr{C}$ has **bounded expansion**

Sparsity

Intuition: Sparsity **gradated** by the depth.

– The larger the depth, the more complicated structures are allowed.

$\mathscr{C}$ has **bounded degree**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ is **minor-free**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ has **bounded expansion**	$\Rightarrow$	$\mathscr{C}$ is **nowhere dense**

Sparsity

Intuition: Sparsity **gradated** by the depth.

– The larger the depth, the more complicated structures are allowed.

$\mathscr{C}$ has **bounded degree**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ is **minor-free**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ has **bounded expansion**	$\Rightarrow$	$\mathscr{C}$ is **nowhere dense**

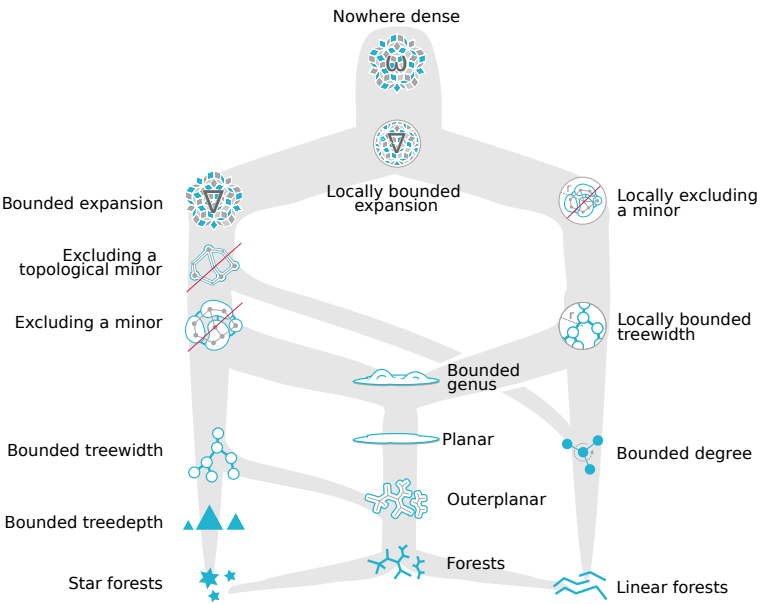


Figure by Felix Reidl

Sparsity

Intuition: Sparsity **gradated** by the depth.

– The larger the depth, the more complicated structures are allowed.

$\mathscr{C}$ has **bounded degree**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ is **minor-free**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ has **bounded expansion**	$\Rightarrow$	$\mathscr{C}$ is **nowhere dense**

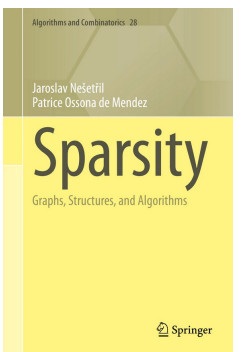
Sparsity

Intuition: Sparsity **gradated** by the depth.

– The larger the depth, the more complicated structures are allowed.

$\mathscr{C}$ has **bounded degree**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ is **minor-free**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ has **bounded expansion**	$\Rightarrow$	$\mathscr{C}$ is **nowhere dense**

This leads to the theory of **Sparsity**.



Sparsity

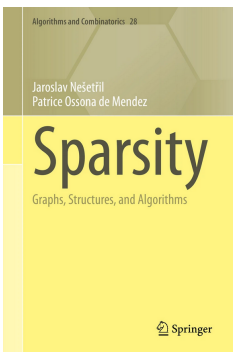
Intuition: Sparsity **gradated** by the depth.

– The larger the depth, the more complicated structures are allowed.

$\mathscr{C}$ has **bounded degree**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ is **minor-free**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ has **bounded expansion**	$\Rightarrow$	$\mathscr{C}$ is **nowhere dense**

This leads to the theory of **Sparsity**.

– **Tools:** coloring numbers, low td colorings,
flatness, neighborhood complexity, ...



Sparsity

Intuition: Sparsity **gradated** by the depth.

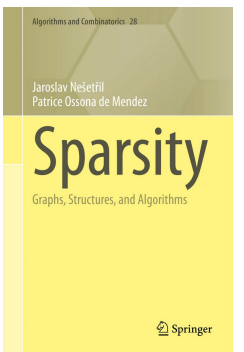
– The larger the depth, the more complicated structures are allowed.

$\mathscr{C}$ has **bounded degree**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ is **minor-free**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ has **bounded expansion**	$\Rightarrow$	$\mathscr{C}$ is **nowhere dense**

This leads to the theory of **Sparsity**.

– **Tools:** coloring numbers, low td colorings,
flatness, neighborhood complexity, ...

– **Algorithms:** parameterized, approximation, distributed...



Sparsity

Intuition: Sparsity **gradated** by the depth.

– The larger the depth, the more complicated structures are allowed.

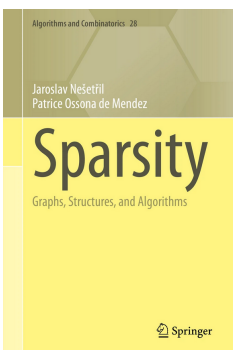
$\mathscr{C}$ has **bounded degree**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ is **minor-free**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ has **bounded expansion**	$\Rightarrow$	$\mathscr{C}$ is **nowhere dense**

This leads to the theory of **Sparsity**.

– **Tools:** coloring numbers, low td colorings,
flatness, neighborhood complexity, ...

– **Algorithms:** parameterized, approximation, distributed...

– **Applications:** problems of **local** character.



Sparsity

Intuition: Sparsity **gradated** by the depth.

– The larger the depth, the more complicated structures are allowed.

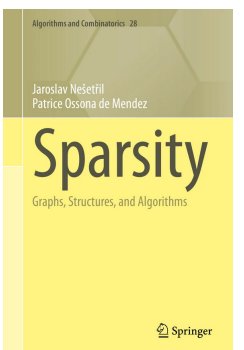
$\mathscr{C}$ has **bounded degree**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ is **minor-free**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ has **bounded expansion**	$\Rightarrow$	$\mathscr{C}$ is **nowhere dense**

This leads to the theory of **Sparsity**.

– **Tools:** coloring numbers, low td colorings,
flatness, neighborhood complexity, ...

– **Algorithms:** parameterized, approximation, distributed...

– **Applications:** problems of **local** character.



Note: We start to speak about **graph classes**.

Sparsity

Intuition: Sparsity **gradated** by the depth.

– The larger the depth, the more complicated structures are allowed.

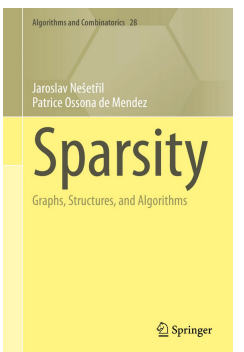
$\mathscr{C}$ has **bounded degree**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ is **minor-free**	$\Rightarrow$	$\mathscr{C}$ has **bounded expansion**
$\mathscr{C}$ has **bounded expansion**	$\Rightarrow$	$\mathscr{C}$ is **nowhere dense**

This leads to the theory of **Sparsity**.

– **Tools:** coloring numbers, low td colorings,
flatness, neighborhood complexity, ...

– **Algorithms:** parameterized, approximation, distributed...

– **Applications:** problems of **local** character.



Note: We start to speak about **graph classes**.

Sparsity is a **limit property** of a class.

Sparsity and model-checking

Sparsity and model-checking

Theorem [Dvořák, Král', Thomas; '10]

Fix a class $\mathscr{C}$ of **bounded expansion**.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_\varphi(n)$.

Sparsity and model-checking

Theorem [Dvořák, Král', Thomas; '10]

Fix a class $\mathscr{C}$ of **bounded expansion**.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_{\varphi}(n)$.

Theorem [Grohe, Kreutzer, Siebertz; '14]

Fix a **nowhere dense** class $\mathscr{C}$ and $\varepsilon > 0$.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_{\varphi,\varepsilon}(n^{1+\varepsilon})$.

Sparsity and model-checking

Theorem [Dvořák, Král', Thomas; '10]

Fix a class $\mathscr{C}$ of **bounded expansion**.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_\varphi(n)$.

Theorem [Grohe, Kreutzer, Siebertz; '14]

Fix a **nowhere dense** class $\mathscr{C}$ and $\varepsilon > 0$.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_{\varphi,\varepsilon}(n^{1+\varepsilon})$.

Theorem [Dvořák, Král', Thomas; '10]

Suppose $\mathscr{C}$ is **somewhere dense** and **subgraph-closed**.

Then FO model-checking on $\mathscr{C}$ is as hard as on general graphs.

Sparsity and model-checking

Theorem [Dvořák, Král', Thomas; '10]

Fix a class $\mathscr{C}$ of **bounded expansion**.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_{\varphi}(n)$.

Theorem [Grohe, Kreutzer, Siebertz; '14]

Fix a **nowhere dense** class $\mathscr{C}$ and $\varepsilon > 0$.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_{\varphi,\varepsilon}(n^{1+\varepsilon})$.

Theorem [Dvořák, Král', Thomas; '10]

Suppose $\mathscr{C}$ is **somewhere dense** and **subgraph-closed**.

Then FO model-checking on $\mathscr{C}$ is as hard as on general graphs.

Are we done?

Sparsity and model-checking

Theorem [Dvořák, Král', Thomas; '10]

Fix a class $\mathscr{C}$ of **bounded expansion**.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_\varphi(n)$.

Theorem [Grohe, Kreutzer, Siebertz; '14]

Fix a **nowhere dense** class $\mathscr{C}$ and $\varepsilon > 0$.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_{\varphi,\varepsilon}(n^{1+\varepsilon})$.

Theorem [Dvořák, Král', Thomas; '10]

Suppose $\mathscr{C}$ is **somewhere dense** and **subgraph-closed**.

Then FO model-checking on $\mathscr{C}$ is as hard as on general graphs.

Are we done? **Not quite**

Sparsity and model-checking

Theorem [Dvořák, Král', Thomas; '10]

Fix a class $\mathscr{C}$ of **bounded expansion**.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_\varphi(n)$.

Theorem [Grohe, Kreutzer, Siebertz; '14]

Fix a **nowhere dense** class $\mathscr{C}$ and $\varepsilon > 0$.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_{\varphi,\varepsilon}(n^{1+\varepsilon})$.

Theorem [Dvořák, Král', Thomas; '10]

Suppose $\mathscr{C}$ is **somewhere dense** and **subgraph-closed**.

Then FO model-checking on $\mathscr{C}$ is as hard as on general graphs.

Are we done? **Not quite**

Examples of classes with tractable FO model-checking:

Sparsity and model-checking

Theorem [Dvořák, Král', Thomas; '10]

Fix a class $\mathscr{C}$ of **bounded expansion**.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_\varphi(n)$.

Theorem [Grohe, Kreutzer, Siebertz; '14]

Fix a **nowhere dense** class $\mathscr{C}$ and $\varepsilon > 0$.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_{\varphi,\varepsilon}(n^{1+\varepsilon})$.

Theorem [Dvořák, Král', Thomas; '10]

Suppose $\mathscr{C}$ is **somewhere dense** and **subgraph-closed**.

Then FO model-checking on $\mathscr{C}$ is as hard as on general graphs.

Are we done? **Not quite**

Examples of classes with tractable FO model-checking:

– complements of planar graphs;

Sparsity and model-checking

Theorem [Dvořák, Král', Thomas; '10]

Fix a class $\mathscr{C}$ of **bounded expansion**.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_\varphi(n)$.

Theorem [Grohe, Kreutzer, Siebertz; '14]

Fix a **nowhere dense** class $\mathscr{C}$ and $\varepsilon > 0$.

Then FO model-checking on $\mathscr{C}$ can be done in time $\mathcal{O}_{\varphi,\varepsilon}(n^{1+\varepsilon})$.

Theorem [Dvořák, Král', Thomas; '10]

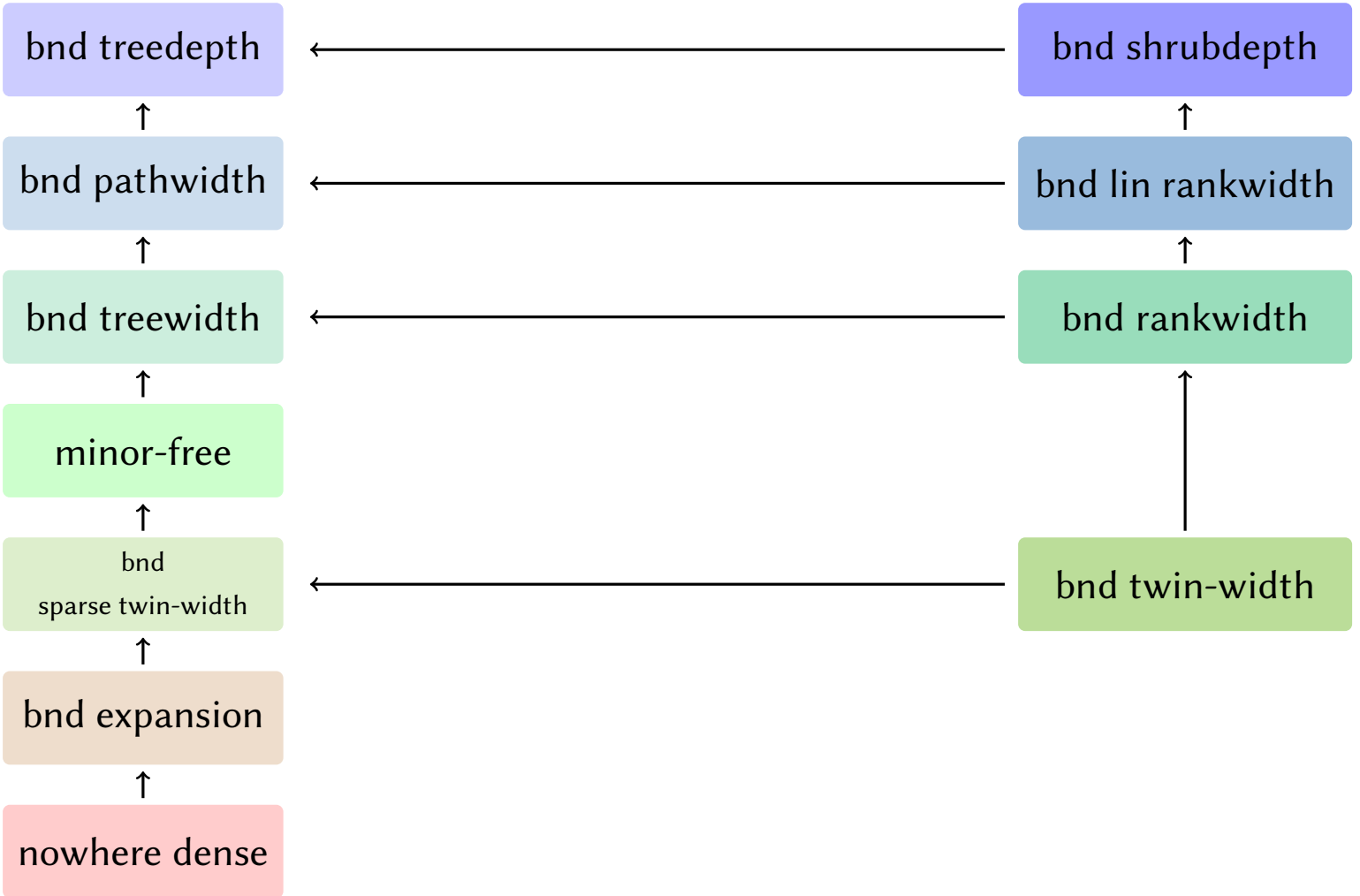
Suppose $\mathscr{C}$ is **somewhere dense** and **subgraph-closed**.

