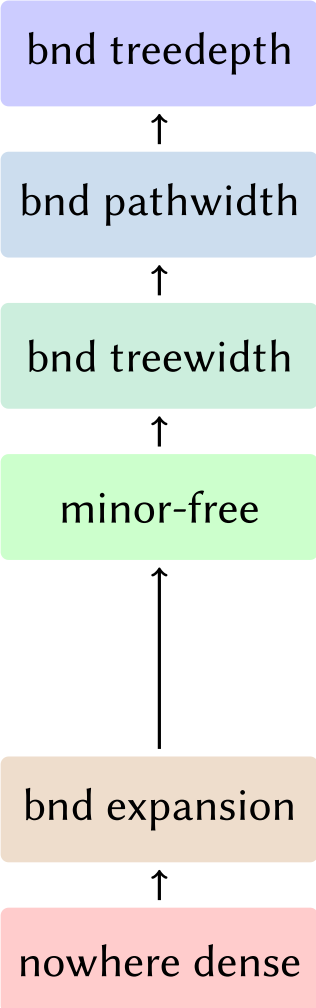
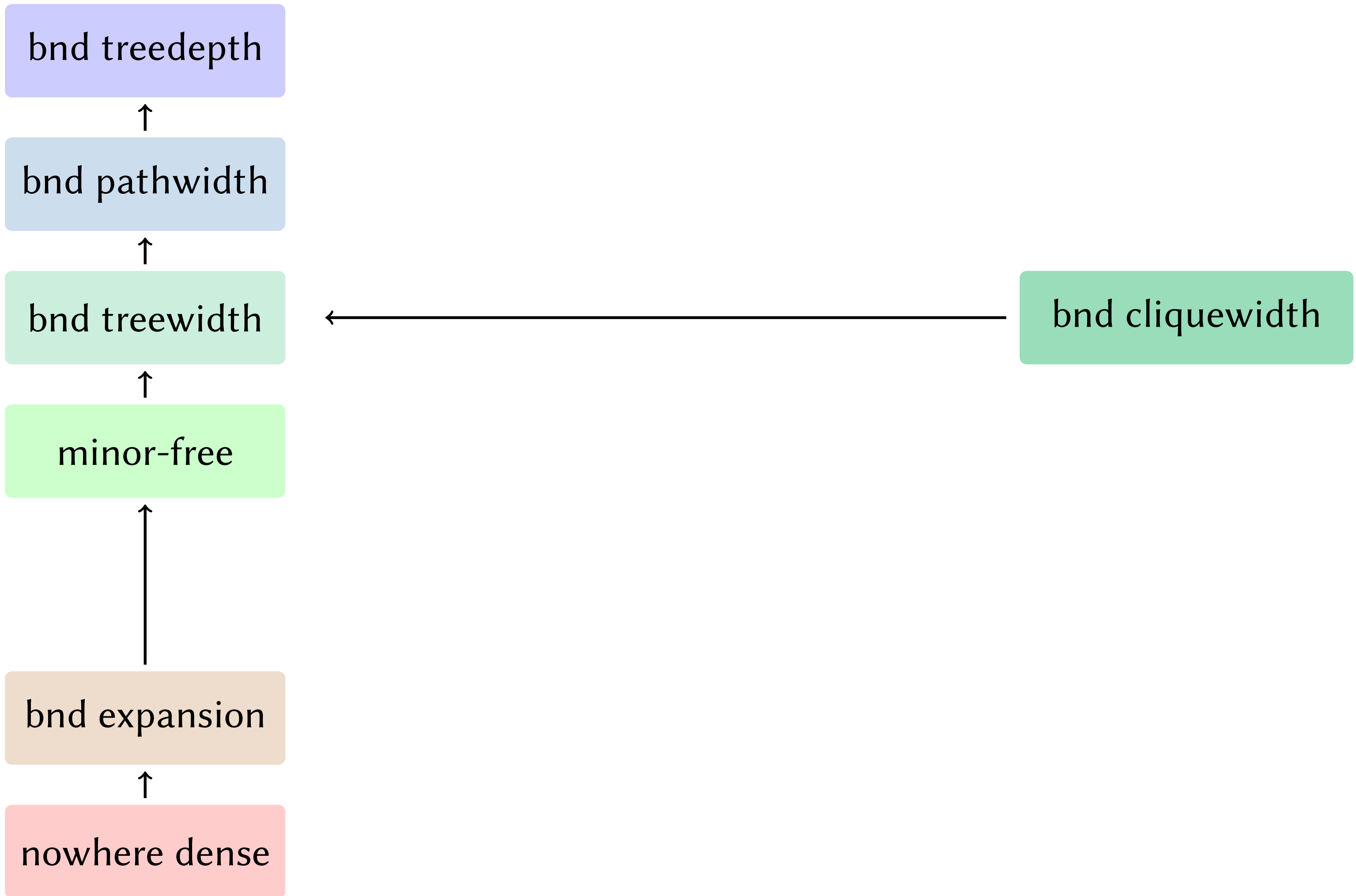
Stable graphs of bounded twin-width

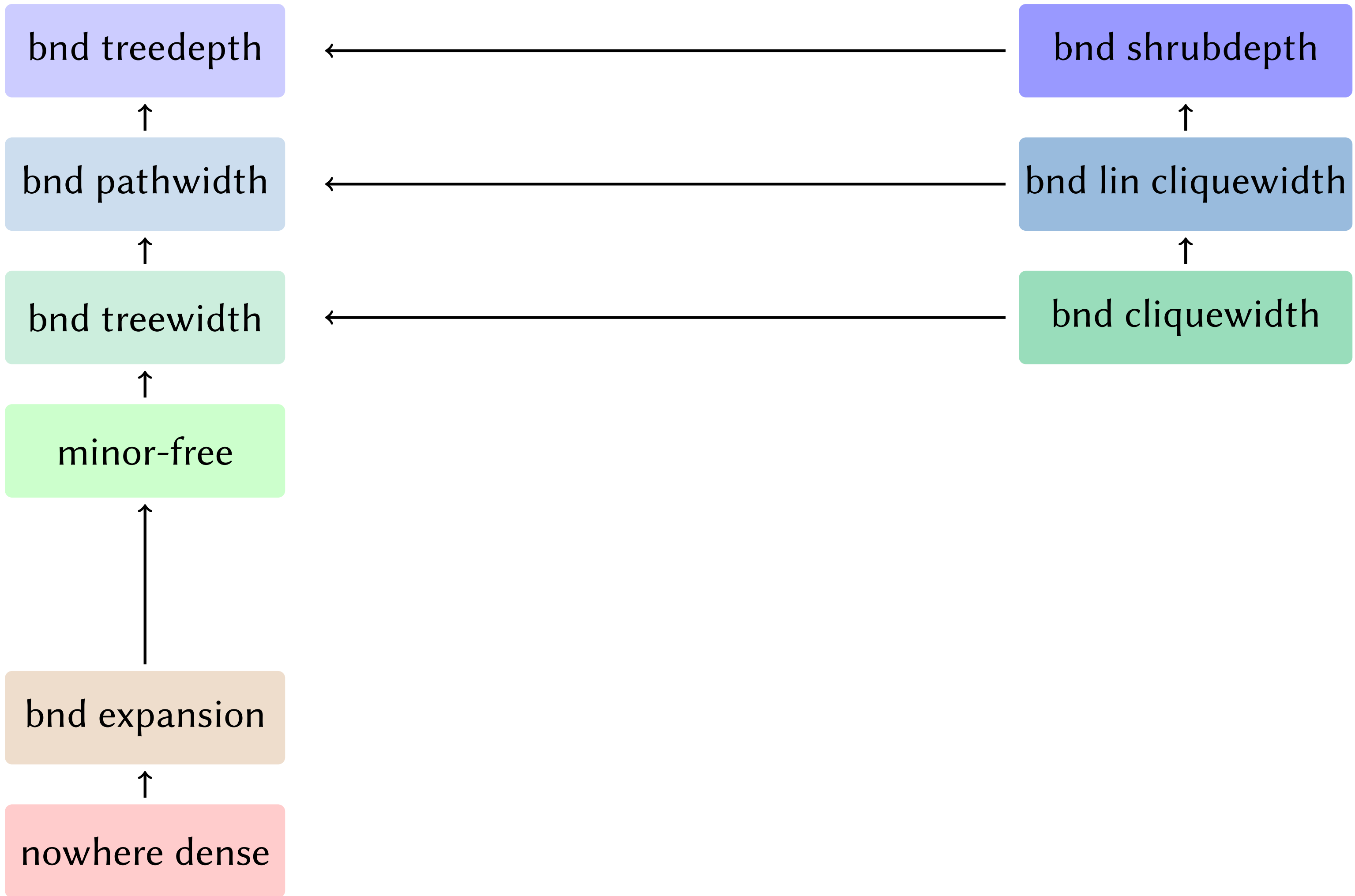
joint work with Jakub Gajarský and Szymon Toruńczyk

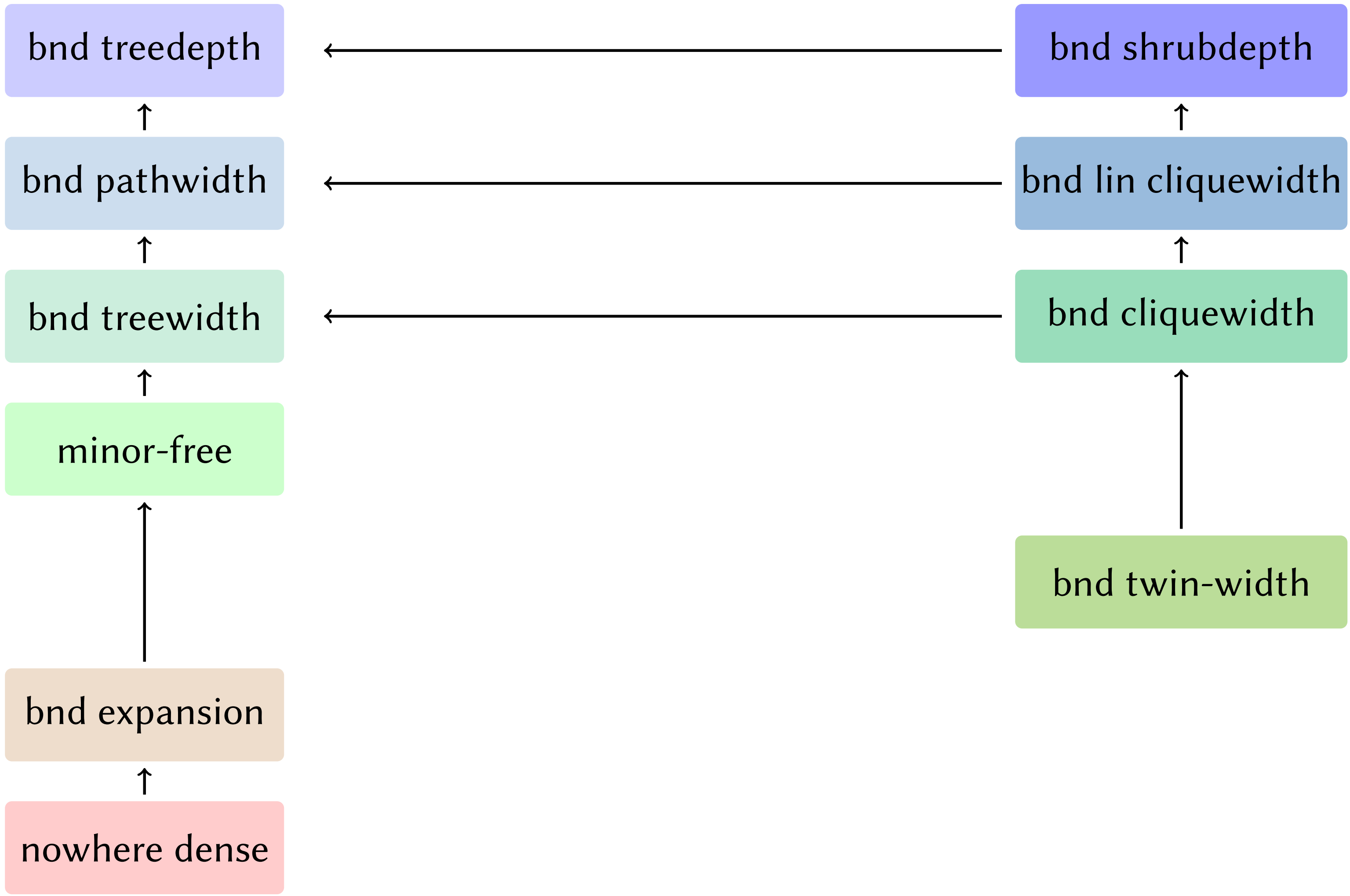
1st Twin-width Workshop

Aussois, May 24th, 2023


bnd treedepth
bnd pathwidth
bnd treewidth
minor-free
bnd expansion
nowhere dense


bnd treedepth
bnd pathwidth
bnd treewidth
bnd cliquewidth
minor-free
bnd expansion
nowhere dense


bnd treedepth
bnd shrubdepth
bnd pathwidth
bnd lin cliquewidth
bnd treewidth
bnd cliquewidth
minor-free
bnd expansion
nowhere dense


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

Bounded sparse twin-width

Bounded sparse twin-width

Def: Class $\mathscr{C}$ is **weakly sparse** if $\mathscr{C}$ is $K_{t,t}$-subgraph-free for some t.

Bounded sparse twin-width

Def: Class $\mathscr{C}$ is **weakly sparse** if $\mathscr{C}$ is $K_{t,t}$-subgraph-free for some t.

bnd treewidth $=$ bnd cliquewidth $\cap$ weakly sparse

Bounded sparse twin-width

Def: Class $\mathscr{C}$ is **weakly sparse** if $\mathscr{C}$ is $K_{t,t}$-subgraph-free for some t.

bnd treewidth $=$ bnd cliquewidth $\cap$ weakly sparse

bnd pathwidth $=$ bnd lin cliquewidth $\cap$ weakly sparse

Bounded sparse twin-width

Def: Class $\mathscr{C}$ is **weakly sparse** if $\mathscr{C}$ is $K_{t,t}$-subgraph-free for some t.

$$\text{bnd treewidth} = \text{bnd cliquewidth} \cap \text{weakly sparse}$$

$$\text{bnd pathwidth} = \text{bnd lin cliquewidth} \cap \text{weakly sparse}$$

$$\text{bnd treedepth} = \text{bnd shrubdepth} \cap \text{weakly sparse}$$

Bounded sparse twin-width

Def: Class $\mathscr{C}$ is **weakly sparse** if $\mathscr{C}$ is $K_{t,t}$-subgraph-free for some t.

$$\text{bnd treewidth} = \text{bnd cliquewidth} \cap \text{weakly sparse}$$

$$\text{bnd pathwidth} = \text{bnd lin cliquewidth} \cap \text{weakly sparse}$$

$$\text{bnd treedepth} = \text{bnd shrubdepth} \cap \text{weakly sparse}$$

Def: Class $\mathscr{C}$ has **bnd sparse twin-width** if $\mathscr{C}$ has **bnd twin-width** and is **weakly sparse**.

Bounded sparse twin-width

Def: Class $\mathscr{C}$ is **weakly sparse** if $\mathscr{C}$ is $K_{t,t}$-subgraph-free for some t.

bnd treewidth = bnd cliquewidth ∩ weakly sparse

bnd pathwidth = bnd lin cliquewidth ∩ weakly sparse

bnd treedepth = bnd shrubdepth ∩ weakly sparse

Def: Class $\mathscr{C}$ has **bnd sparse twin-width** if $\mathscr{C}$ has **bnd twin-width** and is **weakly sparse**.

Theorem (TWW1, TWW2, DGJOdMR'22)

minor-free $\subsetneq$ bnd sparse tww $\subsetneq$ bnd expansion

Bounded sparse twin-width

Def: Class $\mathscr{C}$ is **weakly sparse** if $\mathscr{C}$ is $K_{t,t}$-subgraph-free for some t.

$$\text{bnd treewidth} = \text{bnd cliquewidth} \cap \text{weakly sparse}$$

$$\text{bnd pathwidth} = \text{bnd lin cliquewidth} \cap \text{weakly sparse}$$

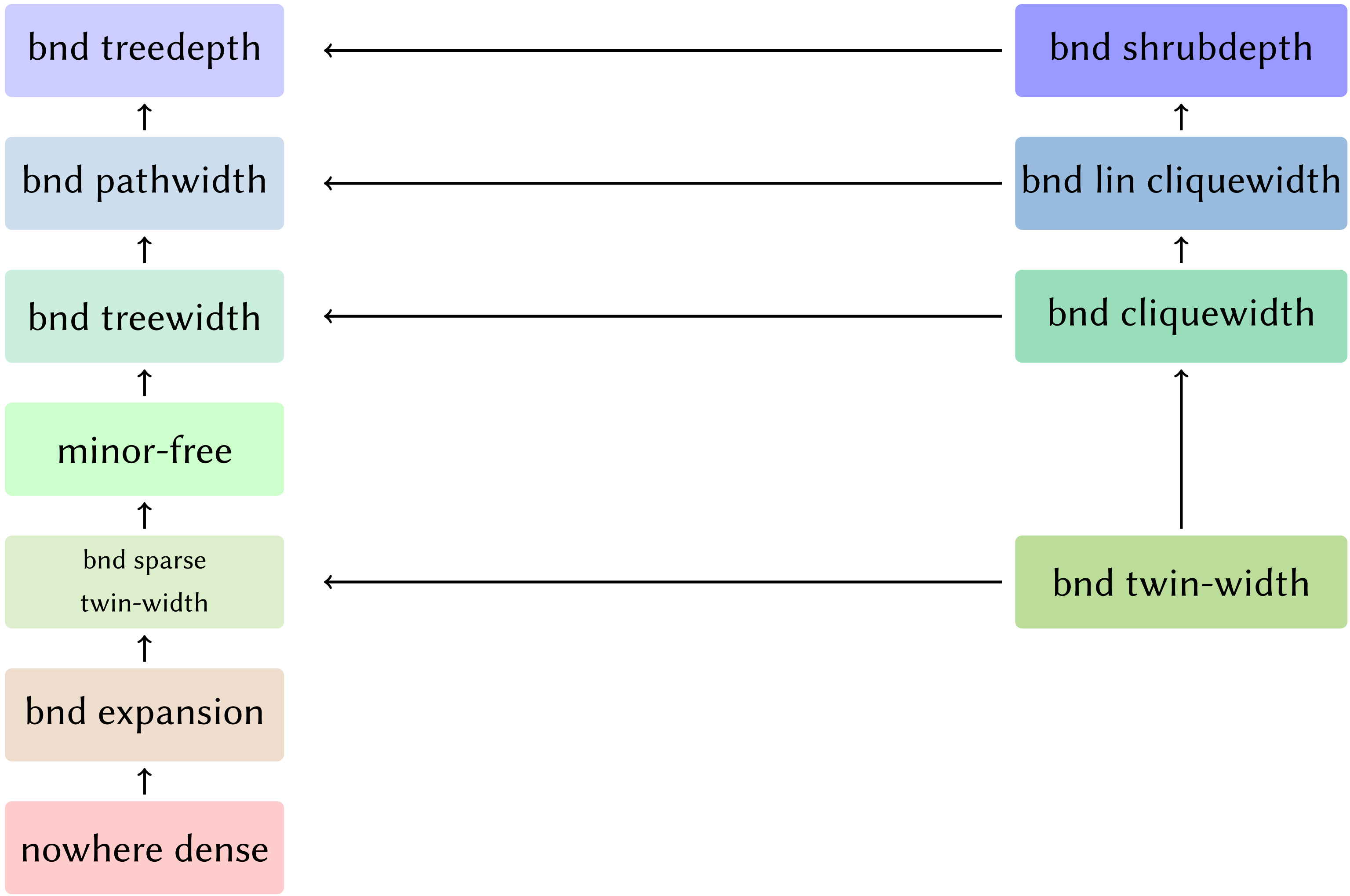
$$\text{bnd treedepth} = \text{bnd shrubdepth} \cap \text{weakly sparse}$$

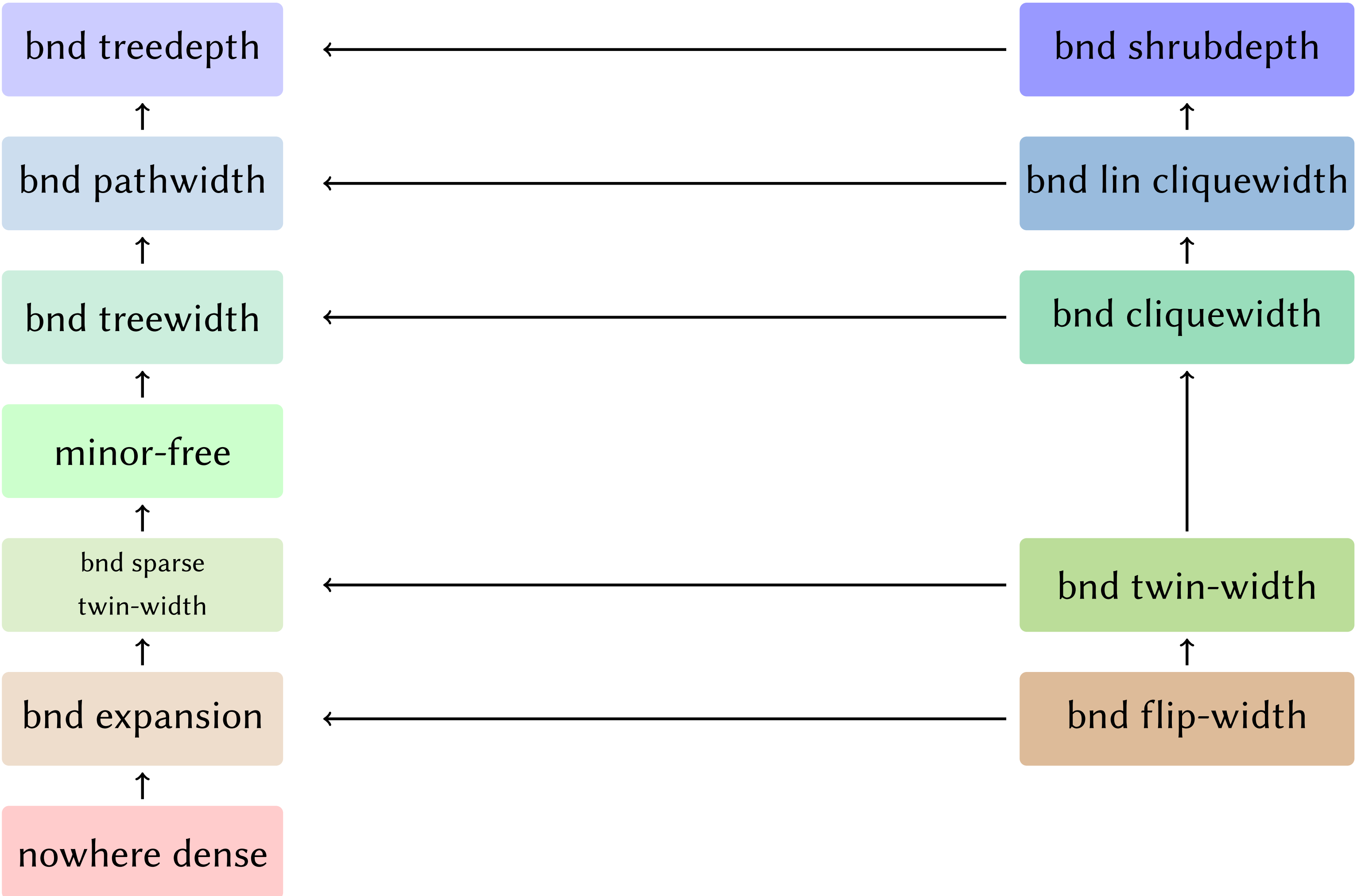
Def: Class $\mathscr{C}$ has **bnd sparse twin-width** if $\mathscr{C}$ has **bnd twin-width** and is **weakly sparse**.