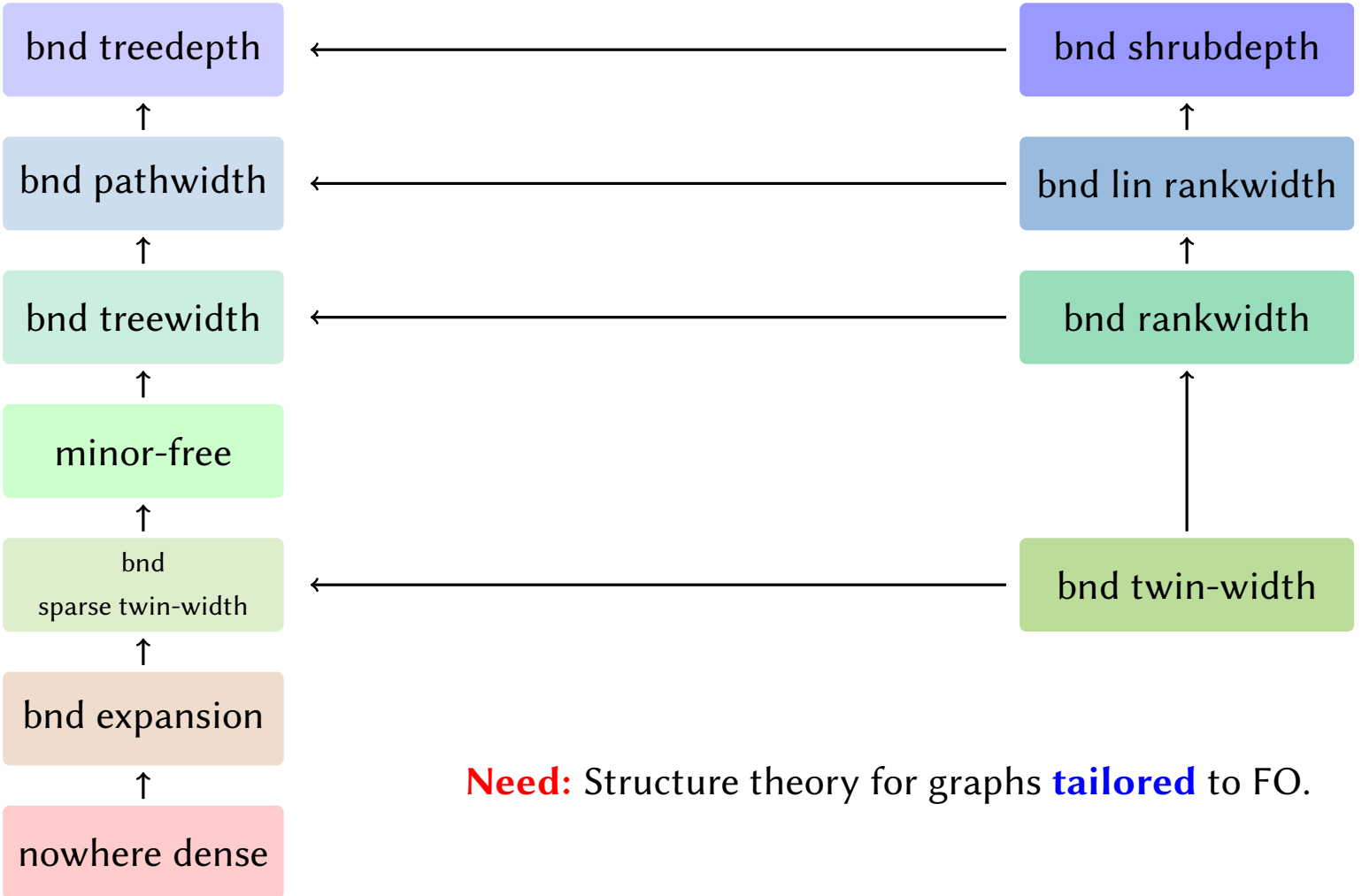
Then FO model-checking on $\mathscr{C}$ is as hard as on general graphs.

Are we done? **Not quite**

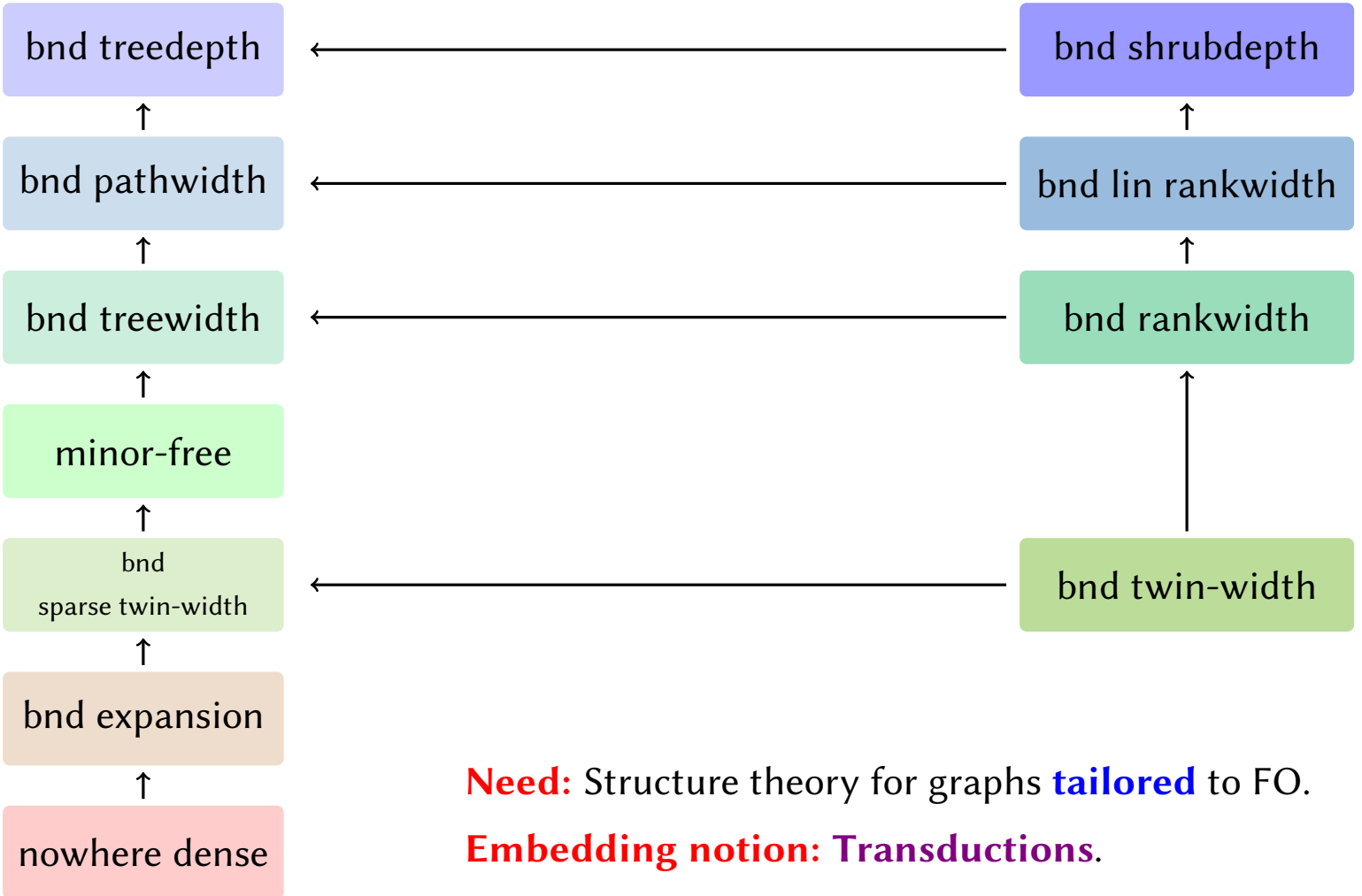
Examples of classes with tractable FO model-checking:

- complements of planar graphs;
- classes of bounded **rankwidth** or **twin-width**.


bnd treedepth
bnd shrubdepth
bnd pathwidth
bnd lin rankwidth
bnd treewidth
bnd rankwidth
minor-free
bnd
sparse twin-width
bnd twin-width
bnd expansion
nowhere dense



Need: Structure theory for graphs **tailored** to FO.



Need: Structure theory for graphs **tailored** to FO.

Embedding notion: Transductions.

Simple FO interpretations

Simple FO interpretations

G is a (colored) graph, $\varphi(x, y)$ is a symmetric FO formula.

Simple FO interpretations

G is a (colored) graph, $\varphi(x, y)$ is a symmetric FO formula.

Define $\varphi(G)$ as follows:

$$\begin{aligned} \text{vertices of } \varphi(G) &= \text{vertices of } G \\ \text{edges of } \varphi(G) &= \text{all } uv \text{ such that } \varphi(u, v) \text{ holds in } G \end{aligned}$$

Simple FO interpretations

G is a (colored) graph, $\varphi(x, y)$ is a symmetric FO formula.

Define $\varphi(G)$ as follows:

$$\begin{aligned} \text{vertices of } \varphi(G) &= \text{vertices of } G \\ \text{edges of } \varphi(G) &= \text{all } uv \text{ such that } \varphi(u, v) \text{ holds in } G \end{aligned}$$

Examples:

Simple FO interpretations

G is a (colored) graph, $\varphi(x, y)$ is a symmetric FO formula.

Define $\varphi(G)$ as follows:

$$\begin{aligned} \text{vertices of } \varphi(G) &= \text{vertices of } G \\ \text{edges of } \varphi(G) &= \text{all } uv \text{ such that } \varphi(u, v) \text{ holds in } G \end{aligned}$$

Examples:

$\varphi(x, y) = \neg \mathrm{adj}(x, y) \qquad \rightsquigarrow \qquad \varphi(G)$ is the **complement** of G

Simple FO interpretations

G is a (colored) graph, $\varphi(x, y)$ is a symmetric FO formula.

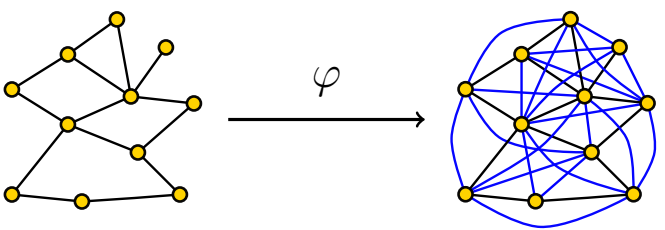
Define $\varphi(G)$ as follows:

$$\begin{aligned} \text{vertices of } \varphi(G) &= \text{vertices of } G \\ \text{edges of } \varphi(G) &= \text{all } uv \text{ such that } \varphi(u, v) \text{ holds in } G \end{aligned}$$

Examples:

$$\varphi(x, y) = \neg \mathrm{adj}(x, y) \quad \rightsquigarrow \quad \varphi(G) \text{ is the } \textbf{complement} \text{ of } G$$

$$\varphi(x, y) = \mathrm{dist}_{\leqslant 2}(x, y) \quad \rightsquigarrow \quad \varphi(G) \text{ is the } \textbf{square} \text{ of } G$$



Simple FO interpretations

G is a (colored) graph, $\varphi(x, y)$ is a symmetric FO formula.

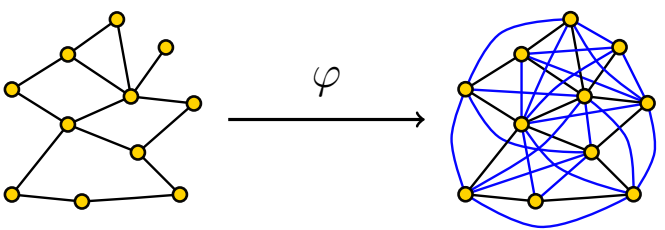
Define $\varphi(G)$ as follows:

$$\begin{aligned} \text{vertices of } \varphi(G) &= \text{vertices of } G \\ \text{edges of } \varphi(G) &= \text{all } uv \text{ such that } \varphi(u, v) \text{ holds in } G \end{aligned}$$

Examples:

$\varphi(x, y) = \neg \mathrm{adj}(x, y) \quad \rightsquigarrow \quad \varphi(G)$ is the **complement** of G

$\varphi(x, y) = \mathrm{dist}_{\leqslant 2}(x, y) \quad \rightsquigarrow \quad \varphi(G)$ is the **square** of G



Intuition: $\varphi(G)$ can be logically encoded in G.

Simple FO interpretations

G is a (colored) graph, $\varphi(x, y)$ is a symmetric FO formula.

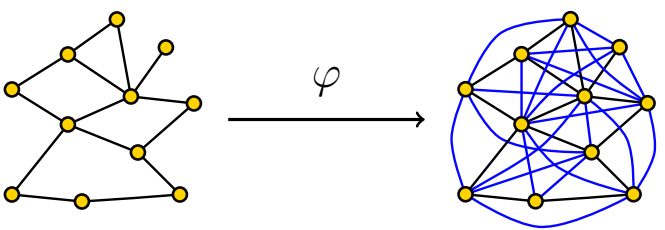
Define $\varphi(G)$ as follows:

$$\begin{aligned} \text{vertices of } \varphi(G) &= \text{vertices of } G \\ \text{edges of } \varphi(G) &= \text{all } uv \text{ such that } \varphi(u, v) \text{ holds in } G \end{aligned}$$

Examples:

$\varphi(x, y) = \neg \mathrm{adj}(x, y) \quad \rightsquigarrow \quad \varphi(G)$ is the **complement** of G

$\varphi(x, y) = \mathrm{dist}_{\leqslant 2}(x, y) \quad \rightsquigarrow \quad \varphi(G)$ is the **square** of G



Intuition: $\varphi(G)$ can be logically encoded in G.

Obs: Model-checking ψ on $\varphi(G)$ **reduces** to

model-checking $\psi[\mathrm{adj} \to \varphi]$ on G.

FO transductions

FO transductions

A **transduction** is a mechanism $\mathsf{T} = (C, \varphi)$ that takes G and applies:

FO transductions

A **transduction** is a mechanism $\mathsf{T} = (C, \varphi)$ that takes G and applies:

– **Color** vertices of G using a fixed palette of colors C. $G \rightsquigarrow \widehat{G}$

FO transductions

A **transduction** is a mechanism $\mathsf{T} = (C, \varphi)$ that takes G and applies:

- **Color** vertices of G using a fixed palette of colors C. $\quad G \rightsquigarrow \widehat{G}$
- Apply a fixed **intepretation** φ. $\quad \widehat{G} \rightsquigarrow \varphi(\widehat{G})$

FO transductions

A **transduction** is a mechanism $\mathsf{T} = (C, \varphi)$ that takes G and applies:

- **Color** vertices of G using a fixed palette of colors C. $G \rightsquigarrow \widehat{G}$
- Apply a fixed **intepretation** φ. $\widehat{G} \rightsquigarrow \varphi(\widehat{G})$
- Output any **induced subgraph** of $\varphi(\widehat{G})$ and drop colors.

FO transductions

A **transduction** is a mechanism $\mathsf{T} = (C, \varphi)$ that takes G and applies:

- **Color** vertices of G using a fixed palette of colors C. $G \rightsquigarrow \widehat{G}$
- Apply a fixed **intepretation** φ. $\widehat{G} \rightsquigarrow \varphi(\widehat{G})$
- Output any **induced subgraph** of $\varphi(\widehat{G})$ and drop colors.

Note: Coloring makes a formula for output domain unnecessary.

FO transductions

A **transduction** is a mechanism $\mathsf{T} = (C, \varphi)$ that takes G and applies:

- **Color** vertices of G using a fixed palette of colors C. $G \rightsquigarrow \widehat{G}$
- Apply a fixed **intepretation** φ. $\widehat{G} \rightsquigarrow \varphi(\widehat{G})$
- Output any **induced subgraph** of $\varphi(\widehat{G})$ and drop colors.

Note: Coloring makes a formula for output domain unnecessary.

Note: A transduction is a **nondeterministic** mechanism.

FO transductions

A **transduction** is a mechanism $\mathsf{T} = (C, \varphi)$ that takes G and applies:

- **Color** vertices of G using a fixed palette of colors C. $\quad G \rightsquigarrow \widehat{G}$
- Apply a fixed **intepretation** φ. $\quad \widehat{G} \rightsquigarrow \varphi(\widehat{G})$
- Output any **induced subgraph** of $\varphi(\widehat{G})$ and drop colors.

Note: Coloring makes a formula for output domain unnecessary.

Note: A transduction is a **nondeterministic** mechanism.

$$\mathsf{T}(G) \coloneqq \text{all possible outputs of } \mathsf{T} \text{ on } G.$$

FO transductions

A **transduction** is a mechanism $\mathsf{T} = (C, \varphi)$ that takes G and applies:

- **Color** vertices of G using a fixed palette of colors C. $G \rightsquigarrow \widehat{G}$
- Apply a fixed **intepretation** φ. $\widehat{G} \rightsquigarrow \varphi(\widehat{G})$
- Output any **induced subgraph** of $\varphi(\widehat{G})$ and drop colors.

Note: Coloring makes a formula for output domain unnecessary.

Note: A transduction is a **nondeterministic** mechanism.

$$\mathsf{T}(G) := \text{all possible outputs of } \mathsf{T} \text{ on } G.$$

For a class $\mathscr{C}$ and a transduction T, define

$$\mathsf{T}(\mathscr{C}) := \bigcup_{G \in \mathscr{C}} \mathsf{T}(G)$$

FO transductions

A **transduction** is a mechanism $\mathsf{T} = (C, \varphi)$ that takes G and applies:

- **Color** vertices of G using a fixed palette of colors C. $\quad G \rightsquigarrow \widehat{G}$
- Apply a fixed **intepretation** φ. $\quad \widehat{G} \rightsquigarrow \varphi(\widehat{G})$
- Output any **induced subgraph** of $\varphi(\widehat{G})$ and drop colors.

Note: Coloring makes a formula for output domain unnecessary.

Note: A transduction is a **nondeterministic** mechanism.

$$\mathsf{T}(G) := \text{all possible outputs of } \mathsf{T} \text{ on } G.$$

For a class $\mathscr{C}$ and a transduction T, define

$$\mathsf{T}(\mathscr{C}) := \bigcup_{G \in \mathscr{C}} \mathsf{T}(G)$$

Call $\mathscr{D}$ **transducible** from $\mathscr{C}$ if there is a transduction T such that

$$\mathscr{D} \subseteq \mathsf{T}(\mathscr{C}).$$

FO transductions

A **transduction** is a mechanism $\mathsf{T} = (C, \varphi)$ that takes G and applies:

- **Color** vertices of G using a fixed palette of colors C. $\quad G \rightsquigarrow \widehat{G}$
- Apply a fixed **intepretation** φ. $\quad \widehat{G} \rightsquigarrow \varphi(\widehat{G})$
- Output any **induced subgraph** of $\varphi(\widehat{G})$ and drop colors.