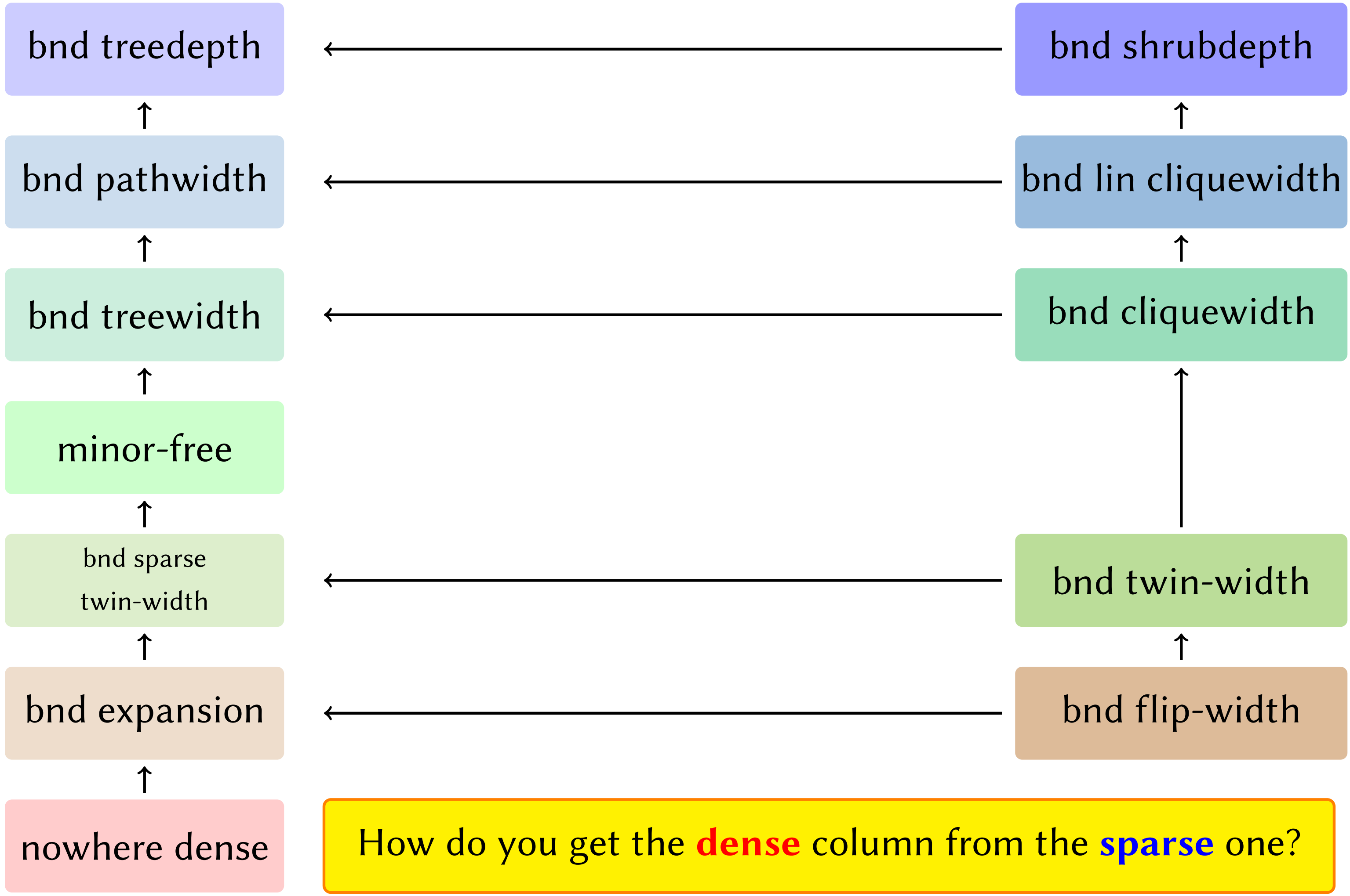
Theorem (TWW1, TWW2, DGJOdMR'22)

$$\text{minor-free} \subsetneq \text{bnd sparse tww} \subsetneq \text{bnd expansion}$$

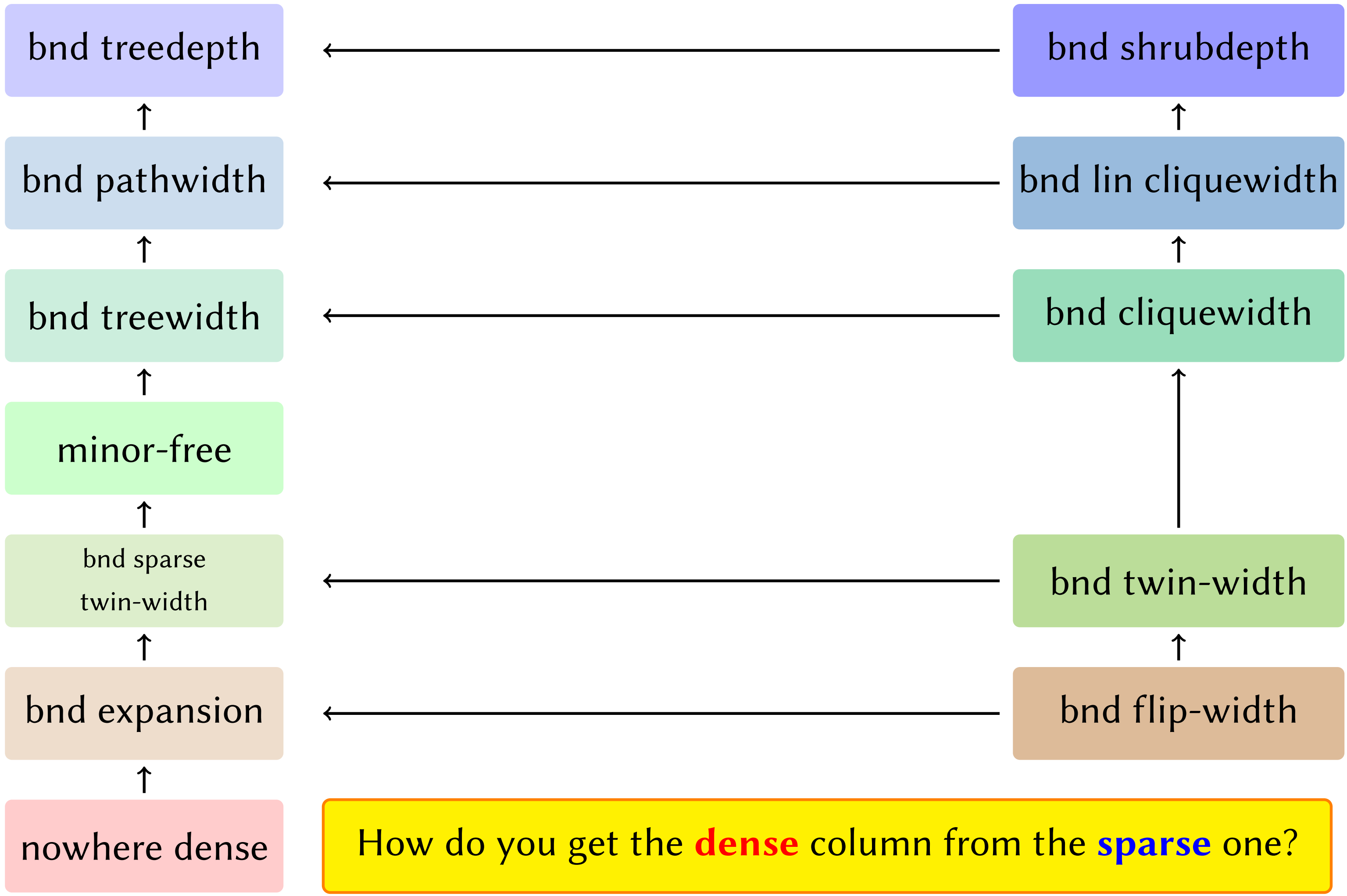
Also: mixed minors $\rightsquigarrow$ grid minors.


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


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



How do you get the **dense** column from the **sparse** one?

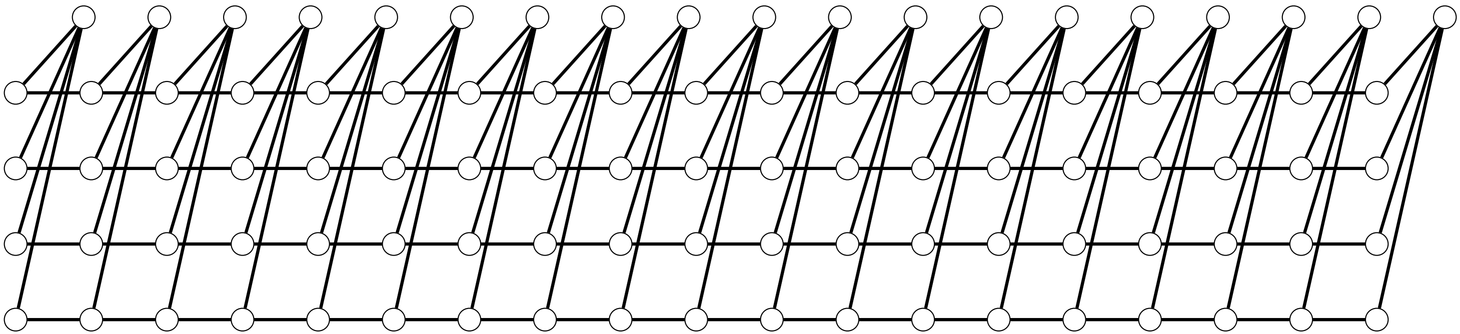


Idea: Close under logically defined operations.

Transductions: example

Transductions: example

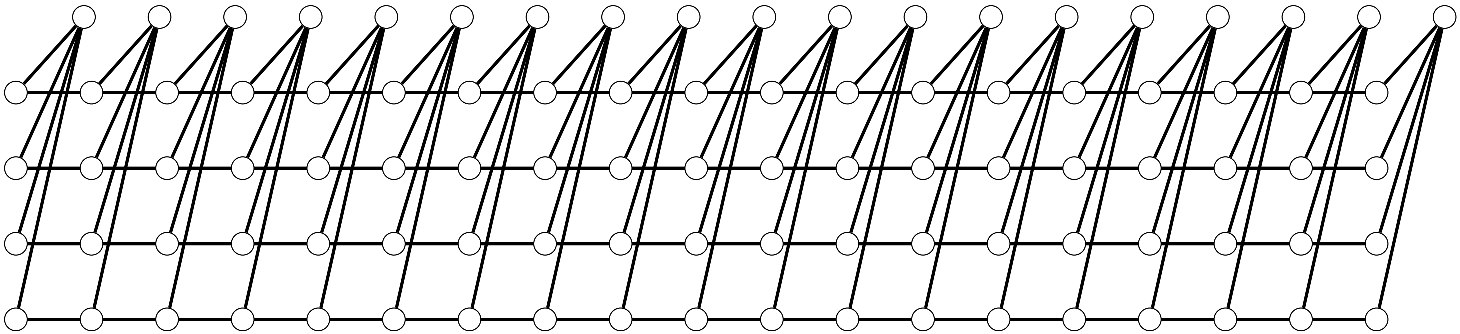
$\mathscr{D}$:= class of James Davies' examples



Transductions: example

$\mathscr{D}$:= class of James Davies' examples

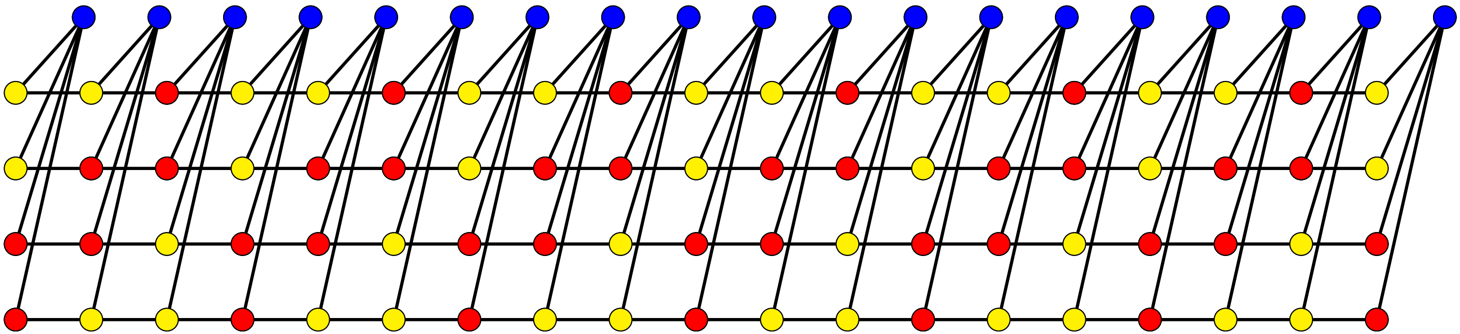
Claim: $\mathscr{D}$ transduces a class $\mathscr{C}$ that contains a subdivision of every wall.



Transductions: example

$\mathscr{D}$:= class of James Davies' examples

Claim: $\mathscr{D}$ transduces a class $\mathscr{C}$ that contains a subdivision of every wall.

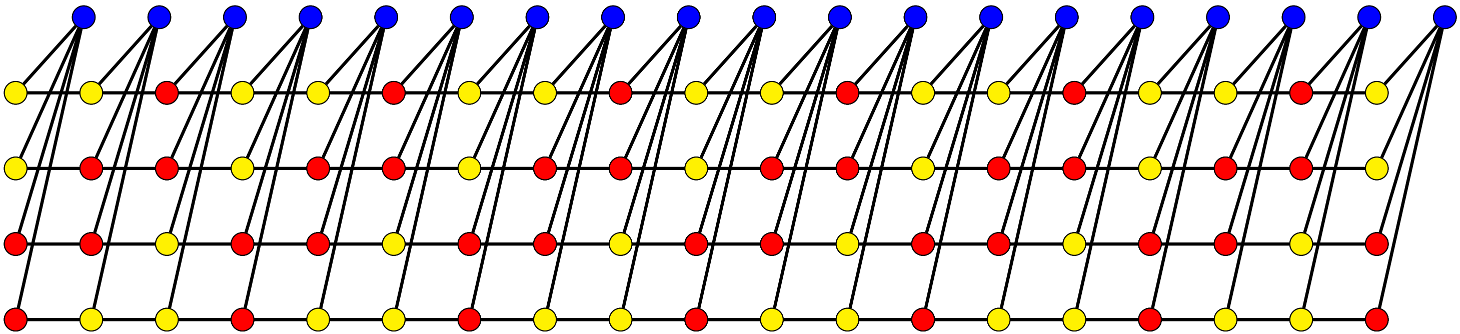


Step 1: Color vertices using **yellow**, **red**, and **blue**.