Note: Coloring makes a formula for output domain unnecessary.

Note: A transduction is a **nondeterministic** mechanism.

$$\mathsf{T}(G) := \text{all possible outputs of } \mathsf{T} \text{ on } G.$$

For a class $\mathscr{C}$ and a transduction T, define

$$\mathsf{T}(\mathscr{C}) := \bigcup_{G \in \mathscr{C}} \mathsf{T}(G)$$

Call $\mathscr{D}$ **transducible** from $\mathscr{C}$ if there is a transduction T such that

$$\mathscr{D} \subseteq \mathsf{T}(\mathscr{C}).$$

Idea: Graphs from $\mathscr{D}$ can be **logically encoded** in colored graphs from $\mathscr{C}$.

FO transductions

A **transduction** is a mechanism $\mathsf{T} = (C, \varphi)$ that takes G and applies:

- **Color** vertices of G using a fixed palette of colors C. $\quad G \rightsquigarrow \widehat{G}$
- Apply a fixed **intepretation** φ. $\quad \widehat{G} \rightsquigarrow \varphi(\widehat{G})$
- Output any **induced subgraph** of $\varphi(\widehat{G})$ and drop colors.

Note: Coloring makes a formula for output domain unnecessary.

Note: A transduction is a **nondeterministic** mechanism.

$$\mathsf{T}(G) := \text{all possible outputs of } \mathsf{T} \text{ on } G.$$

For a class $\mathscr{C}$ and a transduction T, define

$$\mathsf{T}(\mathscr{C}) := \bigcup_{G \in \mathscr{C}} \mathsf{T}(G)$$

Call $\mathscr{D}$ **transducible** from $\mathscr{C}$ if there is a transduction T such that

$$\mathscr{D} \subseteq \mathsf{T}(\mathscr{C}).$$

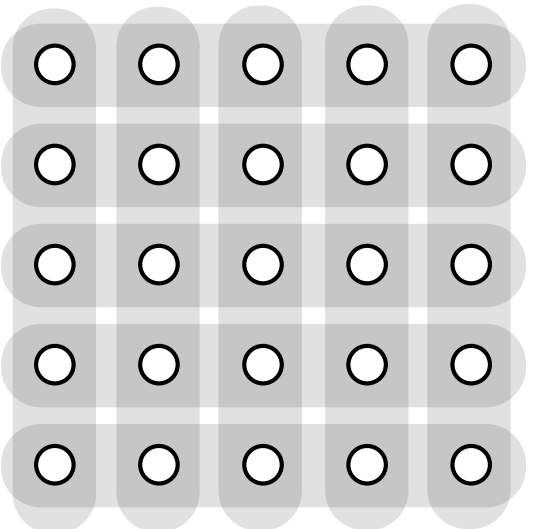
Idea: Graphs from $\mathscr{D}$ can be **logically encoded** in colored graphs from $\mathscr{C}$.

Obs: Transductions closed under composition $\Rightarrow$ **Quasi-order** on classes

Transductions: Example

Transductions: Example

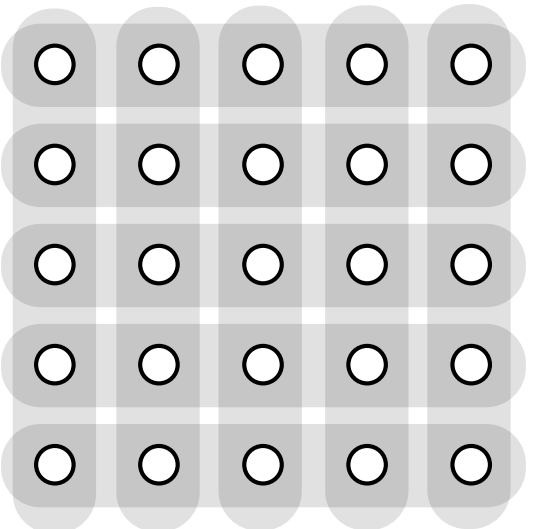
$\mathscr{C} := \{\textit{rook graphs}\} =$ grids with rows and columns made into cliques.



Transductions: Example

$\mathscr{C} := \{\textit{rook graphs}\}$ = grids with rows and columns made into cliques.

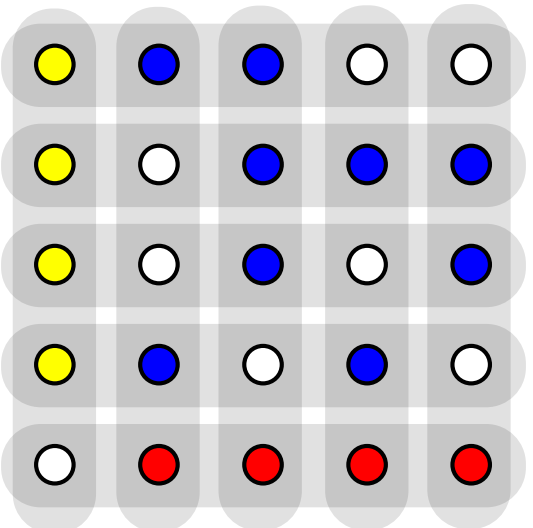
Claim: {all bipartite graphs} is transducible from $\mathscr{C}$.



Transductions: Example

$\mathscr{C} := \{\textit{rook graphs}\}$ = grids with rows and columns made into cliques.

Claim: {all bipartite graphs} is transducible from $\mathscr{C}$.

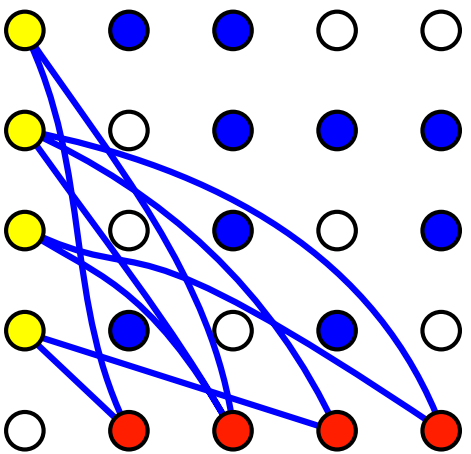


First row & column using **red** & **yellow**, the adjacency matrix using **blue**.

Transductions: Example

$\mathscr{C} := \{\textit{rook graphs}\}$ = grids with rows and columns made into cliques.

Claim: {all bipartite graphs} is transducible from $\mathscr{C}$.



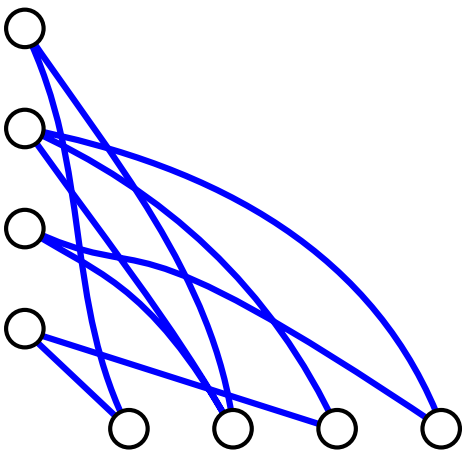
First row & column using **red** & **yellow**, the adjacency matrix using **blue**.

$\varphi(x, y)$ = **red**(x) and **yellow**(y) and x, y have a common **blue** neighbor.

Transductions: Example

$\mathscr{C} := \{\textit{rook graphs}\}$ = grids with rows and columns made into cliques.

Claim: {all bipartite graphs} is transducible from $\mathscr{C}$.



First row & column using **red** & **yellow**, the adjacency matrix using **blue**.

$\varphi(x, y)$ = **red**(x) and **yellow**(y) and x, y have a common **blue** neighbor.

Drop all **blue** and **white** vertices.

Ideals

Theorem

The following properties of graph classes are closed under transductions:

bnd shrubdepth, **bnd lin rankwidth,**

bnd rankwidth, **bnd twin-width.**

Ideals

Theorem

The following properties of graph classes are closed under transductions:

bnd shrubdepth, **bnd lin rankwidth,**

bnd rankwidth, **bnd twin-width.**

We say that these are **FO ideals**.

Ideals

Theorem

The following properties of graph classes are closed under transductions:

bnd shrubdepth, **bnd lin rankwidth,**

bnd rankwidth, **bnd twin-width.**

We say that these are **FO ideals**.

Natural ways of conceiving new ideals:

Ideals

Theorem

The following properties of graph classes are closed under transductions:

bnd shrubdepth, **bnd lin rankwidth,**

bnd rankwidth, **bnd twin-width.**

We say that these are **FO ideals**.

Natural ways of conceiving new ideals:

Obstructions:

If $\mathscr{O}$ is a class, then $\{$All classes that do **not** transduce $\mathscr{O}\}$ is an ideal.

Ideals

Theorem

The following properties of graph classes are closed under transductions:

bnd shrubdepth, **bnd lin rankwidth,**

bnd rankwidth, **bnd twin-width.**

We say that these are **FO ideals**.

Natural ways of conceiving new ideals:

Obstructions:

If $\mathscr{O}$ is a class, then $\{$All classes that do **not** transduce $\mathscr{O}\}$ is an ideal.

Closure:

If $\mathcal{P}$ is a property, then

$\{$All classes **transducible** from any $\mathscr{C} \in \mathcal{P}\}$ is an ideal.

We call those classes **structurally** $\mathcal{P}$.

Monadic stability and dependence

Monadic stability and dependence

Def: $\mathscr{C}$ is **monadically dependent** if $\{\text{all graphs}\}$ is not transducible from $\mathscr{C}$.

Monadic stability and dependence

Def: $\mathscr{C}$ is **monadically dependent** if $\{\text{all graphs}\}$ is not transducible from $\mathscr{C}$.

Monadic stability and dependence

Def: $\mathscr{C}$ is **monadically dependent** if $\{\text{all graphs}\}$ is not transducible from $\mathscr{C}$.

Intuition: Weakest possible restriction, cannot encode all graphs in $\mathscr{C}$.

Monadic stability and dependence

Def: $\mathscr{C}$ is **monadically dependent** if $\{$all graphs$\}$ is not transducible from $\mathscr{C}$.

Intuition: Weakest possible restriction, cannot encode all graphs in $\mathscr{C}$.

Ex: $\{$rook graphs$\}$ is **not mon dependent**.

Monadic stability and dependence

Def: $\mathscr{C}$ is **monadically dependent** if $\{$all graphs$\}$ is not transducible from $\mathscr{C}$.

Intuition: Weakest possible restriction, cannot encode all graphs in $\mathscr{C}$.

Ex: $\{$rook graphs$\}$ is **not mon dependent**.

Ex: mon dependence for $\mathrm{MSO}_1/\mathrm{MSO}_2$ = **bnd rankwidth/treewidth**.

Monadic stability and dependence

Def: $\mathscr{C}$ is **monadically dependent** if $\{\text{all graphs}\}$ is not transducible from $\mathscr{C}$.

Intuition: Weakest possible restriction, cannot encode all graphs in $\mathscr{C}$.

Ex: $\{\text{rook graphs}\}$ is **not mon dependent**.

Ex: mon dependence for $\mathrm{MSO}_1/\mathrm{MSO}_2$ = **bnd rankwidth/treewidth**.

Ex: mon dependence for ordered graphs = **bnd twin-width**. [BGOdMSTT; '22]

Monadic stability and dependence

Def: $\mathscr{C}$ is **monadically dependent** if $\{\text{all graphs}\}$ is not transducible from $\mathscr{C}$.

Intuition: Weakest possible restriction, cannot encode all graphs in $\mathscr{C}$.

Ex: $\{\text{rook graphs}\}$ is **not mon dependent**.

Ex: mon dependence for $\mathrm{MSO}_1/\mathrm{MSO}_2$ = **bnd rankwidth/treewidth**.

Ex: mon dependence for ordered graphs = **bnd twin-width**. [BGOdMSTT; '22]

Direct consequences of characterizations via grid-like obstructions.

Monadic stability and dependence

Def: $\mathscr{C}$ is **monadically dependent** if {all graphs} is not transducible from $\mathscr{C}$.

Intuition: Weakest possible restriction, cannot encode all graphs in $\mathscr{C}$.

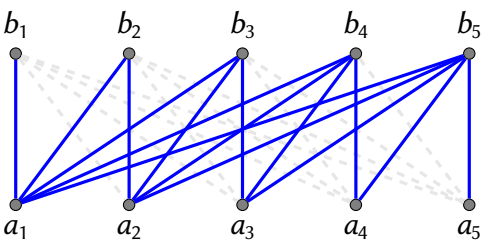
Ex: {rook graphs} is **not mon dependent**.

Ex: mon dependence for $\mathrm{MSO}_1/\mathrm{MSO}_2$ = **bnd rankwidth/treewidth**.

Ex: mon dependence for ordered graphs = **bnd twin-width**. [BGOdMSTT; '22]

Direct consequences of characterizations via grid-like obstructions.

Def: $\mathscr{C}$ is **monadically stable** if {all half-graphs} is not transducible from $\mathscr{C}$.



a_i and b_j adjacent $\iff$ $i \leqslant j$

Monadic stability and dependence

Def: $\mathscr{C}$ is **monadically dependent** if {all graphs} is not transducible from $\mathscr{C}$.

Intuition: Weakest possible restriction, cannot encode all graphs in $\mathscr{C}$.

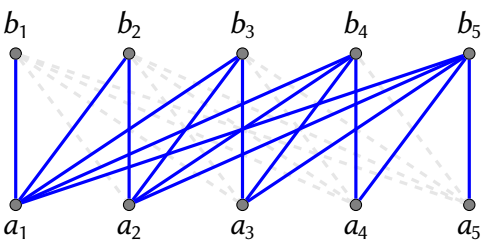
Ex: {rook graphs} is **not mon dependent**.

Ex: mon dependence for $\mathrm{MSO}_1/\mathrm{MSO}_2$ = **bnd rankwidth/treewidth**.

Ex: mon dependence for ordered graphs = **bnd twin-width**. [BGOdMSTT; '22]

Direct consequences of characterizations via grid-like obstructions.

Def: $\mathscr{C}$ is **monadically stable** if {all half-graphs} is not transducible from $\mathscr{C}$.



a_i and b_j adjacent $\iff$ $i \leqslant j$

Monadic stability and dependence

Def: $\mathscr{C}$ is **monadically dependent** if {all graphs} is not transducible from $\mathscr{C}$.

Intuition: Weakest possible restriction, cannot encode all graphs in $\mathscr{C}$.

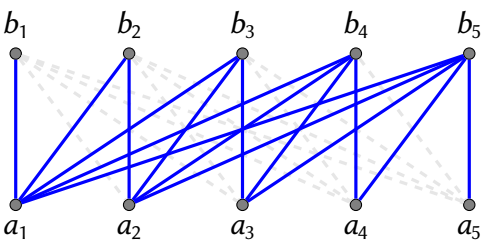
Ex: {rook graphs} is **not mon dependent**.

Ex: mon dependence for $\mathrm{MSO}_1/\mathrm{MSO}_2$ = **bnd rankwidth/treewidth**.

Ex: mon dependence for ordered graphs = **bnd twin-width**. [BGOdMSTT; '22]

Direct consequences of characterizations via grid-like obstructions.

Def: $\mathscr{C}$ is **monadically stable** if {all half-graphs} is not transducible from $\mathscr{C}$.



a_i and b_j adjacent $\iff$ $i \leqslant j$

Intuition: **Monadically stable** classes comprise of **orderless** graphs.

Monadic stability and dependence

Def: $\mathscr{C}$ is **monadically dependent** if {all graphs} is not transducible from $\mathscr{C}$.

Intuition: Weakest possible restriction, cannot encode all graphs in $\mathscr{C}$.

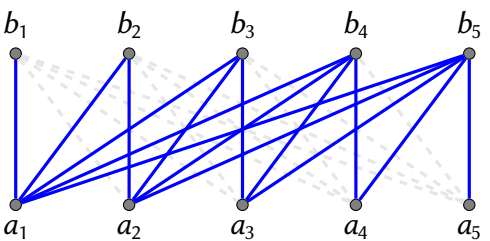
Ex: {rook graphs} is **not mon dependent**.

Ex: mon dependence for $\mathrm{MSO}_1/\mathrm{MSO}_2$ = **bnd rankwidth/treewidth**.

Ex: mon dependence for ordered graphs = **bnd twin-width**. [BGOdMSTT; '22]

Direct consequences of characterizations via grid-like obstructions.

Def: $\mathscr{C}$ is **monadically stable** if {all half-graphs} is not transducible from $\mathscr{C}$.



a_i and b_j adjacent $\iff$ $i \leqslant j$

Intuition: **Monadically stable** classes comprise of **orderless** graphs.

Concepts studied by Baldwin and Shelah in the 80s.

Relation to sparsity

Relation to sparsity

Theorem [Adler and Adler, after Podewski and Ziegler; '14 and '79]

Every **nowhere dense** class $\mathscr{C}$ is **monadically stable**.