Transductions: example

$\mathscr{D}$:= class of James Davies' examples

Claim: $\mathscr{D}$ transduces a class $\mathscr{C}$ that contains a subdivision of every wall.



Step 1: Color vertices using **yellow**, **red**, and **blue**.

Step 2: Interpret a new adjacency relation using:

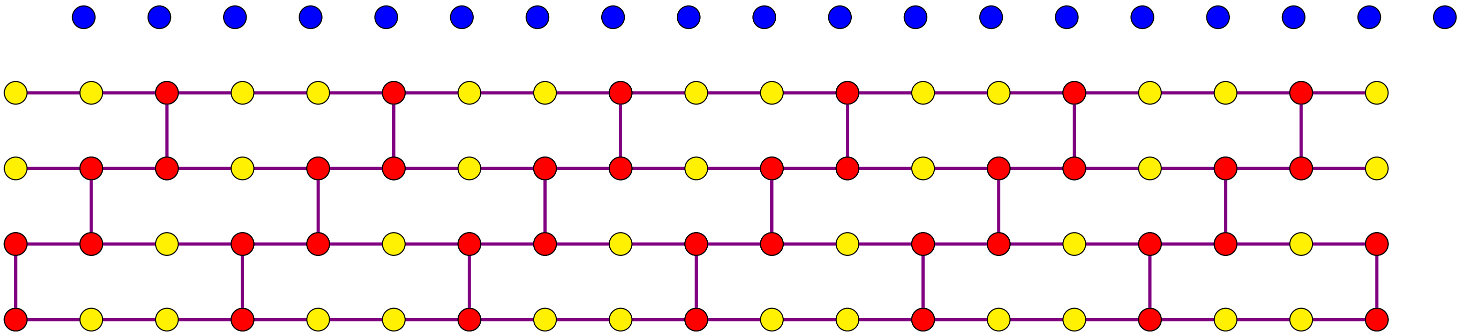
$\varphi(x, y) =$ (x and y are **yellow** or **red** and adjacent) or

(x and y are **red** and have a common **blue** neighbor)

Transductions: example

$\mathscr{D}$:= class of James Davies' examples

Claim: $\mathscr{D}$ transduces a class $\mathscr{C}$ that contains a subdivision of every wall.



Step 1: Color vertices using **yellow**, **red**, and **blue**.

Step 2: Interpret a new adjacency relation using:

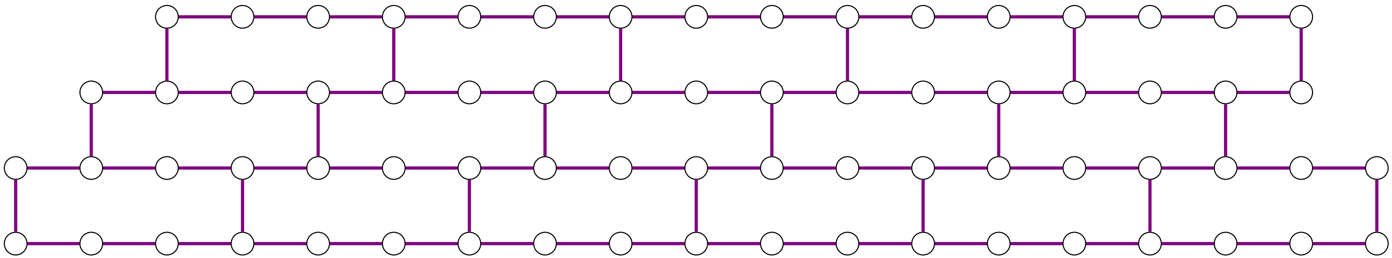
$\varphi(x, y) =$ (x and y are **yellow** or **red** and adjacent) or

(x and y are **red** and have a common **blue** neighbor)

Transductions: example

$\mathscr{D}$:= class of James Davies' examples

Claim: $\mathscr{D}$ transduces a class $\mathscr{C}$ that contains a subdivision of every wall.



Step 1: Color vertices using **yellow**, **red**, and **blue**.

Step 2: Interpret a new adjacency relation using:

$\varphi(x, y) =$ (x and y are **yellow** or **red** and adjacent) or

(x and y are **red** and have a common **blue** neighbor)

Step 3: Take any induced subgraph.

Transductions

Transductions

Transduction $\mathsf{T} = (C, \varphi(x, y))$, run on G:

Transductions

Transduction $\mathsf{T} = (C, \varphi(x, y))$, run on G:

– color vertices using the palette C;

Transductions

Transduction $\mathsf{T} = (C, \varphi(x, y))$, run on G:

- color vertices using the palette C;
- interpret a new edge relation using $\varphi(x, y) \in \mathsf{FO}$;

Transductions

Transduction $\mathsf{T} = (C, \varphi(x, y))$, run on G:

– color vertices using the palette C;

– interpret a new edge relation using $\varphi(x, y) \in \mathsf{FO}$;

– output any induced subgraph.

Transductions

Transduction $\mathsf{T} = (C, \varphi(x, y))$, run on G:

- color vertices using the palette C;
- interpret a new edge relation using $\varphi(x, y) \in \mathsf{FO}$;
- output any induced subgraph.

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

Transductions

Transduction $\mathsf{T} = (C, \varphi(x, y))$, run on G:

- color vertices using the palette C;
- interpret a new edge relation using $\varphi(x, y) \in \mathsf{FO}$;
- output any induced subgraph.

$\mathsf{T}(G) :=$ all possible outputs of T on G $\qquad$ $\mathsf{T}(\mathscr{C}) := \bigcup_{G \in \mathscr{C}} \mathsf{T}(G)$

Transductions

Transduction $\mathsf{T} = (C, \varphi(x,y))$, run on G:

- color vertices using the palette C;
- interpret a new edge relation using $\varphi(x,y) \in \mathsf{FO}$;
- output any induced subgraph.

$\mathsf{T}(G) :=$ all possible outputs of T on G $\qquad$ $\mathsf{T}(\mathscr{C}) := \bigcup_{G \in \mathscr{C}} \mathsf{T}(G)$

Def: $\mathscr{D}$ is **transducible** from $\mathscr{C}$ if $\mathscr{D} \subseteq \mathsf{T}(\mathscr{C})$ for some transduction T.

Transductions

Transduction $\mathsf{T} = (C, \varphi(x, y))$, run on G:

- color vertices using the palette C;
- interpret a new edge relation using $\varphi(x, y) \in \mathsf{FO}$;
- output any induced subgraph.

$\mathsf{T}(G) :=$ all possible outputs of T on G $\qquad$ $\mathsf{T}(\mathscr{C}) := \bigcup_{G \in \mathscr{C}} \mathsf{T}(G)$

Def: $\mathscr{D}$ is **transducible** from $\mathscr{C}$ if $\mathscr{D} \subseteq \mathsf{T}(\mathscr{C})$ for some transduction T.

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

Transductions

Transduction $\mathsf{T} = (C, \varphi(x, y))$, run on G:

- color vertices using the palette C;
- interpret a new edge relation using $\varphi(x, y) \in \mathsf{FO}$;
- output any induced subgraph.

$\mathsf{T}(G) :=$ all possible outputs of T on G $\qquad$ $\mathsf{T}(\mathscr{C}) := \bigcup_{G \in \mathscr{C}} \mathsf{T}(G)$

Def: $\mathscr{D}$ is **transducible** from $\mathscr{C}$ if $\mathscr{D} \subseteq \mathsf{T}(\mathscr{C})$ for some transduction T.

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

Notation: $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$.

Transductions

Transduction $\mathsf{T} = (C, \varphi(x,y))$, run on G:

- color vertices using the palette C;
- interpret a new edge relation using $\varphi(x,y) \in \mathsf{FO}$;
- output any induced subgraph.

$\mathsf{T}(G) :=$ all possible outputs of T on G $\qquad$ $\mathsf{T}(\mathscr{C}) := \bigcup_{G \in \mathscr{C}} \mathsf{T}(G)$

Def: $\mathscr{D}$ is **transducible** from $\mathscr{C}$ if $\mathscr{D} \subseteq \mathsf{T}(\mathscr{C})$ for some transduction T.

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

Notation: $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$.

Def: $\mathscr{L}$-transduction = transduction where $\varphi \in \mathscr{L}$.

Transductions

Transduction $\mathsf{T} = (C, \varphi(x, y))$, run on G:

- color vertices using the palette C;
- interpret a new edge relation using $\varphi(x, y) \in \mathsf{FO}$;
- output any induced subgraph.

$\mathsf{T}(G) :=$ all possible outputs of T on G $\qquad$ $\mathsf{T}(\mathscr{C}) := \bigcup_{G \in \mathscr{C}} \mathsf{T}(G)$

Def: $\mathscr{D}$ is **transducible** from $\mathscr{C}$ if $\mathscr{D} \subseteq \mathsf{T}(\mathscr{C})$ for some transduction T.

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

Notation: $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$.

Def: $\mathscr{L}$-transduction = transduction where $\varphi \in \mathscr{L}$.

FO-transductions, MSO_1-transductions, MSO_2-transductions, ...

Transductions and parameters

Transductions and parameters

Fact: $\mathscr{C}$ has **bnd cliquewidth** iff $\mathscr{C}$ can be MSO_1-transduced from the class of trees.

Transductions and parameters

Fact: $\mathscr{C}$ has **bnd cliquewidth** iff $\mathscr{C}$ can be MSO_1-transduced from the class of trees.

Cor: If $\mathscr{C}$ has **bnd cliquewidth** and $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$, then so does $\mathscr{D}$.

Transductions and parameters

Fact: $\mathscr{C}$ has **bnd cliquewidth** iff $\mathscr{C}$ can be MSO_1-transduced from the class of trees.

Cor: If $\mathscr{C}$ has **bnd cliquewidth** and $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$, then so does $\mathscr{D}$.

We say that bnd cliquewidth is a **transduction ideal**.

Transductions and parameters

Fact: $\mathscr{C}$ has **bnd cliquewidth** iff $\mathscr{C}$ can be MSO_1-transduced from the class of trees.

Cor: If $\mathscr{C}$ has **bnd cliquewidth** and $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$, then so does $\mathscr{D}$.

We say that bnd cliquewidth is a **transduction ideal**.

Other transduction ideals:

Transductions and parameters

Fact: $\mathscr{C}$ has **bnd cliquewidth** iff $\mathscr{C}$ can be MSO_1-transduced from the class of trees.

Cor: If $\mathscr{C}$ has **bnd cliquewidth** and $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$, then so does $\mathscr{D}$.

We say that bnd cliquewidth is a **transduction ideal**.

Other transduction ideals:

– bnd shrubdepth;

Transductions and parameters

Fact: $\mathscr{C}$ has **bnd cliquewidth** iff $\mathscr{C}$ can be MSO_1-transduced from the class of trees.

Cor: If $\mathscr{C}$ has **bnd cliquewidth** and $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$, then so does $\mathscr{D}$.

We say that bnd cliquewidth is a **transduction ideal**.

Other transduction ideals:

– bnd shrubdepth;

– bnd lin cliquewidth;

Transductions and parameters

Fact: $\mathscr{C}$ has **bnd cliquewidth** iff $\mathscr{C}$ can be MSO_1-transduced from the class of trees.

Cor: If $\mathscr{C}$ has **bnd cliquewidth** and $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$, then so does $\mathscr{D}$.

We say that bnd cliquewidth is a **transduction ideal**.

Other transduction ideals:

– bnd shrubdepth;

– bnd lin cliquewidth;

– bnd twin-width;

Transductions and parameters

Fact: $\mathscr{C}$ has **bnd cliquewidth** iff $\mathscr{C}$ can be MSO_1-transduced from the class of trees.

Cor: If $\mathscr{C}$ has **bnd cliquewidth** and $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$, then so does $\mathscr{D}$.

We say that bnd cliquewidth is a **transduction ideal**.

Other transduction ideals:

- bnd shrubdepth;
- bnd lin cliquewidth;
- bnd twin-width;
- bnd flip-width.

Transductions and parameters

Fact: $\mathscr{C}$ has **bnd cliquewidth** iff $\mathscr{C}$ can be MSO_1-transduced from the class of trees.

Cor: If $\mathscr{C}$ has **bnd cliquewidth** and $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$, then so does $\mathscr{D}$.

We say that bnd cliquewidth is a **transduction ideal**.

Other transduction ideals:

- bnd shrubdepth;
- bnd lin cliquewidth;
- bnd twin-width;
- bnd flip-width.

Question: Can every class of **bnd cliquewidth** be transduced from a class of **bnd treewidth**?

Transductions and parameters

Fact: $\mathscr{C}$ has **bnd cliquewidth** iff $\mathscr{C}$ can be MSO_1-transduced from the class of trees.

Cor: If $\mathscr{C}$ has **bnd cliquewidth** and $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$, then so does $\mathscr{D}$.

We say that bnd cliquewidth is a **transduction ideal**.

Other transduction ideals:

– bnd shrubdepth;

– bnd lin cliquewidth;

– bnd twin-width;

– bnd flip-width.

Question: Can every class of **bnd cliquewidth** be transduced from a class of **bnd treewidth**?