Relation to sparsity

Theorem [Adler and Adler, after Podewski and Ziegler; '14 and '79]

Every **nowhere dense** class $\mathscr{C}$ is **monadically stable**.

In fact, if $\mathscr{C}$ is **subgraph-closed**, then

$\mathscr{C}$ is **nowhere dense** $\Leftrightarrow$ $\mathscr{C}$ is **mon stable** $\Leftrightarrow$ $\mathscr{C}$ is **mon dependent**.

Relation to sparsity

Theorem [Adler and Adler, after Podewski and Ziegler; '14 and '79]

Every **nowhere dense** class $\mathscr{C}$ is **monadically stable**.

In fact, if $\mathscr{C}$ is **subgraph-closed**, then

$\mathscr{C}$ is **nowhere dense** $\Leftrightarrow$ $\mathscr{C}$ is **mon stable** $\Leftrightarrow$ $\mathscr{C}$ is **mon dependent**.

Idea: **Mon stability** and **mon dependence** are purely logic notions,
and collapse to **nowhere denseness** for subgraph-closed classes.

Relation to sparsity

Theorem [Adler and Adler, after Podewski and Ziegler; '14 and '79]

Every **nowhere dense** class $\mathscr{C}$ is **monadically stable**.

In fact, if $\mathscr{C}$ is **subgraph-closed**, then

$\mathscr{C}$ is **nowhere dense** $\Leftrightarrow$ $\mathscr{C}$ is **mon stable** $\Leftrightarrow$ $\mathscr{C}$ is **mon dependent**.

Idea: **Mon stability** and **mon dependence** are purely logic notions,

and collapse to **nowhere denseness** for subgraph-closed classes.

Conjecture

Suppose $\mathscr{C}$ is **monadically dependent**.

Then FO model-checking on $\mathscr{C}$ can be solved in time $\mathcal{O}_\varphi(n^c)$.

Relation to sparsity

Theorem [Adler and Adler, after Podewski and Ziegler; '14 and '79]

Every **nowhere dense** class $\mathscr{C}$ is **monadically stable**.

In fact, if $\mathscr{C}$ is **subgraph-closed**, then

$\mathscr{C}$ is **nowhere dense** $\Leftrightarrow$ $\mathscr{C}$ is **mon stable** $\Leftrightarrow$ $\mathscr{C}$ is **mon dependent**.

Idea: **Mon stability** and **mon dependence** are purely logic notions,
and collapse to **nowhere denseness** for subgraph-closed classes.

Conjecture

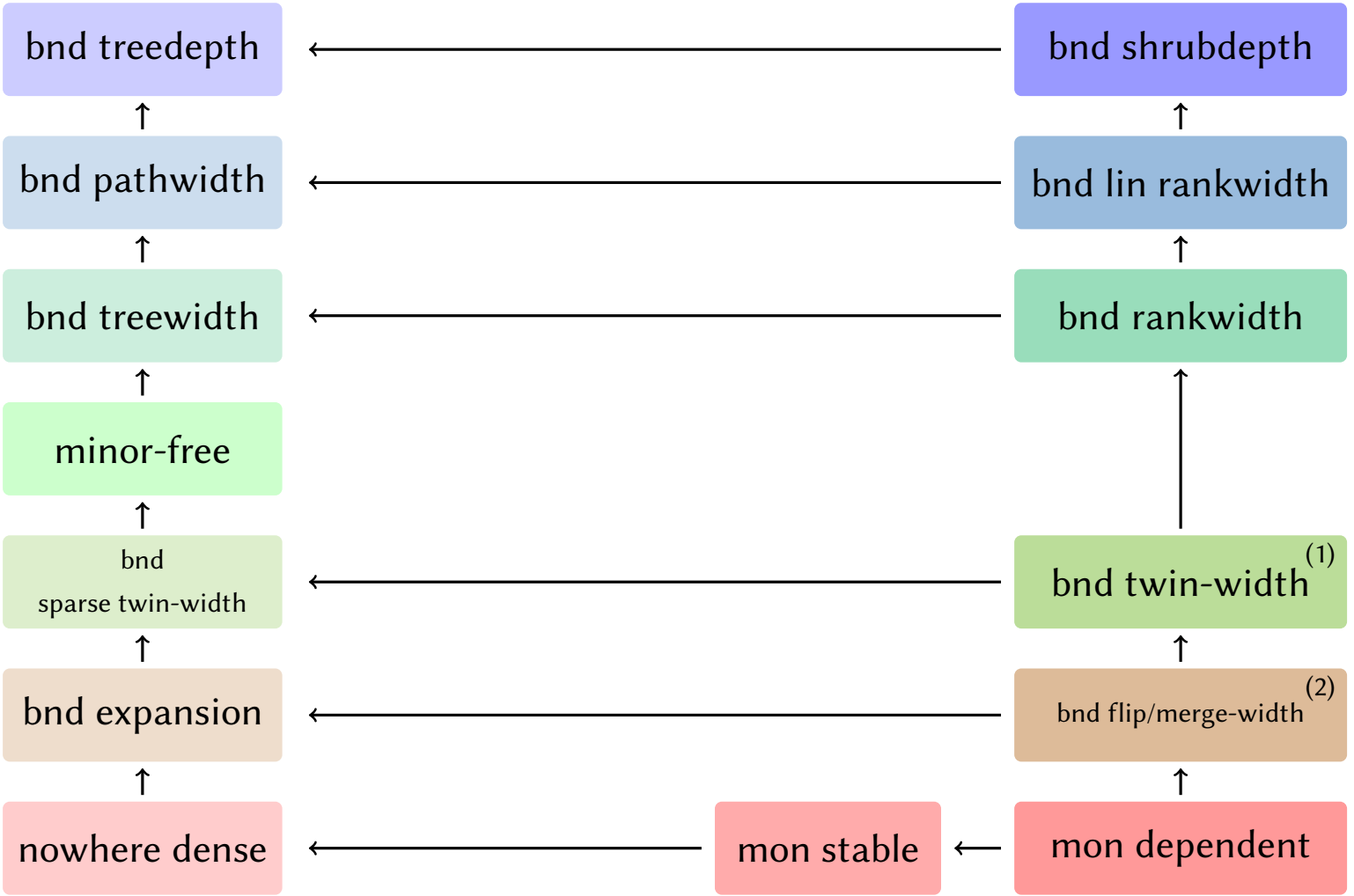
Suppose $\mathscr{C}$ is **monadically dependent**.

Then FO model-checking on $\mathscr{C}$ can be solved in time $\mathcal{O}_\varphi(n^c)$.

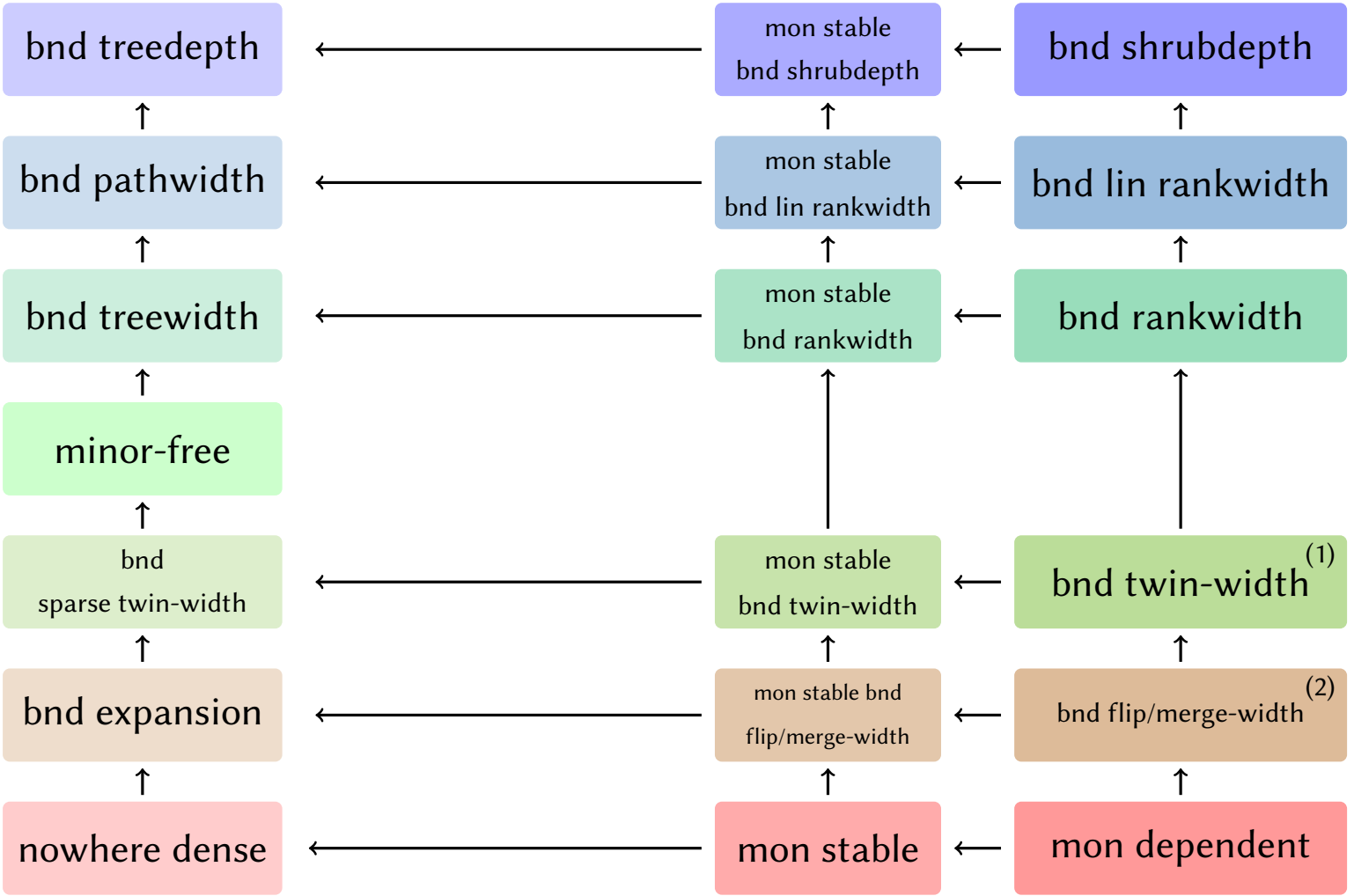
Theorem [Dreier, Eleftheriadis, Mählmann, McCarty, P, Toruńczyk; '23]

Suppose $\mathscr{C}$ is **monadically stable**.

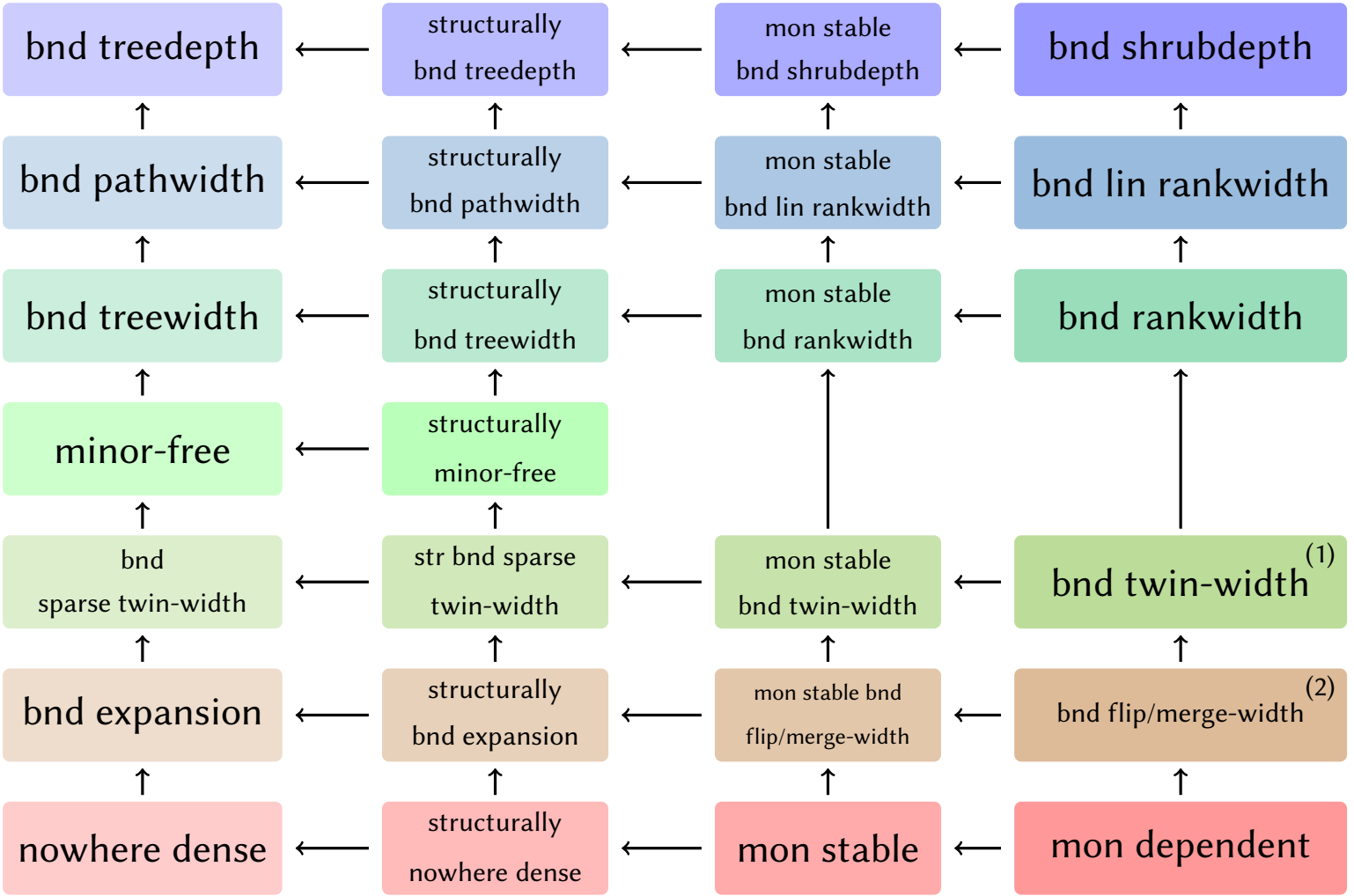
Then FO model-checking on $\mathscr{C}$ can be solved in time $\mathcal{O}_\varphi(n^6)$.



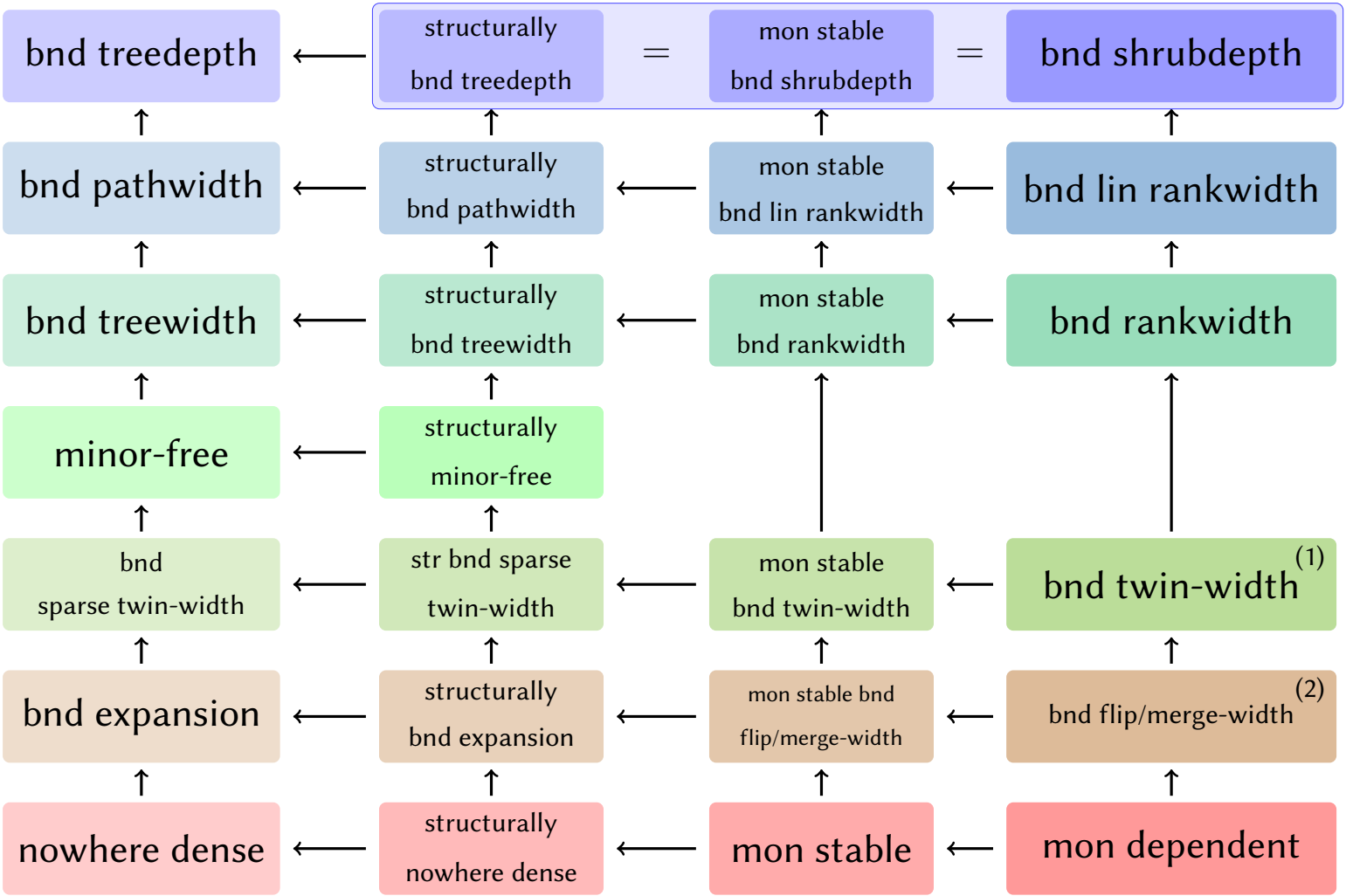
(1): Bonnet, Kim, Thomassé, Watrigant; '20
(2): Toruńczyk; '23 Dreier, Toruńczyk; '25^{+}



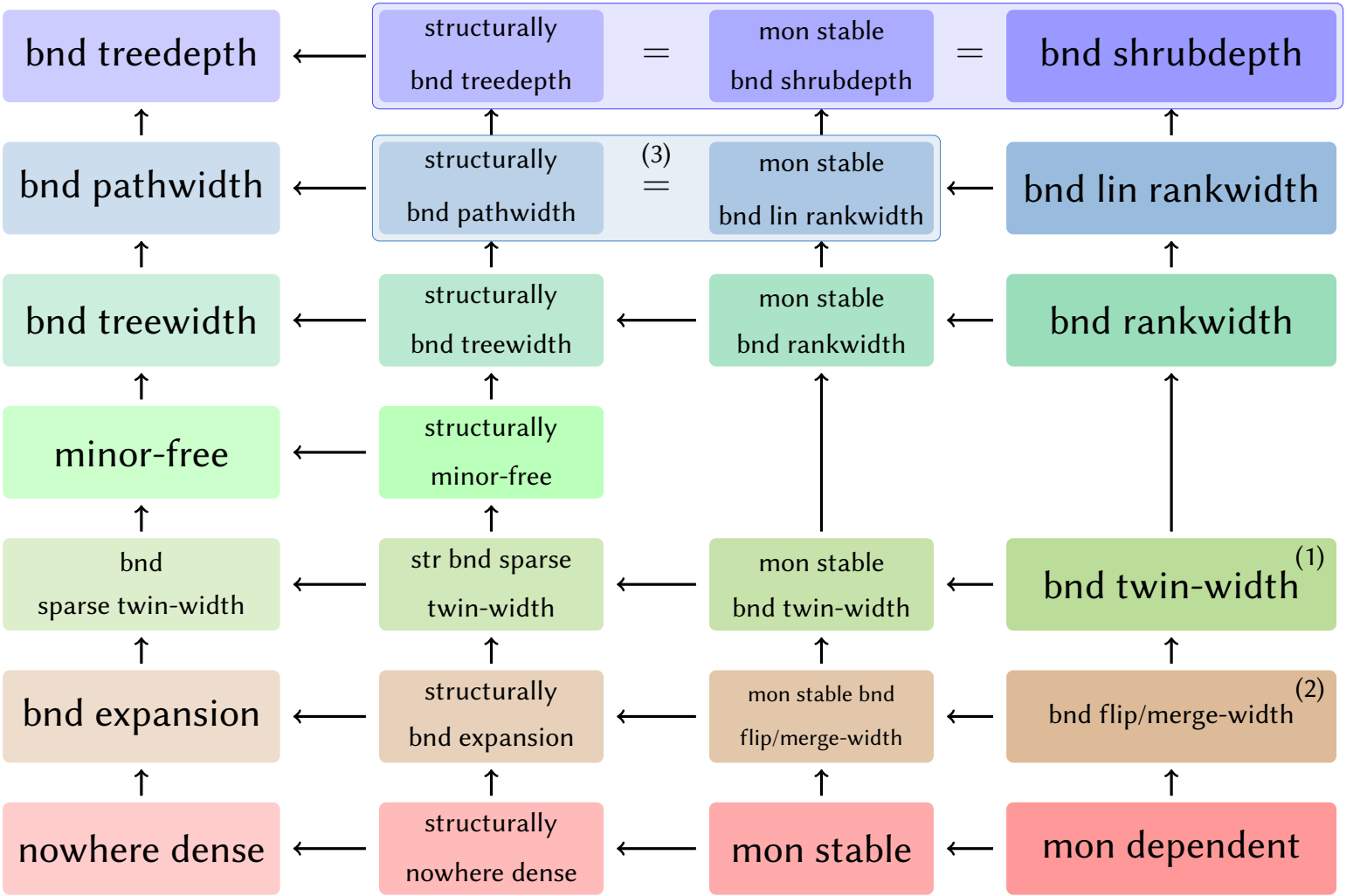
(1): Bonnet, Kim, Thomassé, Watrigant; ’20
(2): Toruńczyk; ’23 Dreier, Toruńczyk; ’25^{+}



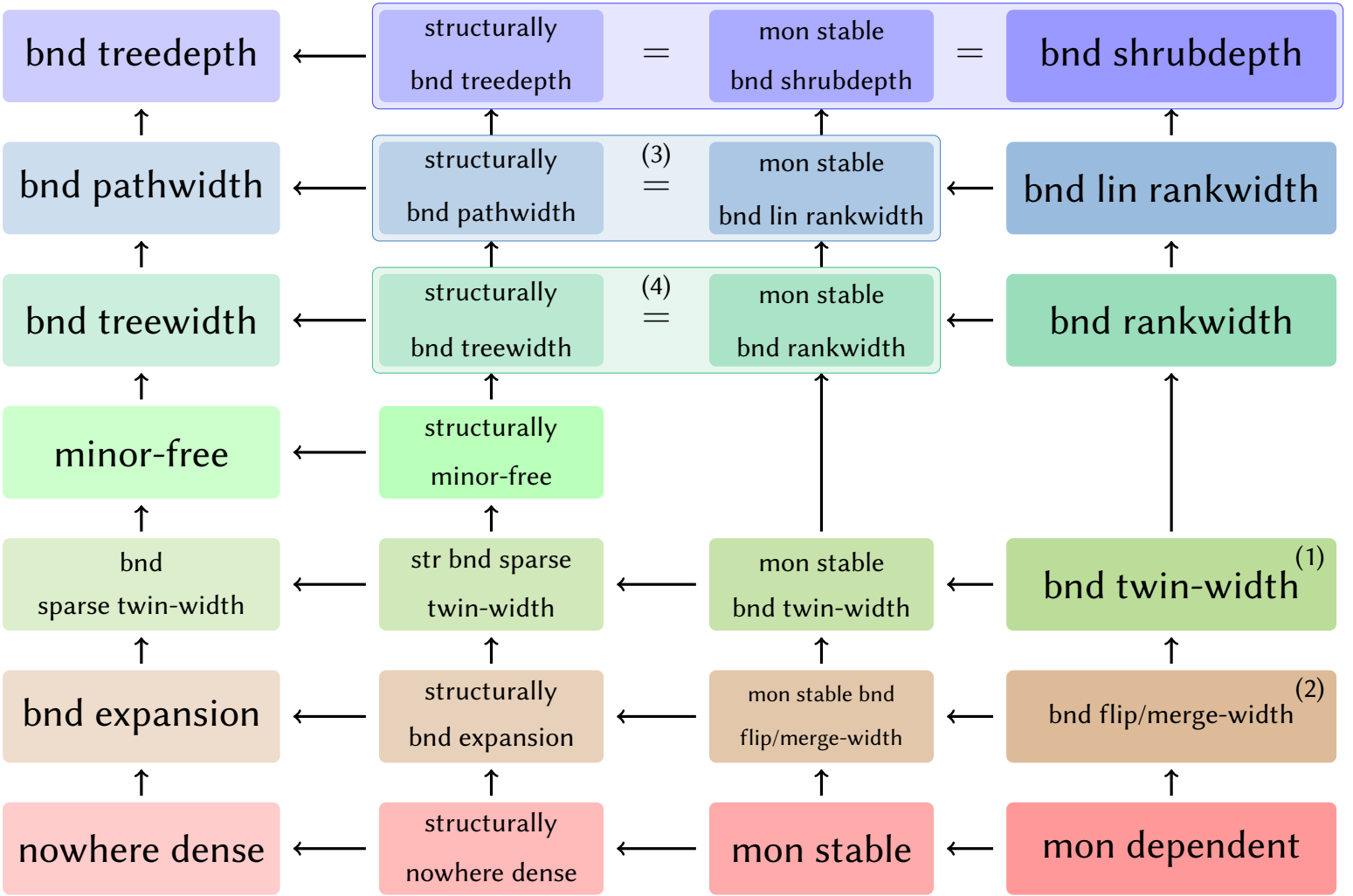
(1): Bonnet, Kim, Thomassé, Watrigant; ’20
(2): Toruńczyk; ’23 Dreier, Toruńczyk; ’25^{+}



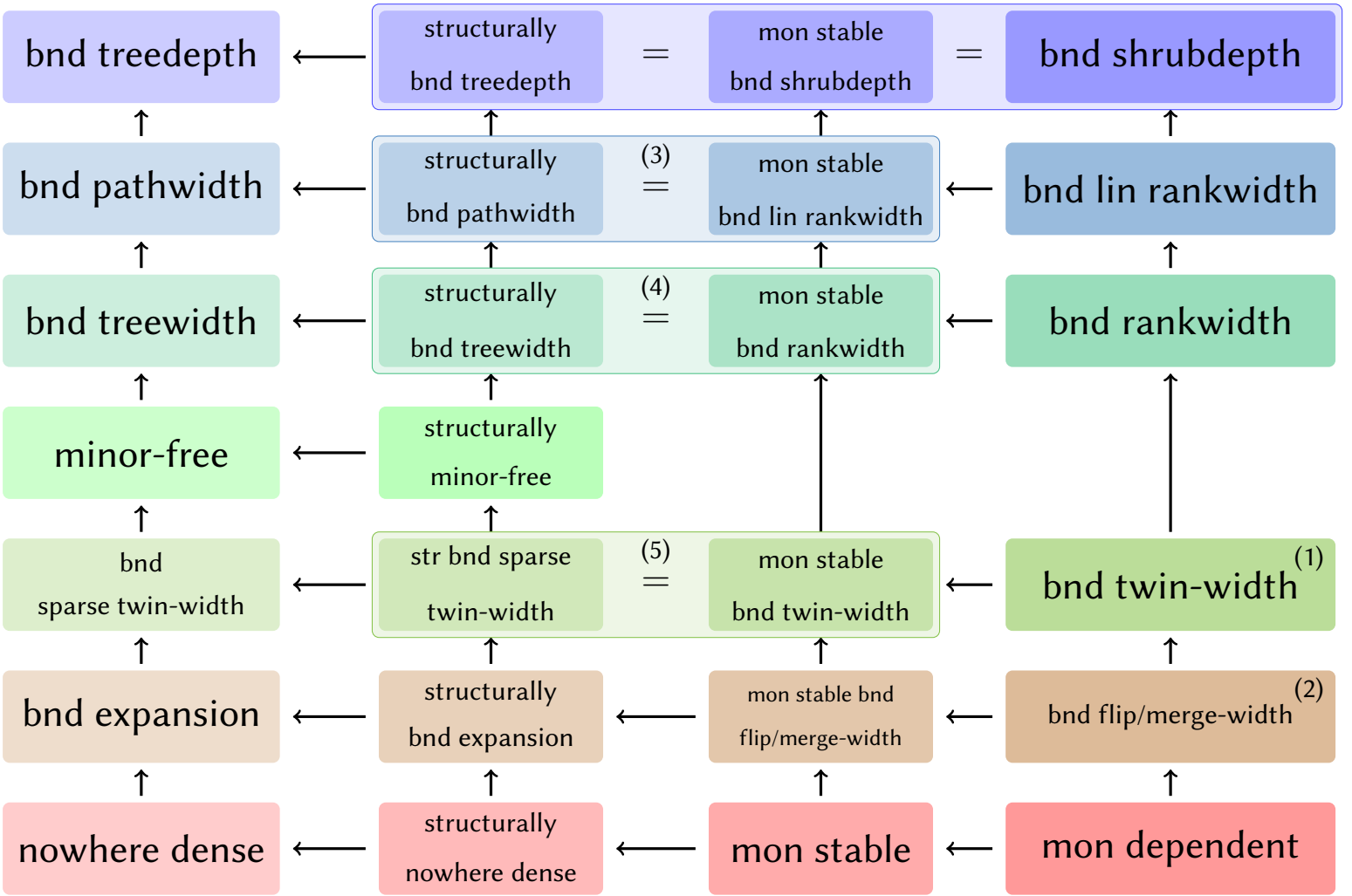
(1): Bonnet, Kim, Thomassé, Watrigant; ’20
(2): Toruńczyk; ’23 Dreier, Toruńczyk; ’25^{+}



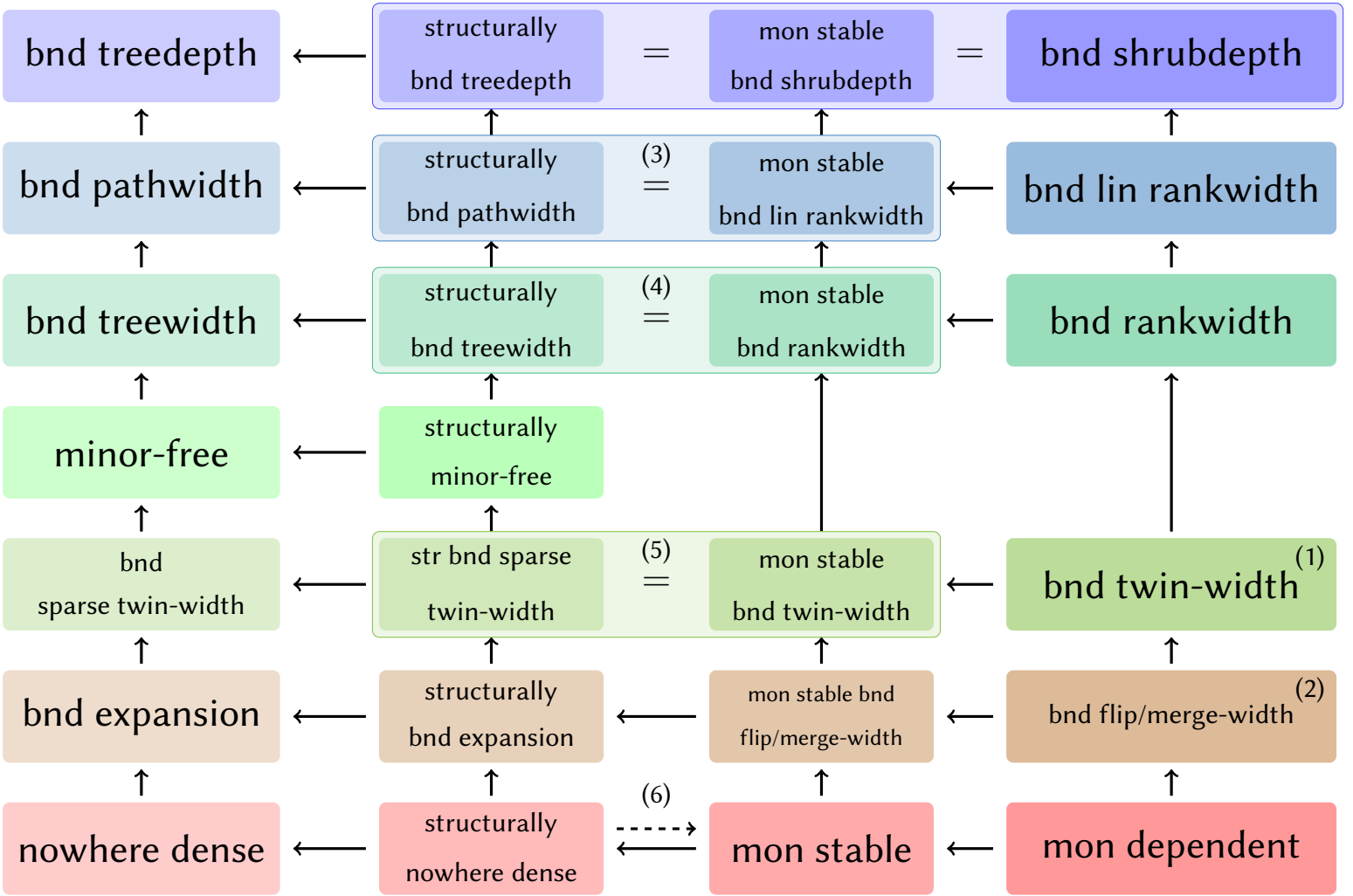
(1): Bonnet, Kim, Thomassé, Watrigant; '20 (3): Nešetřil, Ossona de Mendez, Rabinovich, Siebertz; '20
(2): Toruńczyk; '23 Dreier, Toruńczyk; '25^{+}