Transductions and parameters

Fact: $\mathscr{C}$ has **bnd cliquewidth** iff $\mathscr{C}$ can be MSO_1-transduced from the class of trees.

Cor: If $\mathscr{C}$ has **bnd cliquewidth** and $\mathscr{D} \sqsubseteq_{\mathsf{FO}} \mathscr{C}$, then so does $\mathscr{D}$.

We say that bnd cliquewidth is a **transduction ideal**.

Other transduction ideals:

– bnd shrubdepth;

– bnd lin cliquewidth;

– bnd twin-width;

– bnd flip-width.

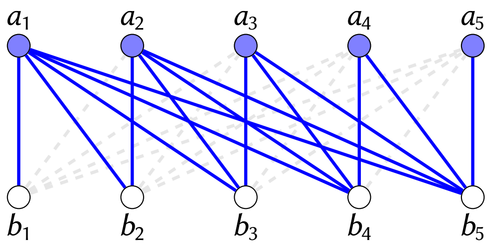
Question: Can every class of **bnd cliquewidth** be transduced from a class of **bnd treewidth**?

Equivalently: **bnd cliquewidth** = **structurally bnd treewidth**?

Monadic stability

Monadic stability

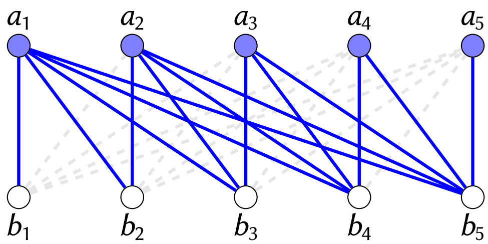
NO: The obstacle are **half-graphs**.



$a_i b_j \in E \iff i \leqslant j$

Monadic stability

NO: The obstacle are **half-graphs**.



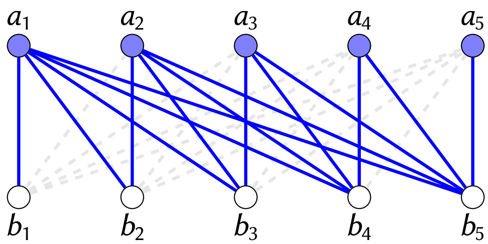
$a_i b_j \in E \iff i \leqslant j$

Theorem (Podewski and Ziegler; Adler and Adler)

If $\mathscr{C}$ is **nowhere dense**, then Half-graphs $\not\sqsubseteq_{\mathsf{FO}} \mathscr{C}$.

Monadic stability

NO: The obstacle are **half-graphs**.



$a_i b_j \in E \Leftrightarrow i \leqslant j$

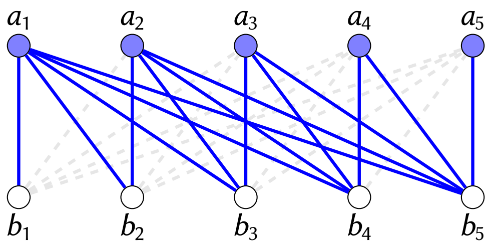
Theorem (Podewski and Ziegler; Adler and Adler)

If $\mathscr{C}$ is **nowhere dense**, then Half-graphs $\not\sqsubseteq_{\mathrm{FO}} \mathscr{C}$.

Def: A class $\mathscr{C}$ is **monadically stable** if Half-graphs $\not\sqsubseteq_{\mathrm{FO}} \mathscr{C}$.

Monadic stability

NO: The obstacle are **half-graphs**.



$a_i b_j \in E \;\Leftrightarrow\; i \leqslant j$

Theorem (Podewski and Ziegler; Adler and Adler)

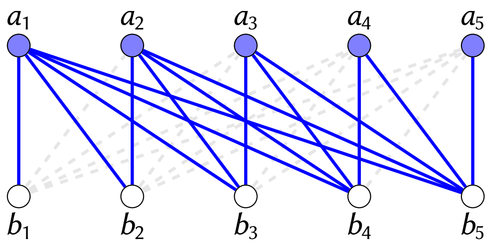
If $\mathscr{C}$ is **nowhere dense**, then Half-graphs $\not\sqsubseteq_{\mathrm{FO}} \mathscr{C}$.

Def: A class $\mathscr{C}$ is **monadically stable** if Half-graphs $\not\sqsubseteq_{\mathrm{FO}} \mathscr{C}$.

Intuition: $\mathscr{C}$ is **monadically stable** iff one cannot define arbitrarily long **total orders** in graphs from $\mathscr{C}$.

Monadic stability

NO: The obstacle are **half-graphs**.



$a_i b_j \in E \ \Leftrightarrow \ i \leqslant j$

Theorem (Podewski and Ziegler; Adler and Adler)

If $\mathscr{C}$ is **nowhere dense**, then Half-graphs $\not\sqsubseteq_{\mathsf{FO}} \mathscr{C}$.

Def: A class $\mathscr{C}$ is **monadically stable** if Half-graphs $\not\sqsubseteq_{\mathsf{FO}} \mathscr{C}$.

Intuition: $\mathscr{C}$ is **monadically stable** iff one cannot define arbitrarily long **total orders** in graphs from $\mathscr{C}$.

Intuition: Whatever we transduce from sparse classes, no half-graphs.

Monadic dependence

Monadic dependence

Def: A class $\mathscr{C}$ is **monadically dependent (NIP)** if Graphs $\not\sqsubseteq_{\mathsf{FO}} \mathscr{C}$.

Monadic dependence

Def: A class $\mathscr{C}$ is **monadically dependent (NIP)** if Graphs $\not\sqsubseteq_{\mathsf{FO}} \mathscr{C}$.

nowhere dense $\subseteq$ **mon stable** $\subseteq$ **mon dependent**

Monadic dependence

Def: A class $\mathscr{C}$ is **monadically dependent (NIP)** if Graphs $\not\sqsubseteq_{\mathsf{FO}} \mathscr{C}$.

nowhere dense $\subseteq$ **mon stable** $\subseteq$ **mon dependent**

Fact: If $\mathscr{C}$ is **weakly sparse**, then

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

Monadic dependence

Def: A class $\mathscr{C}$ is **monadically dependent (NIP)** if Graphs $\not\sqsubseteq_{\mathsf{FO}} \mathscr{C}$.

nowhere dense $\subseteq$ **mon stable** $\subseteq$ **mon dependent**

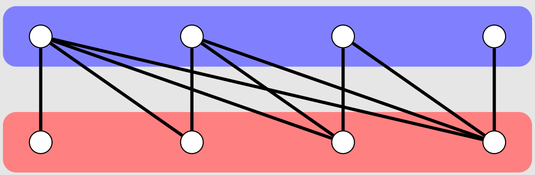
Fact: If $\mathscr{C}$ is **weakly sparse**, then

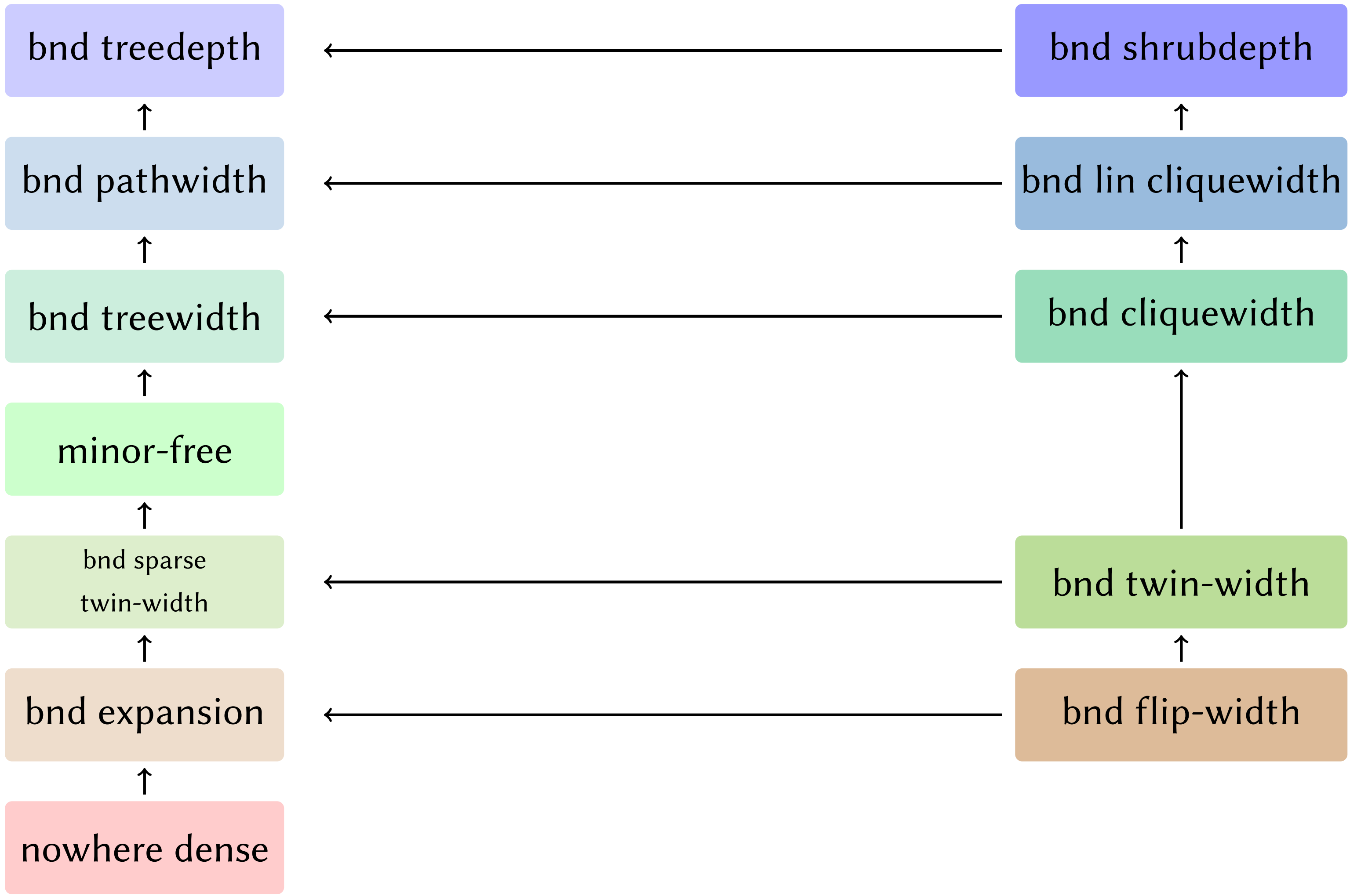
$\mathscr{C}$ is **nowhere dense** $\Leftrightarrow$ $\mathscr{C}$ is **mon stable** $\Leftrightarrow$ $\mathscr{C}$ is **mon dependent**.

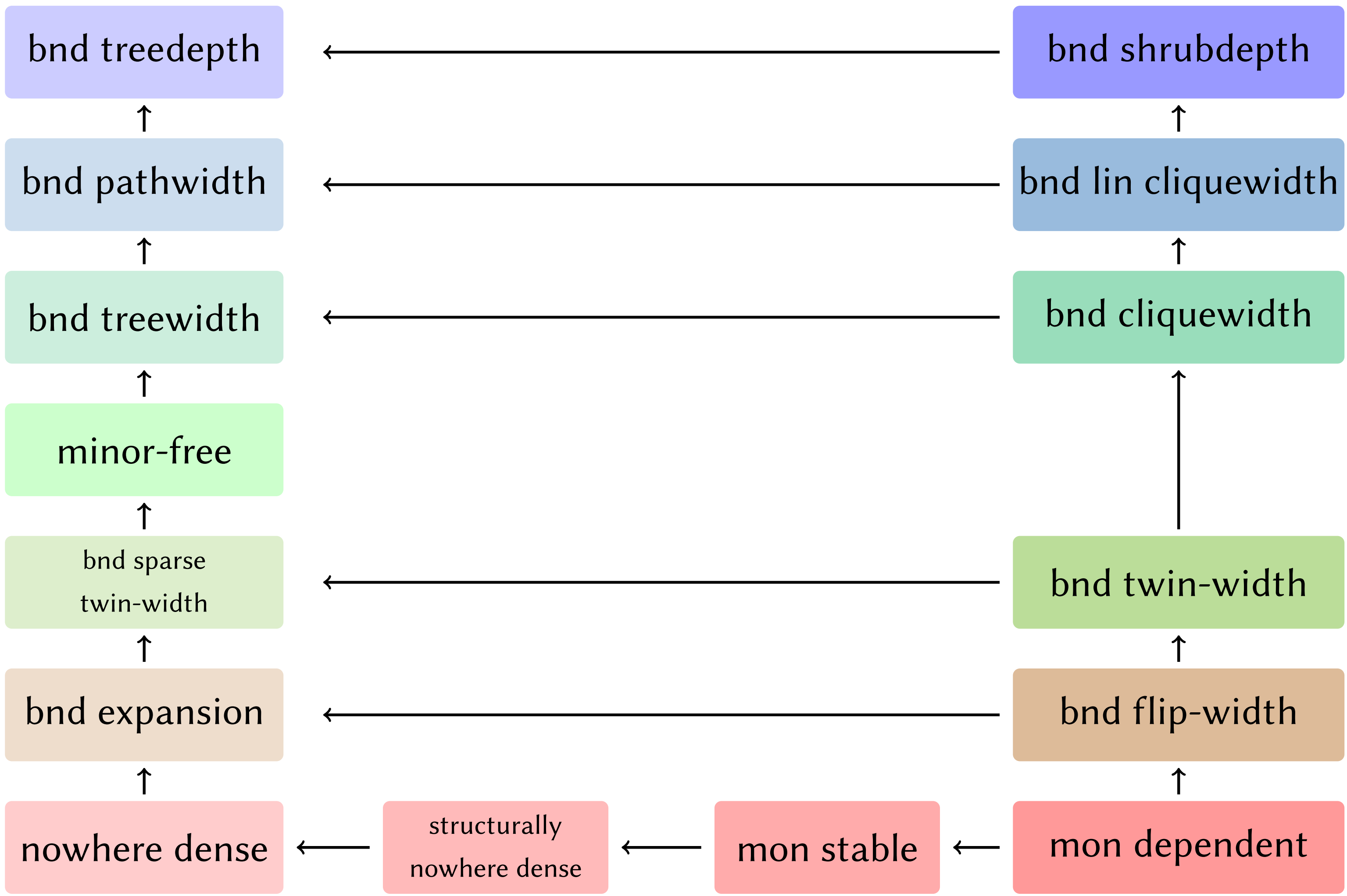
Fact: If $\mathscr{C}$ is **mon dependent**, then