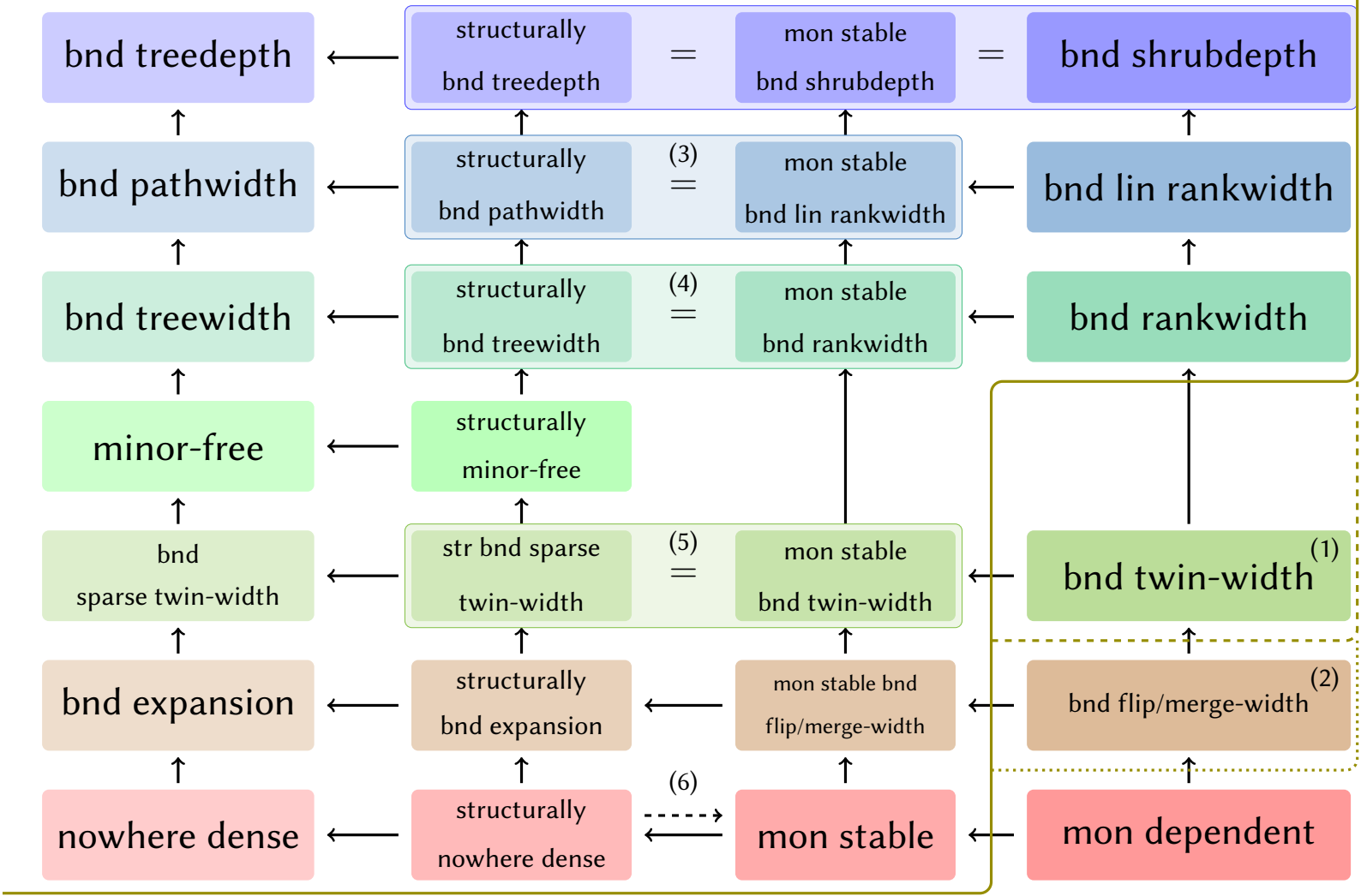
(1): Bonnet, Kim, Thomassé, Watrigant; '20
(2): Toruńczyk; '23 Dreier, Toruńczyk; '25^{+}
(3): Nešetřil, Ossona de Mendez, Rabinovich, Siebertz; '20
(4): Nešetřil, Ossona de Mendez, P, Rabinovich, Siebertz; '21



(1): Bonnet, Kim, Thomassé, Watrigant; '20
(2): Toruńczyk; '23 Dreier, Toruńczyk; '25+
(3): Nešetřil, Ossona de Mendez, Rabinovich, Siebertz; '20
(4): Nešetřil, Ossona de Mendez, P, Rabinovich, Siebertz; '21
(5): Gajarský, P, Toruńczyk; '22



(1): Bonnet, Kim, Thomassé, Watrigant; '20
(2): Toruńczyk; '23 Dreier, Toruńczyk; '25+
(3): Nešetřil, Ossona de Mendez, Rabinovich, Siebertz; '20
(4): Nešetřil, Ossona de Mendez, P, Rabinovich, Siebertz; '21
(5): Gajarský, P, Toruńczyk; '22
(6): Braunfeld, Nešetřil, Ossona de Mendez, Siebertz; '24+



(1): Bonnet, Kim, Thomassé, Watrigant; '20
(2): Toruńczyk; '23 Dreier, Toruńczyk; '25+
(3): Nešetřil, Ossona de Mendez, Rabinovich, Siebertz; '20
(4): Nešetřil, Ossona de Mendez, P, Rabinovich, Siebertz; '21
(5): Gajarský, P, Toruńczyk; '22
(6): Braunfeld, Nešetřil, Ossona de Mendez, Siebertz; '24+

Tools

mon stable

mon dependent

Tools

mon stable	mon dependent
Separators	
flip-flatness (1)	flip-breakability (2)

(1): Dreier, Mählmann, Toruńczyk; ’23
(2): Dreier, Mählmann, Toruńczyk; ’24

Tools

mon stable	mon dependent
Separators	
flip-flatness (1)	flip-breakability (2)
Decompositions	
Flipper Game (3)	?

(1): Dreier, Mählmann, Toruńczyk; ’23
(2): Dreier, Mählmann, Toruńczyk; ’24
(3): GMMcCOPPSST; ’23

Tools

mon stable		mon dependent
	Separators	
flip-flatness (1)		flip-breakability (2)
	Decompositions	
Flipper Game (3)		?
	Structure of neighborhoods	
VC density $1+\varepsilon$ (4)		
$\Rightarrow$ Welzl orders		
$\Rightarrow$ Neighborhood covers		

(1): Dreier, Mählmann, Toruńczyk; '23
(2): Dreier, Mählmann, Toruńczyk; '24
(3): GMMcCOPPSST; '23
(4): DEMMcCPT; '24

Tools

mon stable		mon dependent
Separators		
flip-flatness [1]		flip-breakability [2]
Decompositions		
Flipper Game [3]		?
Structure of neighborhoods		
VC density $1+\varepsilon$ [4]		VC density $1+\varepsilon$ [5]
$\Rightarrow$ Welzl orders		$\Rightarrow$ Welzl orders
$\Rightarrow$ Neighborhood covers		$\Rightarrow$ Neighborhood covers

(1): Dreier, Mählmann, Toruńczyk; '23
(2): Dreier, Mählmann, Toruńczyk; '24
(3): GMMcCOPPSST; '23
(4): DEMMcCPT; '24
(5): Dreier, Mählmann, McCarty, P, Toruńczyk; unpublished

Tools

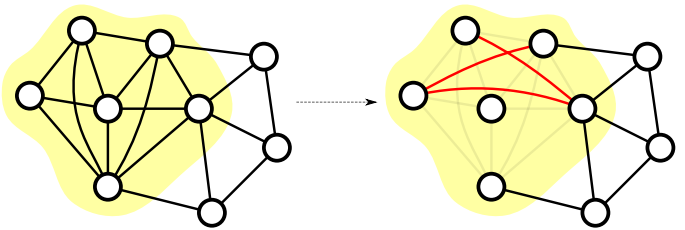
mon stable		mon dependent
Separators		
flip-flatness [1]		flip-breakability [2]
Decompositions		
Flipper Game [3]		?
Structure of neighborhoods		
VC density $1+\varepsilon$ [4]		VC density $1+\varepsilon$ [5]
$\Rightarrow$ Welzl orders		$\Rightarrow$ Welzl orders
$\Rightarrow$ Neighborhood covers		$\Rightarrow$ Neighborhood covers
Obstructions		
Patterns under flips [2,4]		Patterns under flips [2]
Shallow vertex-minors [6]		Shallow vertex-minors [6]

(1): Dreier, Mählmann, Toruńczyk; '23
(2): Dreier, Mählmann, Toruńczyk; '24
(3): GMMcCOPPSST; '23
(4): DEMMcCPT; '24
(5): Dreier, Mählmann, McCarty, P, Toruńczyk; unpublished
(6): Buffière, Kim, Ossona de Mendez; '24

Flip-flatness and flip-breakability

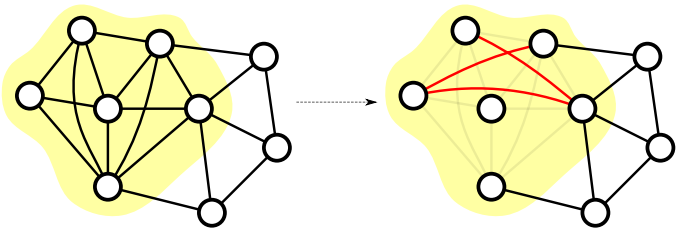
Flip-flatness and flip-breakability

Flip/Perturbation: Complement the edge relation on a vertex subset.

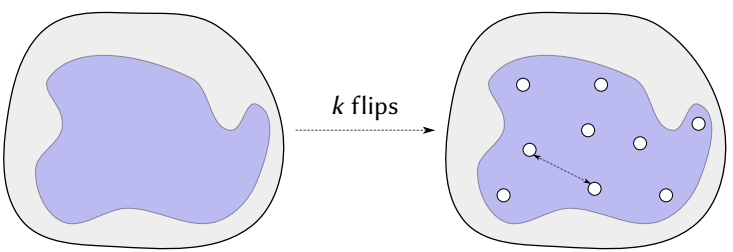


Flip-flatness and flip-breakability

Flip/Perturbation: Complement the edge relation on a vertex subset.

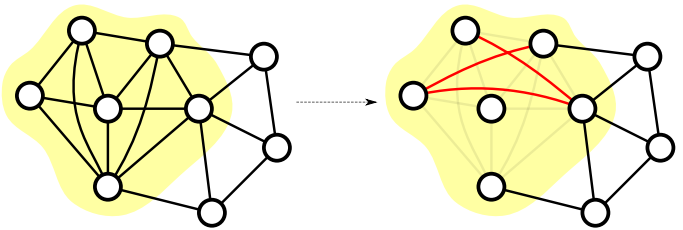


Def: $\mathscr{C}$ is **flip-flat** if $\forall d\, \exists k$ so that $\forall G \in \mathscr{C}\, \forall W \subseteq V(G)$, there is $A \subseteq W$ of size $\mathcal{U}(|W|)$ that can be made d-scattered by applying k flips.



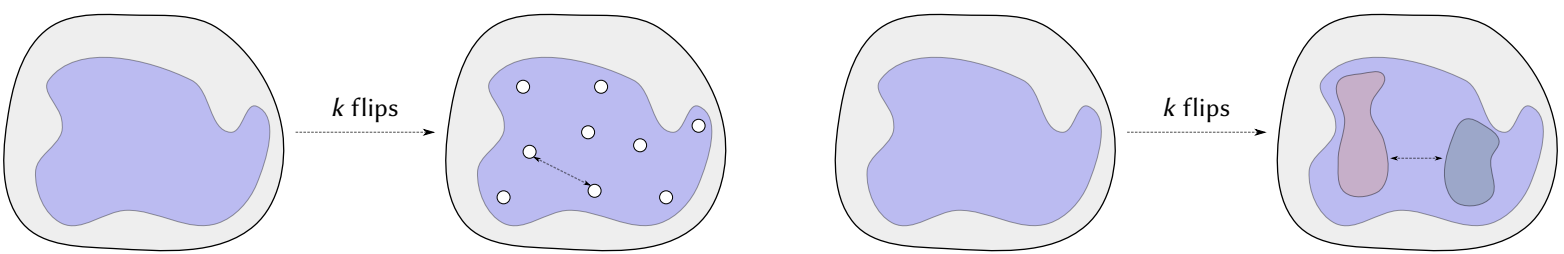
Flip-flatness and flip-breakability

Flip/Perturbation: Complement the edge relation on a vertex subset.



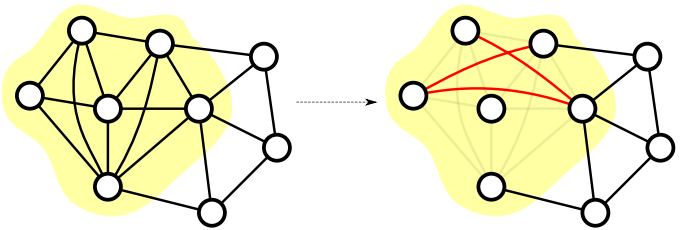
Def: $\mathscr{C}$ is **flip-flat** if $\forall d\, \exists k$ so that $\forall G \in \mathscr{C}\, \forall W \subseteq V(G)$, there is $A \subseteq W$ of size $\mathcal{U}(|W|)$ that can be made d-scattered by applying k flips.

Def: $\mathscr{C}$ is **flip-breakable** if $\forall d\, \exists k$ so that $\forall G \in \mathscr{C}\, \forall W \subseteq V(G)$, there are $A, B \subseteq W$ of size $\mathcal{U}(|W|)$, so that $\mathrm{dist}(A, B) > d$ after applying k flips.



Flip-flatness and flip-breakability

Flip/Perturbation: Complement the edge relation on a vertex subset.



Def: $\mathscr{C}$ is **flip-flat** if $\forall d\, \exists k$ so that $\forall G \in \mathscr{C}\, \forall W \subseteq V(G)$, there is $A \subseteq W$ of size $\mathcal{U}(|W|)$ that can be made d-scattered by applying k flips.

Def: $\mathscr{C}$ is **flip-breakable** if $\forall d\, \exists k$ so that $\forall G \in \mathscr{C}\, \forall W \subseteq V(G)$, there are $A, B \subseteq W$ of size $\mathcal{U}(|W|)$, so that $\mathrm{dist}(A, B) > d$ after applying k flips.

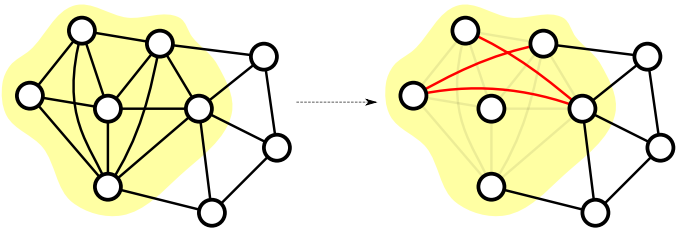
Theorem [Dreier, Mählmann, Toruńczyk; '23, '24]

Monadic stability $\Leftrightarrow$ Flip-flatness

Monadic dependence $\Leftrightarrow$ Flip-breakability

Flip-flatness and flip-breakability

Flip/Perturbation: Complement the edge relation on a vertex subset.



Def: $\mathscr{C}$ is **flip-flat** if $\forall d\, \exists k$ so that $\forall G \in \mathscr{C}\, \forall W \subseteq V(G)$, there is $A \subseteq W$ of size $\mathcal{U}(|W|)$ that can be made d-scattered by applying k flips.

Def: $\mathscr{C}$ is **flip-breakable** if $\forall d\, \exists k$ so that $\forall G \in \mathscr{C}\, \forall W \subseteq V(G)$, there are $A, B \subseteq W$ of size $\mathcal{U}(|W|)$, so that $\mathrm{dist}(A, B) > d$ after applying k flips.

Theorem [Dreier, Mählmann, Toruńczyk; '23, '24]

Monadic stability $\Leftrightarrow$ Flip-flatness

Monadic dependence $\Leftrightarrow$ Flip-breakability

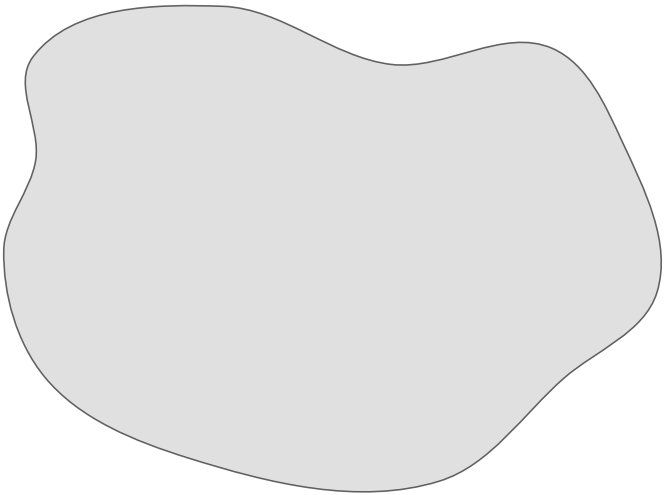
Fact: Flatness $\Leftrightarrow$ Breakability $\Leftrightarrow$ Nowhere denseness,

where **flips** are replaced with **vertex deletions**.

Flipper game

Flipper game

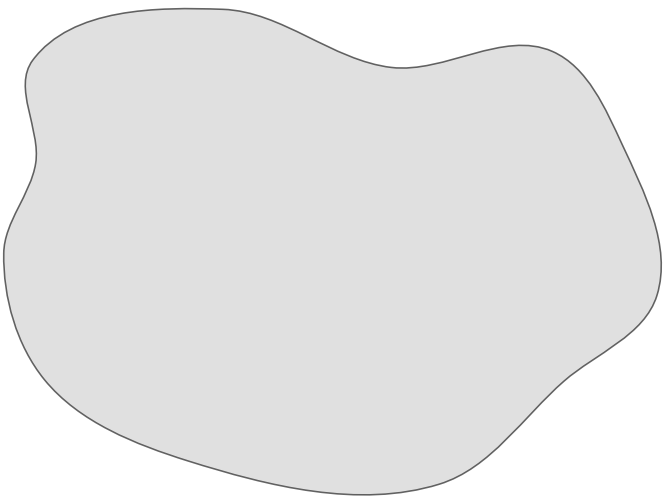
Fixed radius $d \in \mathbb{N}$. Played on a graph G.



Flipper game

Fixed radius $d \in \mathbb{N}$. Played on a graph G.

Two players: **Flipper** and **Connector**

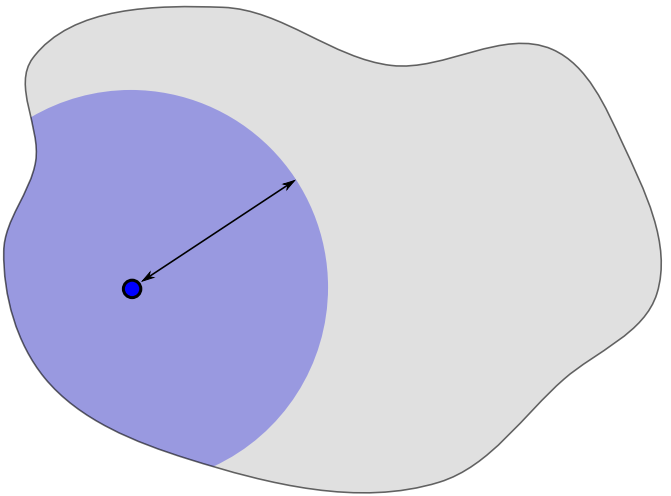


Round

Flipper game

Fixed radius $d \in \mathbb{N}$. Played on a graph G.

Two players: **Flipper** and **Connector**



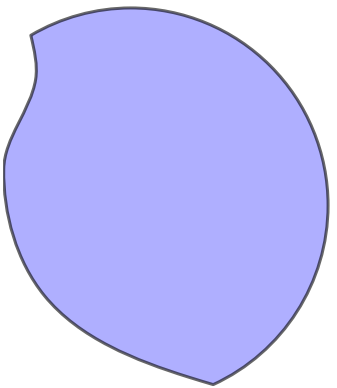
Round

Connector picks a ball of radius d.

Flipper game

Fixed radius $d \in \mathbb{N}$. Played on a graph G.

Two players: **Flipper** and **Connector**



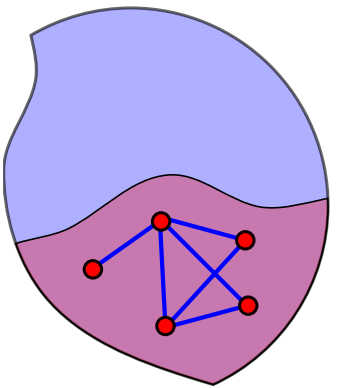
Round

Connector picks a ball of radius d. $\rightsquigarrow$ The arena is **restricted** to this ball.

Flipper game

Fixed radius $d \in \mathbb{N}$. Played on a graph G.

Two players: **Flipper** and **Connector**



Round

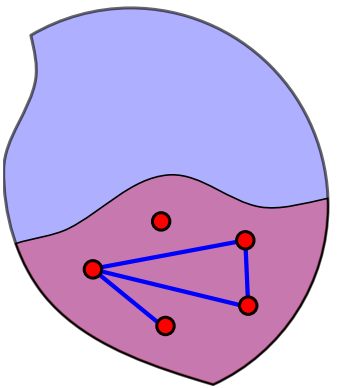
Connector picks a ball of radius d. $\rightsquigarrow$ The arena is **restricted** to this ball.

Flipper chooses any **vertex subset**.

Flipper game

Fixed radius $d \in \mathbb{N}$. Played on a graph G.

Two players: **Flipper** and **Connector**



Round

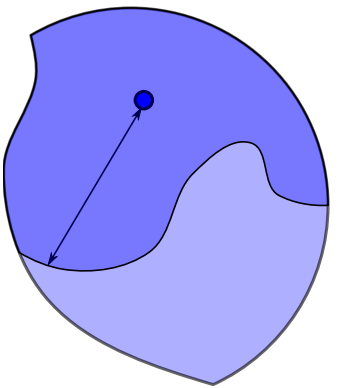
Connector picks a ball of radius d. $\rightsquigarrow$ The arena is **restricted** to this ball.

Flipper chooses any **vertex subset**. $\rightsquigarrow$ A **flip** is applied to this subset.

Flipper game

Fixed radius $d \in \mathbb{N}$. Played on a graph G.

Two players: **Flipper** and **Connector**



Round

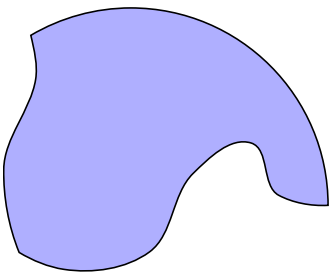
Connector picks a ball of radius d. $\rightsquigarrow$ The arena is **restricted** to this ball.

Flipper chooses any **vertex subset**. $\rightsquigarrow$ A **flip** is applied to this subset.

Flipper game

Fixed radius $d \in \mathbb{N}$. Played on a graph G.

Two players: **Flipper** and **Connector**



Round

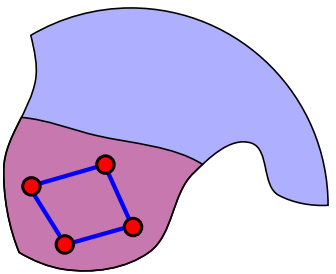
Connector picks a ball of radius d. $\rightsquigarrow$ The arena is **restricted** to this ball.

Flipper chooses any **vertex subset**. $\rightsquigarrow$ A **flip** is applied to this subset.

Flipper game

Fixed radius $d \in \mathbb{N}$. Played on a graph G.

Two players: **Flipper** and **Connector**



Round

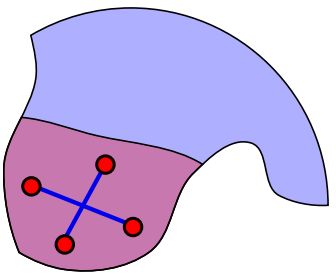
Connector picks a ball of radius d. $\rightsquigarrow$ The arena is **restricted** to this ball.

Flipper chooses any **vertex subset**. $\rightsquigarrow$ A **flip** is applied to this subset.

Flipper game

Fixed radius $d \in \mathbb{N}$. Played on a graph G.

Two players: **Flipper** and **Connector**



Round

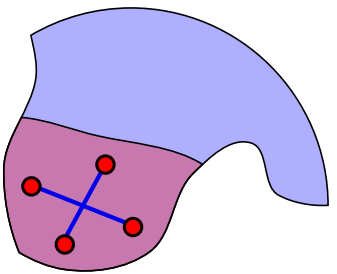
Connector picks a ball of radius d. $\rightsquigarrow$ The arena is **restricted** to this ball.

Flipper chooses any **vertex subset**. $\rightsquigarrow$ A **flip** is applied to this subset.

Flipper game

Fixed radius $d \in \mathbb{N}$. Played on a graph G.

Two players: **Flipper** and **Connector**



Round

Connector picks a ball of radius d. $\rightsquigarrow$ The arena is **restricted** to this ball.

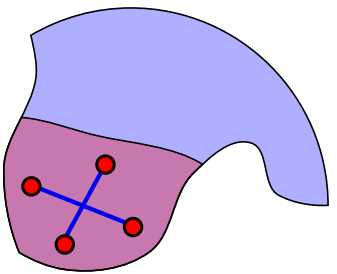
Flipper chooses any **vertex subset**. $\rightsquigarrow$ A **flip** is applied to this subset.

Game **finishes** when the arena gets reduced to **one vertex**.

Flipper game

Fixed radius $d \in \mathbb{N}$. Played on a graph G.

Two players: **Flipper** and **Connector**



Round

Connector picks a ball of radius d. $\rightsquigarrow$ The arena is **restricted** to this ball.

Flipper chooses any **vertex subset**. $\rightsquigarrow$ A **flip** is applied to this subset.

Game **finishes** when the arena gets reduced to **one vertex**.

Flipper: finish the game quickly **Connector**: stay alive

Flipper game and monadic stability

Flipper game and monadic stability

Splitter Game: Same, but Splitter deletes a vertex in every round.

Flipper game and monadic stability

Splitter Game: Same, but Splitter deletes a vertex in every round.

Theorem [Grohe, Kreutzer, Siebertz; '14]

$\mathscr{C}$ is **nowhere dense**. $\Leftrightarrow$

For every $d \in \mathbb{N}$, there is $k \in \mathbb{N}$ such that **Splitter** can win the radius-d Splitter game in any $G \in \mathscr{C}$ in $\leqslant k$ rounds.

Flipper game and monadic stability

Splitter Game: Same, but Splitter deletes a vertex in every round.

Theorem [Grohe, Kreutzer, Siebertz; '14]

$\mathscr{C}$ is **nowhere dense**. $\Leftrightarrow$

For every $d \in \mathbb{N}$, there is $k \in \mathbb{N}$ such that **Splitter** can win the radius-d Splitter game in any $G \in \mathscr{C}$ in $\leqslant k$ rounds.

Theorem [Gajarský, Mählmann, McCarty, Ohlmann, P, Przybyszewski, Siebertz, Sokołowski, Toruńczyk; '23]

$\mathscr{C}$ is **mon stable**. $\Leftrightarrow$

For every $d \in \mathbb{N}$, there is $k \in \mathbb{N}$ such that **Flipper** can win the radius-d Flipper game in any $G \in \mathscr{C}$ in $\leqslant k$ rounds.

Flipper game and monadic stability

Splitter Game: Same, but Splitter deletes a vertex in every round.

Theorem [Grohe, Kreutzer, Siebertz; '14]

$\mathscr{C}$ is **nowhere dense**. $\Leftrightarrow$

For every $d \in \mathbb{N}$, there is $k \in \mathbb{N}$ such that **Splitter** can win the radius-d Splitter game in any $G \in \mathscr{C}$ in $\leqslant k$ rounds.

Theorem [Gajarský, Mählmann, McCarty, Ohlmann, P, Przybyszewski, Siebertz, Sokołowski, Toruńczyk; '23]

$\mathscr{C}$ is **mon stable**. $\Leftrightarrow$

For every $d \in \mathbb{N}$, there is $k \in \mathbb{N}$ such that **Flipper** can win the radius-d Flipper game in any $G \in \mathscr{C}$ in $\leqslant k$ rounds.

Game tree $=$ **Shallow decomposition**

Flipper game and monadic stability

Splitter Game: Same, but Splitter deletes a vertex in every round.

Theorem [Grohe, Kreutzer, Siebertz; '14]

$\mathscr{C}$ is **nowhere dense**. $\Leftrightarrow$

For every $d \in \mathbb{N}$, there is $k \in \mathbb{N}$ such that **Splitter** can win the radius-d Splitter game in any $G \in \mathscr{C}$ in $\leqslant k$ rounds.

Theorem [Gajarský, Mählmann, McCarty, Ohlmann, P, Przybyszewski, Siebertz, Sokołowski, Toruńczyk; '23]

$\mathscr{C}$ is **mon stable**. $\Leftrightarrow$

For every $d \in \mathbb{N}$, there is $k \in \mathbb{N}$ such that **Flipper** can win the radius-d Flipper game in any $G \in \mathscr{C}$ in $\leqslant k$ rounds.

Game tree $=$ **Shallow decomposition**

vertex deletion $\rightsquigarrow$ **flip**

Structure of neighborhoods

Structure of neighborhoods

Theorem [Dreier, Eleftheriadis, Mählmann, McCarty, P, Toruńczyk; '24]

$\mathscr{C}$ **mon stable**. Then for all $G \in \mathscr{C}$, $A \subseteq V(G)$, and $\varepsilon > 0$, we have

$$|\{N[v] \cap A \colon v \in V(G)\}| \leqslant \mathcal{O}_\varepsilon(|A|^{1+\varepsilon}).$$

Structure of neighborhoods

Theorem [Dreier, Eleftheriadis, Mählmann, McCarty, P, Toruńczyk; '24]

$\mathscr{C}$ **mon stable**. Then for all $G \in \mathscr{C}$, $A \subseteq V(G)$, and $\varepsilon > 0$, we have

$$|\{N[v] \cap A \colon v \in V(G)\}| \leqslant \mathcal{O}_\varepsilon(|A|^{1+\varepsilon}).$$

Theorem [follows from Welzl '88]

$\mathscr{C}$ **mon stable**. Then every $G \in \mathscr{C}$ admits a vertex ordering σ such that:

for every vertex v, $N[v]$ breaks into $\mathcal{O}_\varepsilon(n^\varepsilon)$ intervals in σ.

Structure of neighborhoods

Theorem [Dreier, Eleftheriadis, Mählmann, McCarty, P, Toruńczyk; '24]

$\mathscr{C}$ **mon stable**. Then for all $G \in \mathscr{C}$, $A \subseteq V(G)$, and $\varepsilon > 0$, we have

$$|\{N[v] \cap A \colon v \in V(G)\}| \leqslant \mathcal{O}_\varepsilon(|A|^{1+\varepsilon}).$$

Theorem [follows from Welzl '88]

$\mathscr{C}$ **mon stable**. Then every $G \in \mathscr{C}$ admits a vertex ordering σ such that:

for every vertex v, $N[v]$ breaks into $\mathcal{O}_\varepsilon(n^\varepsilon)$ intervals in σ.

Theorem [Dreier, Eleftheriadis, Mählmann, McCarty, P, Toruńczyk; '24]

$\mathscr{C}$ **mon stable**. Then for every $G \in \mathscr{C}$ and $d \in \mathbb{N}$, there is $\mathcal{F} \subseteq 2^{V(G)}$ s.t.:

- every $A \in \mathcal{F}$ has weak diameter $\leqslant 4d$;
- every radius-d ball B is contained in some $A \in \mathcal{F}$;
- every vertex v belongs to $\mathcal{O}_\varepsilon(n^\varepsilon)$ sets $A \in \mathcal{F}$.

Implementing model-checking

Implementing model-checking

Goal: Check whether φ holds in G.

Implemented in [Dreier, Mählmann, Siebertz; '23],
inspired by the strategy from [Grohe, Kreutzer, Siebertz; '14].

Implementing model-checking

Goal: Check whether φ holds in G.

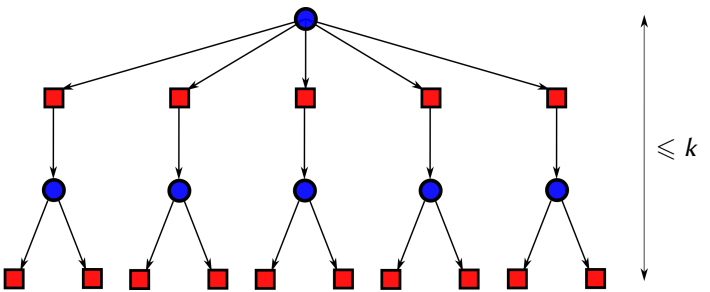
Implemented in [Dreier, Mählmann, Siebertz; '23],
inspired by the strategy from [Grohe, Kreutzer, Siebertz; '14].

Implementing model-checking

Goal: Check whether φ holds in G.

Implemented in [Dreier, Mählmann, Siebertz; '23], inspired by the strategy from [Grohe, Kreutzer, Siebertz; '14].

Pick $d := 2^{\mathcal{O}(\mathrm{qr}(\varphi))}$ and consider the radius-d Game on G, viewed by Flipper.

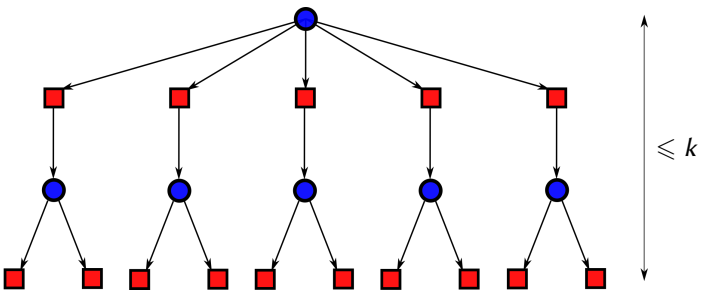


Implementing model-checking

Goal: Check whether φ holds in G.