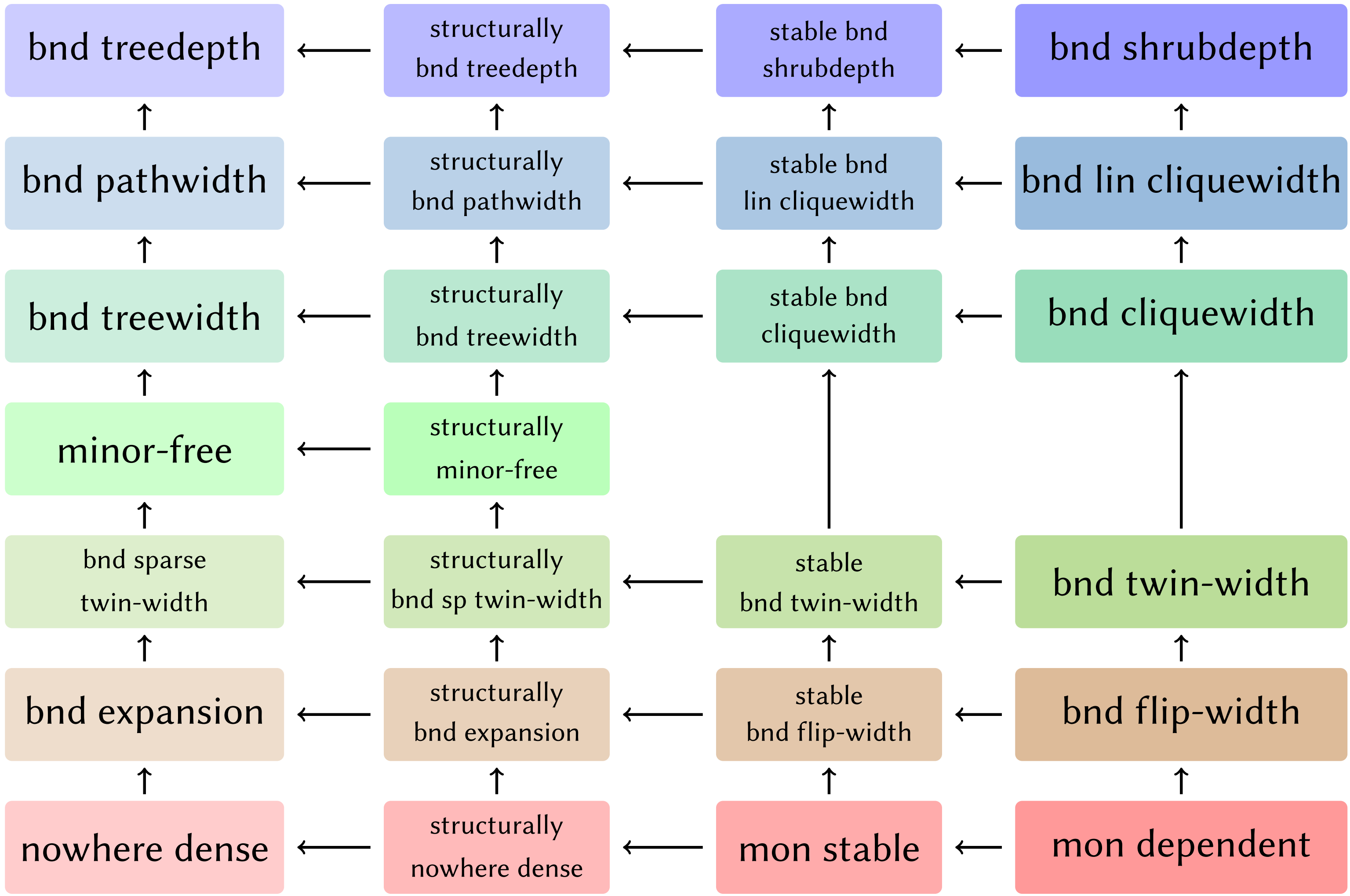
$\mathscr{C}$ is **mon stable** $\Leftrightarrow$ $\mathscr{C}$ has a **stable edge relation**;

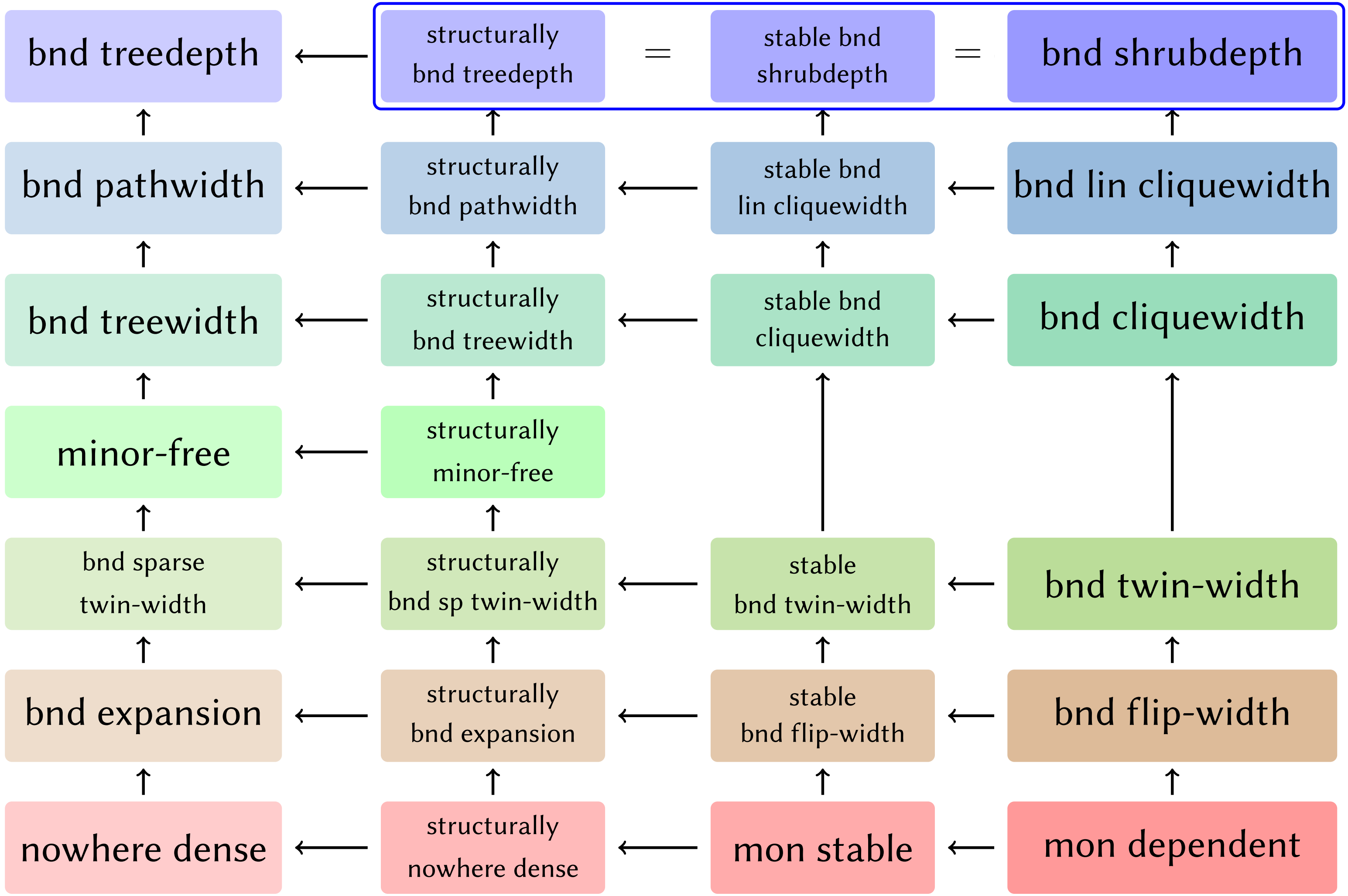
this means excluding some **semi-induced** half-graph.

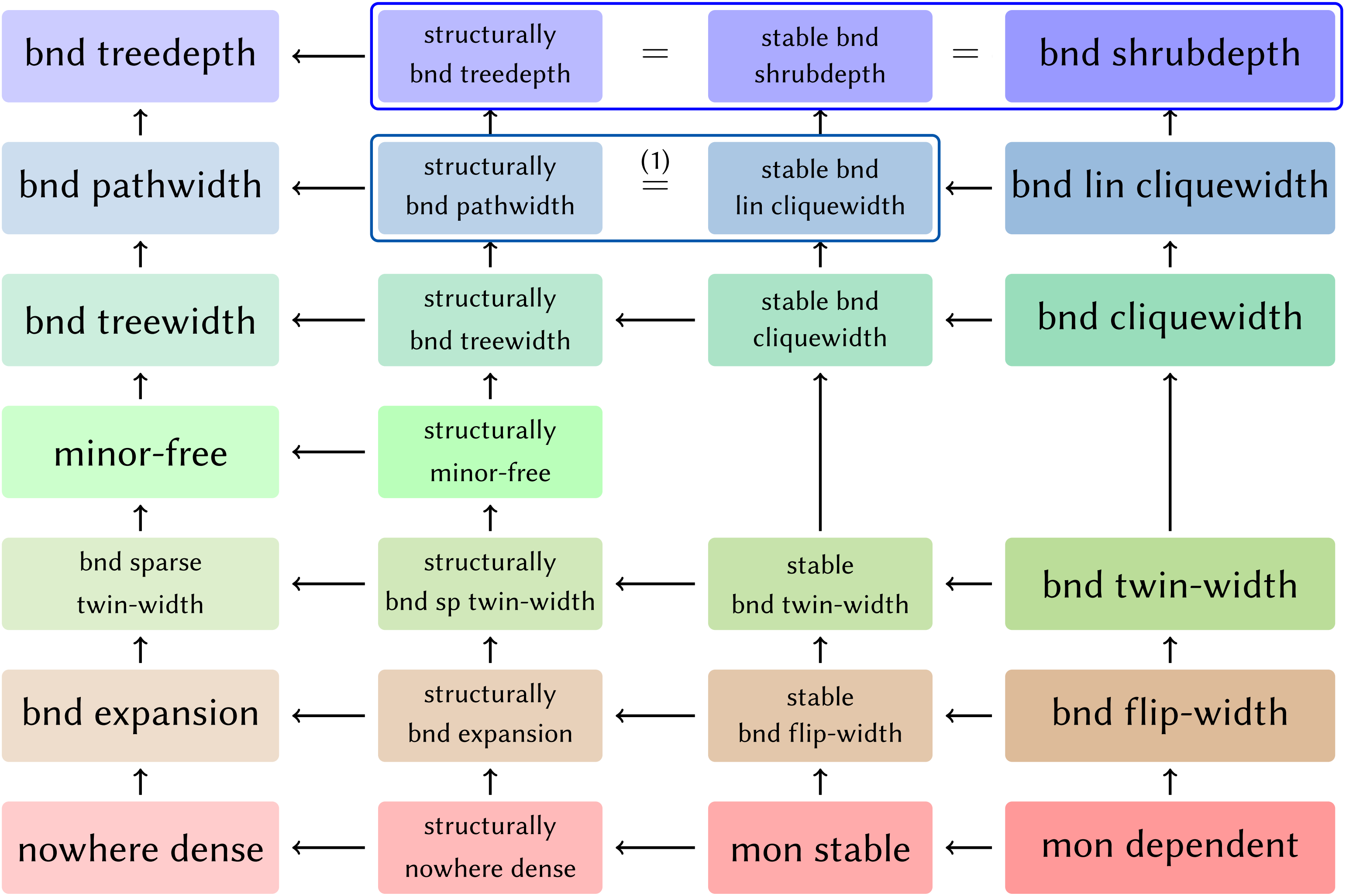



bnd treedepth
bnd shrubdepth
bnd pathwidth
bnd lin cliquewidth
bnd treewidth
bnd cliquewidth
minor-free
bnd sparse twin-width
bnd twin-width
bnd expansion
bnd flip-width
nowhere dense

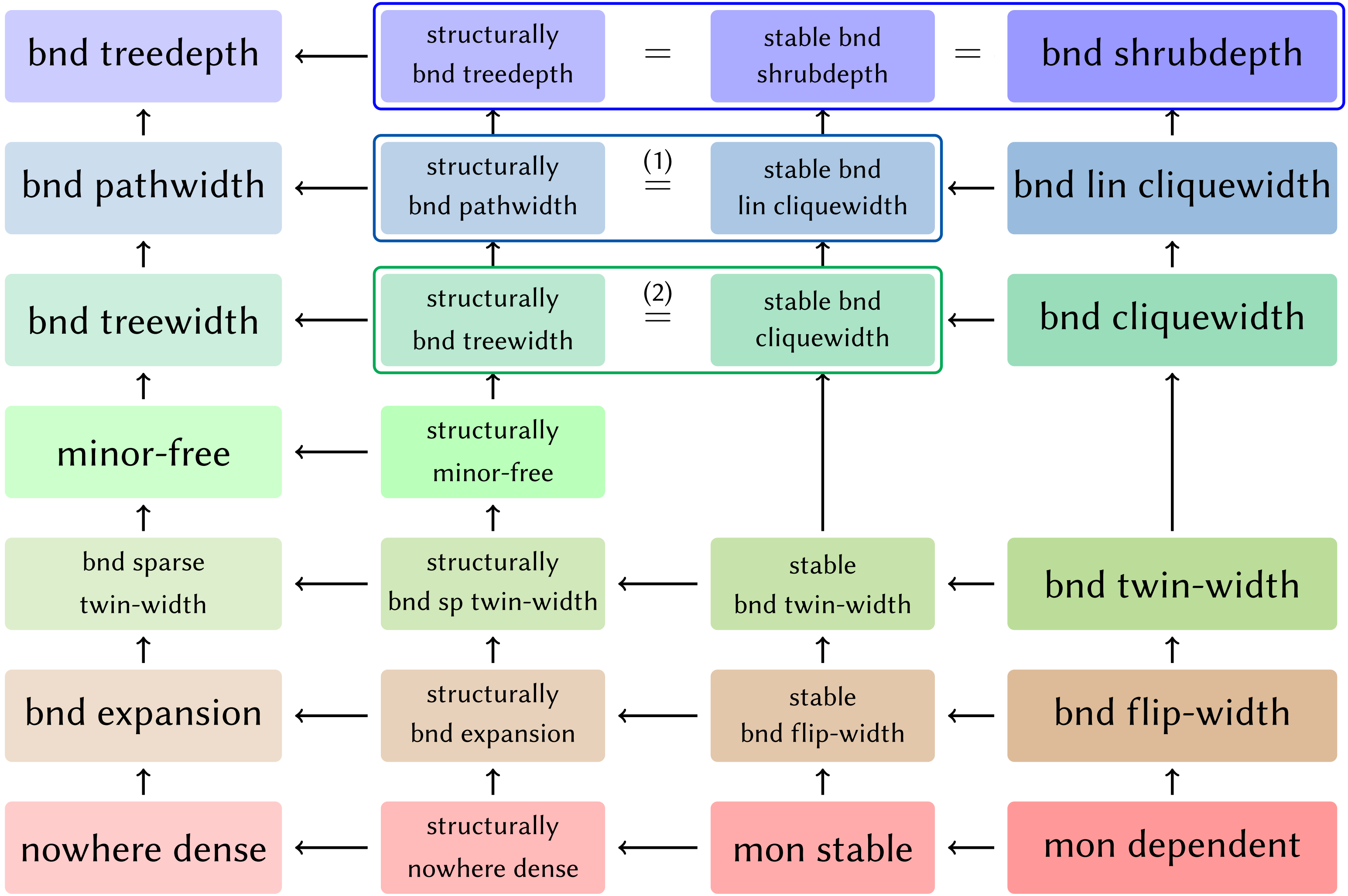

bnd treedepth
bnd shrubdepth
bnd pathwidth
bnd lin cliquewidth
bnd treewidth
bnd cliquewidth
minor-free
bnd sparse twin-width
bnd twin-width
bnd expansion
bnd flip-width
nowhere dense
structurally nowhere dense
mon stable
mon dependent


bnd treedepth
structurally bnd treedepth
stable bnd shrubdepth
bnd shrubdepth
bnd pathwidth
structurally bnd pathwidth
stable bnd lin cliquewidth
bnd lin cliquewidth
bnd treewidth
structurally bnd treewidth
stable bnd cliquewidth
bnd cliquewidth
minor-free
structurally minor-free
bnd sparse twin-width
structurally bnd sp twin-width
stable bnd twin-width
bnd twin-width
bnd expansion
structurally bnd expansion
stable bnd flip-width
bnd flip-width
nowhere dense
structurally nowhere dense
mon stable
mon dependent

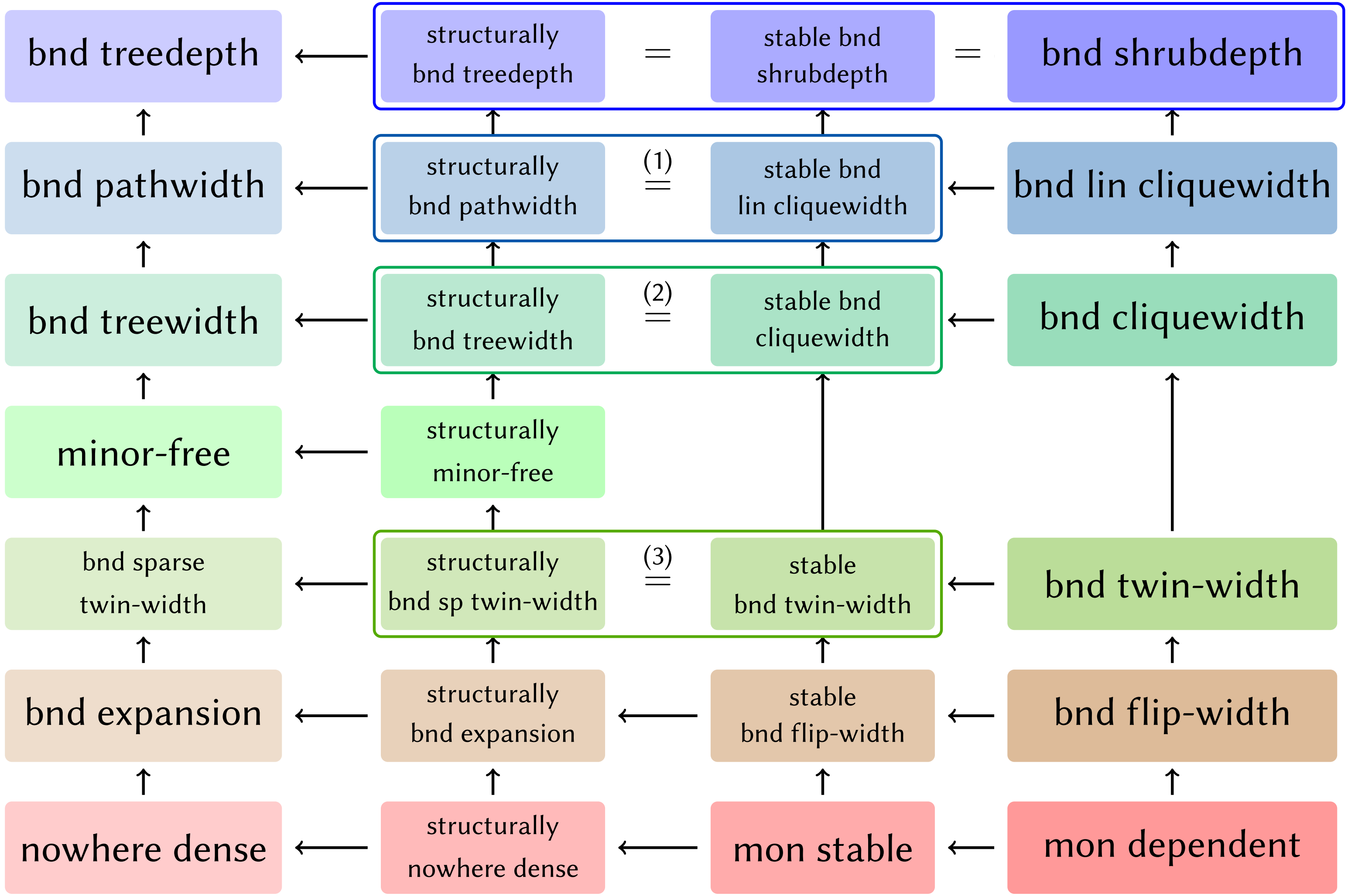

bnd treedepth
structurally bnd treedepth
stable bnd shrubdepth
bnd shrubdepth
bnd pathwidth
structurally bnd pathwidth
stable bnd lin cliquewidth
bnd lin cliquewidth
bnd treewidth
structurally bnd treewidth
stable bnd cliquewidth
bnd cliquewidth
minor-free
structurally minor-free
bnd sparse twin-width
structurally bnd sp twin-width
stable bnd twin-width
bnd twin-width
bnd expansion
structurally bnd expansion
stable bnd flip-width
bnd flip-width
nowhere dense
structurally nowhere dense
mon stable
mon dependent


bnd treedepth
structurally bnd treedepth
=
stable bnd shrubdepth
=
bnd shrubdepth
bnd pathwidth
structurally bnd pathwidth
(1)
=
stable bnd lin cliquewidth
bnd lin cliquewidth
bnd treewidth
structurally bnd treewidth
stable bnd cliquewidth
bnd cliquewidth
minor-free
structurally minor-free
bnd sparse twin-width
structurally bnd sp twin-width
stable bnd twin-width
bnd twin-width
bnd expansion
structurally bnd expansion
stable bnd flip-width
bnd flip-width
nowhere dense
structurally nowhere dense
mon stable
mon dependent

(1): [NOdMRS'20]



(1): [NOdMRS'20] (2): [NOdMPRS'21]



(1): [NOdMRS'20] (2): [NOdMPRS'21] (3): [GPT'22]

Main result

Main result

Theorem

If $\mathscr{C}$ is a class of **bnd twin-width** with a **stable edge relation**,

then $\mathscr{C}$ can be transduced from a class $\mathscr{D}$ of **bounded sparse twin-width**.

In fact, we can have $\mathscr{C} \equiv_{\mathrm{FO}} \mathscr{D}$.

Main result

Theorem

If $\mathscr{C}$ is a class of **bnd twin-width** with a **stable edge relation**,

then $\mathscr{C}$ can be transduced from a class $\mathscr{D}$ of **bounded sparse twin-width**.

In fact, we can have $\mathscr{C} \equiv_{\mathrm{FO}} \mathscr{D}$.

Intuition: Every $G \in \mathscr{C}$ can be **sparsified**.

We can find sparse $H \in \mathscr{D}$ in which G can be encoded.

Main result

Theorem

If $\mathscr{C}$ is a class of **bnd twin-width** with a **stable edge relation**,

then $\mathscr{C}$ can be transduced from a class $\mathscr{D}$ of **bounded sparse twin-width**.

In fact, we can have $\mathscr{C} \equiv_{\mathrm{FO}} \mathscr{D}$.

Intuition: Every $G \in \mathscr{C}$ can be **sparsified**.

We can find sparse $H \in \mathscr{D}$ in which G can be encoded.

Cor: stable bnd twin-width $\subseteq$ structurally bounded expansion.

Main result

Theorem

If $\mathscr{C}$ is a class of **bnd twin-width** with a **stable edge relation**,

then $\mathscr{C}$ can be transduced from a class $\mathscr{D}$ of **bounded sparse twin-width**.

In fact, we can have $\mathscr{C} \equiv_{\mathrm{FO}} \mathscr{D}$.

Intuition: Every $G \in \mathscr{C}$ can be **sparsified**.

We can find sparse $H \in \mathscr{D}$ in which G can be encoded.

Cor: stable bnd twin-width $\subseteq$ **structurally bounded expansion**.

Cor: If $\mathscr{C}$ has **stable bnd twin-width**, then $\mathscr{C}$ is **linearly χ-bounded**:

$$\chi(G) \leqslant c \cdot \omega(G) \qquad \text{for all } G \in \mathscr{C}.$$

Main result

Theorem

If $\mathscr{C}$ is a class of **bnd twin-width** with a **stable edge relation**,

then $\mathscr{C}$ can be transduced from a class $\mathscr{D}$ of **bounded sparse twin-width**.