Implemented in [Dreier, Mählmann, Siebertz; '23], inspired by the strategy from [Grohe, Kreutzer, Siebertz; '14].

Pick $d := 2^{\mathcal{O}(\mathrm{qr}(\varphi))}$ and consider the radius-d Game on G, viewed by Flipper.



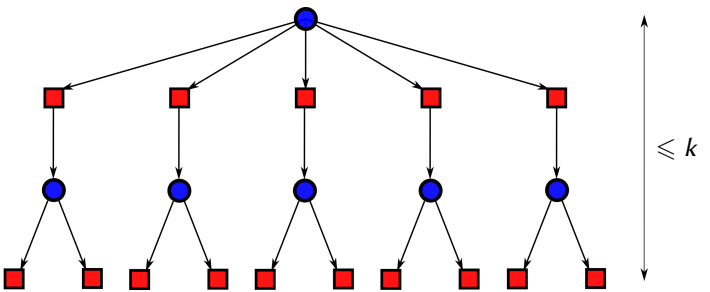
Nodes $\rightsquigarrow$ Induced subgraphs under some flips.

Implementing model-checking

Goal: Check whether φ holds in G.

Implemented in [Dreier, Mählmann, Siebertz; '23], inspired by the strategy from [Grohe, Kreutzer, Siebertz; '14].

Pick $d := 2^{\mathcal{O}(\mathrm{qr}(\varphi))}$ and consider the radius-d Game on G, viewed by Flipper.



Nodes $\rightsquigarrow$ Induced subgraphs under some flips.

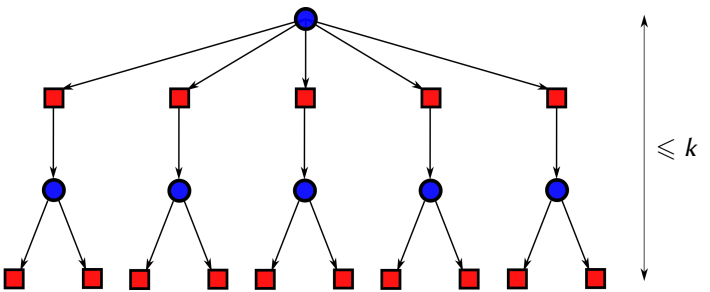
Locality: Type of a node can be computed from **types** of children.

Implementing model-checking

Goal: Check whether φ holds in G.

Implemented in [Dreier, Mählmann, Siebertz; '23], inspired by the strategy from [Grohe, Kreutzer, Siebertz; '14].

Pick $d := 2^{\mathcal{O}(\mathrm{qr}(\varphi))}$ and consider the radius-d Game on G, viewed by Flipper.



Nodes $\rightsquigarrow$ Induced subgraphs under some flips.

Locality: **Type** of a node can be computed from **types** of children.

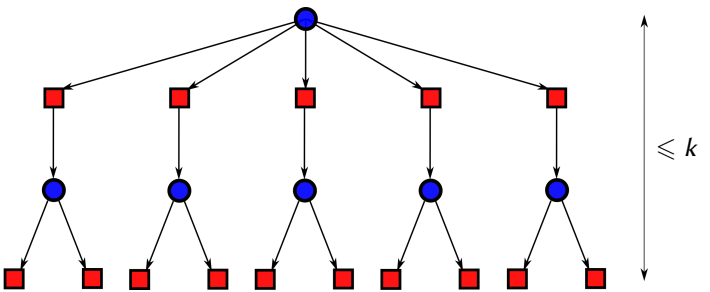
Idea: Compute the types bottom-up.

Implementing model-checking

Goal: Check whether φ holds in G.

Implemented in [Dreier, Mählmann, Siebertz; '23], inspired by the strategy from [Grohe, Kreutzer, Siebertz; '14].

Pick $d := 2^{\mathcal{O}(\mathrm{qr}(\varphi))}$ and consider the radius-d Game on G, viewed by Flipper.



Nodes $\rightsquigarrow$ Induced subgraphs under some flips.

Locality: **Type** of a node can be computed from **types** of children.

Idea: Compute the types bottom-up.

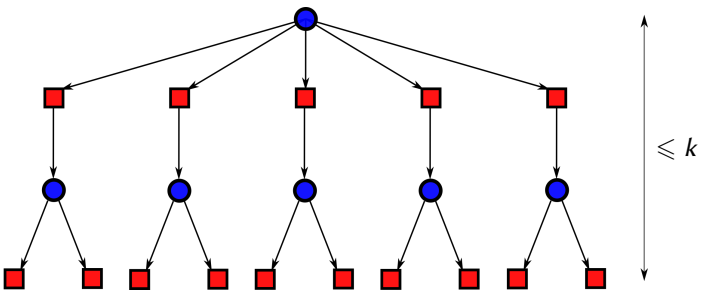
Problem: The tree can be as large as n^k.

Implementing model-checking

Goal: Check whether φ holds in G.

Implemented in [Dreier, Mählmann, Siebertz; '23], inspired by the strategy from [Grohe, Kreutzer, Siebertz; '14].

Pick $d := 2^{\mathcal{O}(\mathrm{qr}(\varphi))}$ and consider the radius-d Game on G, viewed by Flipper.



Nodes $\rightsquigarrow$ Induced subgraphs under some flips.

Locality: **Type** of a node can be computed from **types** of children.

Idea: Compute the types bottom-up.

Problem: The tree can be as large as n^k.

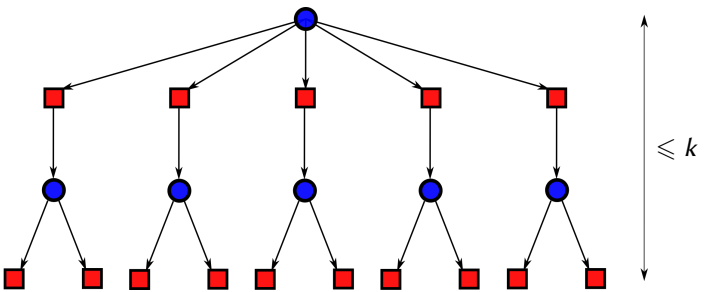
Fix: Restrict the Connector's moves to radius-d cover.

Implementing model-checking

Goal: Check whether φ holds in G.

Implemented in [Dreier, Mählmann, Siebertz; '23], inspired by the strategy from [Grohe, Kreutzer, Siebertz; '14].

Pick $d := 2^{\mathcal{O}(\mathrm{qr}(\varphi))}$ and consider the radius-d Game on G, viewed by Flipper.



Nodes $\rightsquigarrow$ Induced subgraphs under some flips.

Locality: **Type** of a node can be computed from **types** of children.

Idea: Compute the types bottom-up.

Problem: The tree can be as large as n^k.

Fix: Restrict the Connector's moves to radius-d cover.

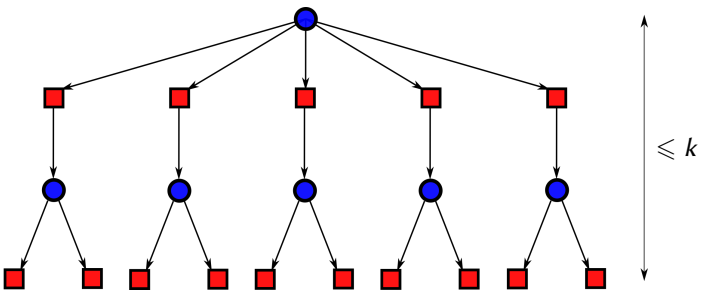
radius-d Game $\leqslant$ Game on radius-d cover $\leqslant$ radius-$4d$ Game

Implementing model-checking

Goal: Check whether φ holds in G.

Implemented in [Dreier, Mählmann, Siebertz; '23], inspired by the strategy from [Grohe, Kreutzer, Siebertz; '14].

Pick $d := 2^{\mathcal{O}(\mathrm{qr}(\varphi))}$ and consider the radius-d Game on G, viewed by Flipper.



Nodes $\rightsquigarrow$ Induced subgraphs under some flips.

Locality: Type of a node can be computed from **types** of children.

Idea: Compute the types bottom-up.

Problem: The tree can be as large as n^k.

Fix: Restrict the Connector's moves to radius-d cover.

radius-d Game $\leqslant$ Game on radius-d cover $\leqslant$ radius-$4d$ Game

Now the game tree has size $\mathcal{O}_\varepsilon(n^{1+\varepsilon})$.

Forbidden patterns

Forbidden patterns

Theorem [Dreier, Mählmann, Toruńczyk; '24]

$\mathscr{C}$ is **mon dependent** iff there are no $r, k \in \mathbb{N}$ s.t. $\mathscr{C}$ contains induced:

- a k-flip of every star r-crossing; or
- a k-flip of every clique r-crossing; or
- a k-flip of every half-graph r-crossing; or
- a k-flip of every comparability grid.

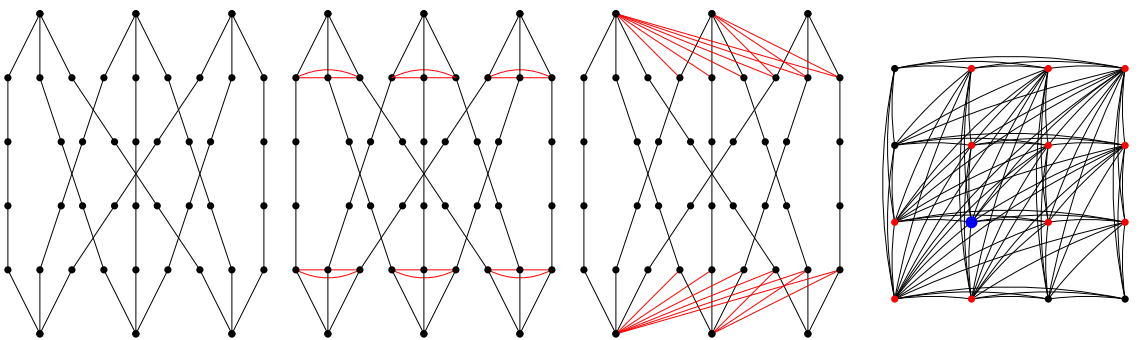


Figure by Niko Mählmann

Forbidden patterns

Theorem [Dreier, Mählmann, Toruńczyk; '24]

$\mathscr{C}$ is **mon dependent** iff there are no $r, k \in \mathbb{N}$ s.t. $\mathscr{C}$ contains induced:

- a k-flip of every star r-crossing; or
- a k-flip of every clique r-crossing; or
- a k-flip of every half-graph r-crossing; or
- a k-flip of every comparability grid.

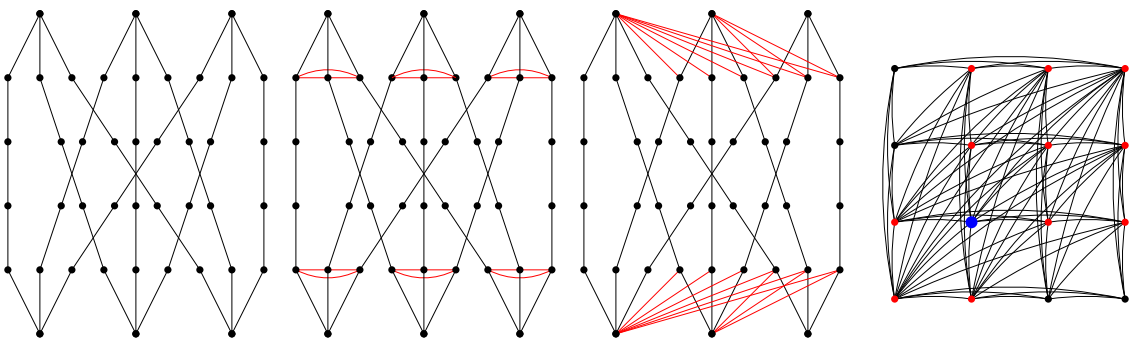


Figure by Niko Mählmann

Cor: $\mathscr{C}$ **mon independent** and **hereditary** $\Rightarrow$ FO-MC is AW[$\star$]-hard.

Forbidden patterns

Theorem [Dreier, Mählmann, Toruńczyk; '24]

$\mathscr{C}$ is **mon dependent** iff there are no $r, k \in \mathbb{N}$ s.t. $\mathscr{C}$ contains induced:

- a k-flip of every star r-crossing; or
- a k-flip of every clique r-crossing; or
- a k-flip of every half-graph r-crossing; or
- a k-flip of every comparability grid.

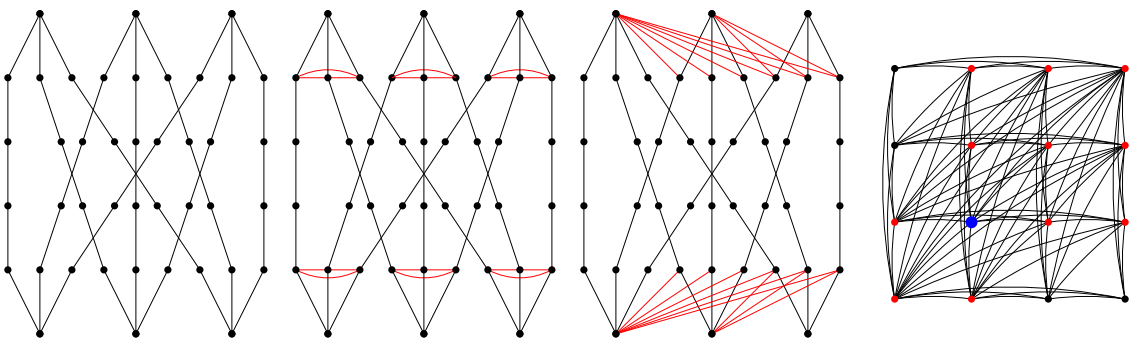


Figure by Niko Mählmann

Cor: $\mathscr{C}$ **mon independent** and **hereditary** $\Rightarrow$ FO-MC is AW[$\star$]-hard.

Fact: For **mon stable**: star crossings, clique crossings, and half-graphs.

Conclusions

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

– Basic local separability using flips.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

– Basic local separability using flips.

Mon stability: being unordered.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

– Basic local separability using flips.

Mon stability: being unordered.

– Bounded-depth decompositions by alternating flips and localization.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

– Basic local separability using flips.

Mon stability: being unordered.

– Bounded-depth decompositions by alternating flips and localization.

Goal: Model-checking FO is fpt on **mon dependent** classes.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

– Basic local separability using flips.

Mon stability: being unordered.

– Bounded-depth decompositions by alternating flips and localization.

Goal: Model-checking FO is fpt on **mon dependent** classes.

– Achieved on **mon stable** graph classes.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

- Basic local separability using flips.

Mon stability: being unordered.

- Bounded-depth decompositions by alternating flips and localization.

Goal: Model-checking FO is fpt on **mon dependent** classes.

- Achieved on **mon stable** graph classes.
- Main hurdle on **mon dependent**: **decomposition**.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

- Basic local separability using flips.

Mon stability: being unordered.

- Bounded-depth decompositions by alternating flips and localization.

Goal: Model-checking FO is fpt on **mon dependent** classes.

- Achieved on **mon stable** graph classes.
- Main hurdle on **mon dependent**: **decomposition**.

A lot of other **avenues** in the theory.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

– Basic local separability using flips.

Mon stability: being unordered.

– Bounded-depth decompositions by alternating flips and localization.

Goal: Model-checking FO is fpt on **mon dependent** classes.

– Achieved on **mon stable** graph classes.

– Main hurdle on **mon dependent**: **decomposition**.

A lot of other **avenues** in the theory.

– Obstruction characterizations for ideals.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

– Basic local separability using flips.

Mon stability: being unordered.

– Bounded-depth decompositions by alternating flips and localization.

Goal: Model-checking FO is fpt on **mon dependent** classes.

– Achieved on **mon stable** graph classes.

– Main hurdle on **mon dependent**: **decomposition**.

A lot of other **avenues** in the theory.

– Obstruction characterizations for ideals.

– Sparsification conjecture.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

– Basic local separability using flips.

Mon stability: being unordered.

– Bounded-depth decompositions by alternating flips and localization.

Goal: Model-checking FO is fpt on **mon dependent** classes.

– Achieved on **mon stable** graph classes.

– Main hurdle on **mon dependent**: **decomposition**.

A lot of other **avenues** in the theory.

– Obstruction characterizations for ideals.

– Sparsification conjecture.

– Fine understanding of the transduction order.

Conclusions

Transductions: a model-theoretic notion of embedding for classes.

Mon dependence: being non-equivalent to all graphs.

– Basic local separability using flips.

Mon stability: being unordered.

– Bounded-depth decompositions by alternating flips and localization.

Goal: Model-checking FO is fpt on **mon dependent** classes.

– Achieved on **mon stable** graph classes.

– Main hurdle on **mon dependent**: **decomposition**.

A lot of other **avenues** in the theory.

– Obstruction characterizations for ideals.

– Sparsification conjecture.

– Fine understanding of the transduction order.

Thanks for attention!