In fact, we can have $\mathscr{C} \equiv_{\mathrm{FO}} \mathscr{D}$.

Intuition: Every $G \in \mathscr{C}$ can be **sparsified**.

We can find sparse $H \in \mathscr{D}$ in which G can be encoded.

Cor: **stable bnd twin-width** $\subseteq$ **structurally bounded expansion**.

Cor: If $\mathscr{C}$ has **stable bnd twin-width**, then $\mathscr{C}$ is **linearly χ-bounded**:

$$\chi(G) \leqslant c \cdot \omega(G) \qquad \text{for all } G \in \mathscr{C}.$$

Now: Proof of the last corollary.

Main result

Theorem

If $\mathscr{C}$ is a class of **bnd twin-width** with a **stable edge relation**,

then $\mathscr{C}$ can be transduced from a class $\mathscr{D}$ of **bounded sparse twin-width**.

In fact, we can have $\mathscr{C} \equiv_{\mathrm{FO}} \mathscr{D}$.

Intuition: Every $G \in \mathscr{C}$ can be **sparsified**.

We can find sparse $H \in \mathscr{D}$ in which G can be encoded.

Cor: stable bnd twin-width $\subseteq$ **structurally bounded expansion**.

Cor: If $\mathscr{C}$ has **stable bnd twin-width**, then $\mathscr{C}$ is **linearly χ-bounded**:

$$\chi(G) \leqslant c \cdot \omega(G) \qquad \text{for all } G \in \mathscr{C}.$$

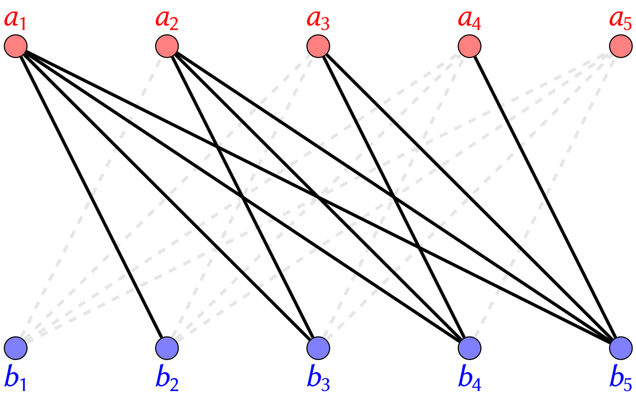
Now: Proof of the last corollary.

— Baby case of the proof of the main theorem.

Index

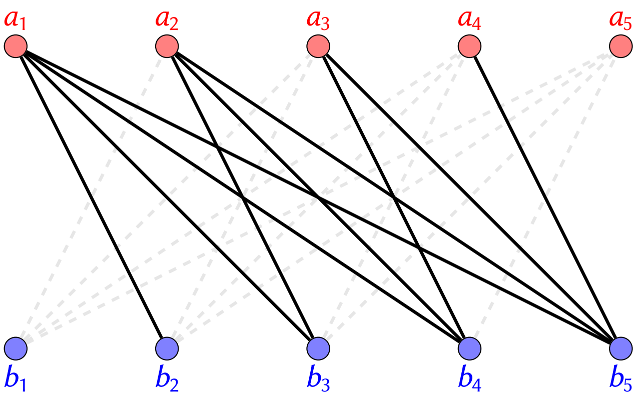
Index

Def: **Index** of G is the largest order of the following structure in G:



Index

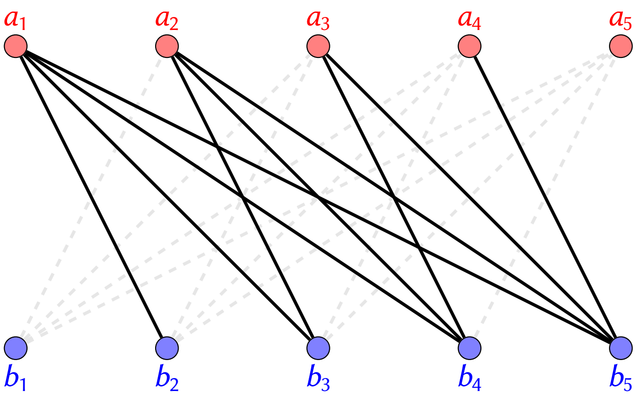
Def: Index of G is the largest order of the following structure in G:



$$(i < j) \quad \Rightarrow \quad a_i \text{ and } b_j \text{ adjacent}$$

Index

Def: **Index** of G is the largest order of the following structure in G:

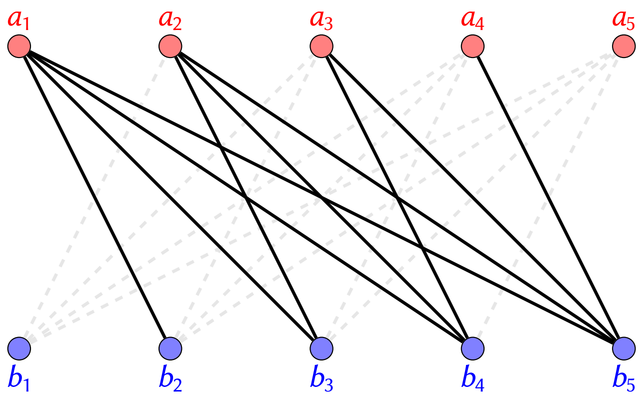


$(i < j) \quad \Rightarrow \quad a_i$ and b_j adjacent

$(i > j) \quad \Rightarrow \quad a_i$ and b_j non-adjacent

Index

Def: **Index** of G is the largest order of the following structure in G:



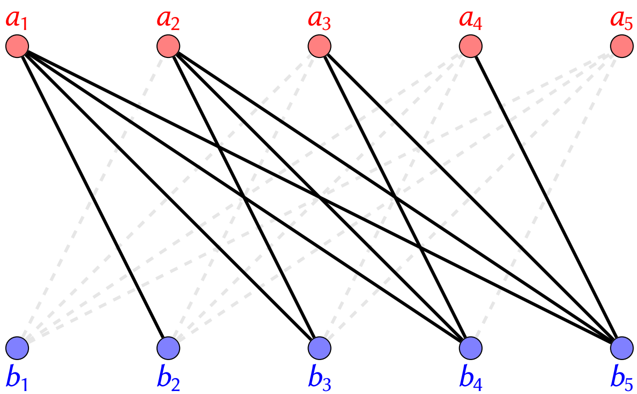
$(i < j) \quad \Rightarrow \quad a_i$ and b_j adjacent

$(i > j) \quad \Rightarrow \quad a_i$ and b_j non-adjacent

$(i = j) \quad \Rightarrow \quad$ no requirement

Index

Def: **Index** of G is the largest order of the following structure in G:



$(i < j) \Rightarrow$ a_i and b_j adjacent

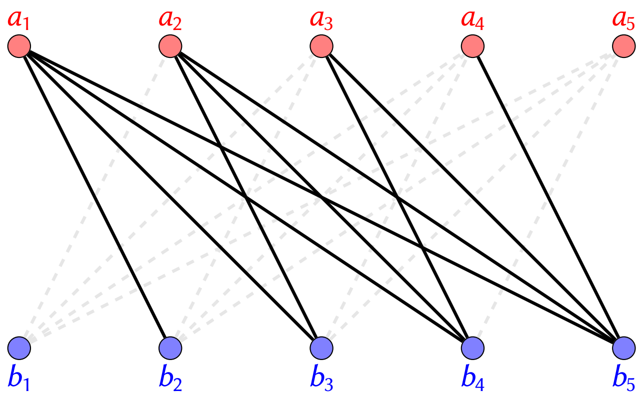
$(i > j) \Rightarrow$ a_i and b_j non-adjacent

$(i = j) \Rightarrow$ no requirement

within $\{a_i\} \Rightarrow$ no requirement

Index

Def: Index of G is the largest order of the following structure in G:



$(i < j) \Rightarrow$ a_i and b_j adjacent

$(i > j) \Rightarrow$ a_i and b_j non-adjacent

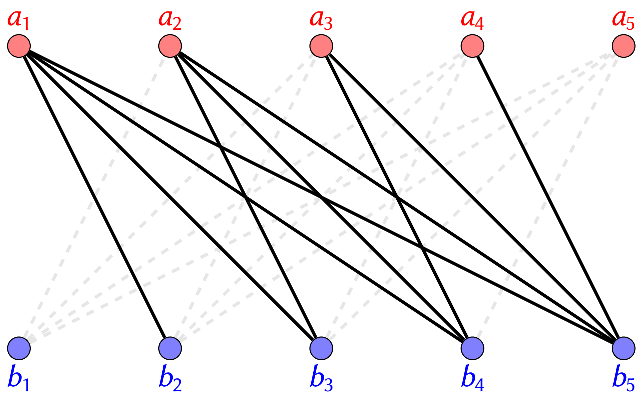
$(i = j) \Rightarrow$ no requirement

within $\{a_i\} \Rightarrow$ no requirement

within $\{b_i\} \Rightarrow$ no requirement

Index

Def: Index of G is the largest order of the following structure in G:



$(i < j) \Rightarrow$ a_i and b_j adjacent

$(i > j) \Rightarrow$ a_i and b_j non-adjacent

$(i = j) \Rightarrow$ no requirement

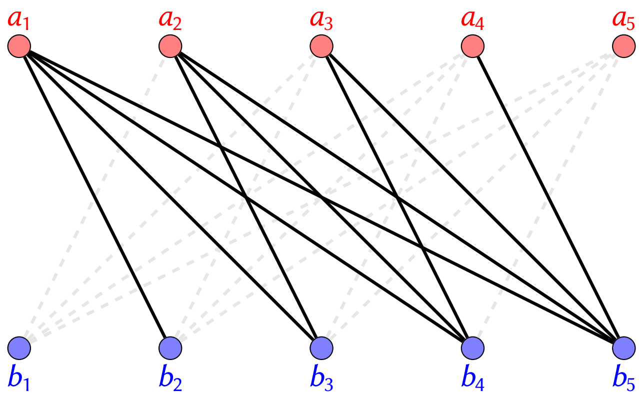
within $\{a_i\}$ $\Rightarrow$ no requirement

within $\{b_i\}$ $\Rightarrow$ no requirement

Obs: $\mathscr{C}$ has stable edge relation $\Rightarrow$ $\mathscr{C}$ has bounded index.

Index

Def: **Index** of G is the largest order of the following structure in G:



$(i < j) \Rightarrow$ a_i and b_j adjacent

$(i > j) \Rightarrow$ a_i and b_j non-adjacent

$(i = j) \Rightarrow$ no requirement

within $\{a_i\} \Rightarrow$ no requirement

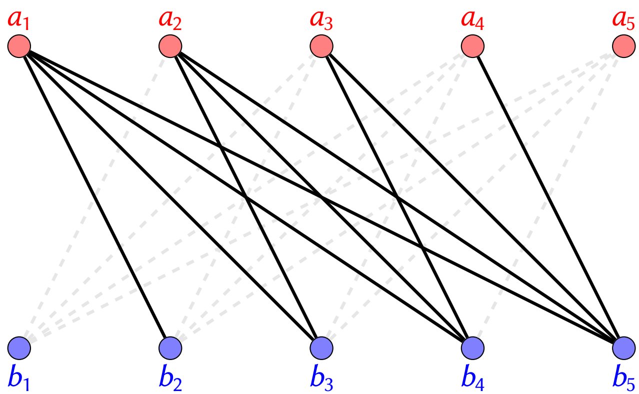
within $\{b_i\} \Rightarrow$ no requirement

Obs: $\mathscr{C}$ has stable edge relation $\Rightarrow$ $\mathscr{C}$ has bounded index.

Idea: Use index as a **progress measure**.

Index

Def: **Index** of G is the largest order of the following structure in G:



$(i < j) \Rightarrow$ a_i and b_j adjacent

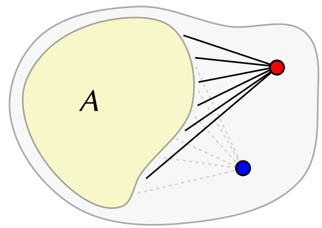
$(i > j) \Rightarrow$ a_i and b_j non-adjacent

$(i = j) \Rightarrow$ no requirement

within $\{a_i\} \Rightarrow$ no requirement

within $\{b_i\} \Rightarrow$ no requirement

Obs: $\mathscr{C}$ has stable edge relation $\Rightarrow$ $\mathscr{C}$ has bounded index.



Idea: Use index as a **progress measure**.

Obs: If $A \subseteq V(G)$ has a complete and an anti-complete vertex, then

$$\mathrm{index}(G[A]) < \mathrm{index}(G).$$

Partition into cographs

Partition into cographs

Lemma

If G has twin-width d and index k, then G can be colored with

$$(2d+4)^{k-1} \text{ colors}$$

so that every color induces a **cograph**.

Partition into cographs

Lemma

If G has twin-width d and index k, then G can be colored with

$$(2d+4)^{k-1} \text{ colors}$$

so that every color induces a **cograph**.

Cograph $=$ P_4-free graph

Partition into cographs

Lemma

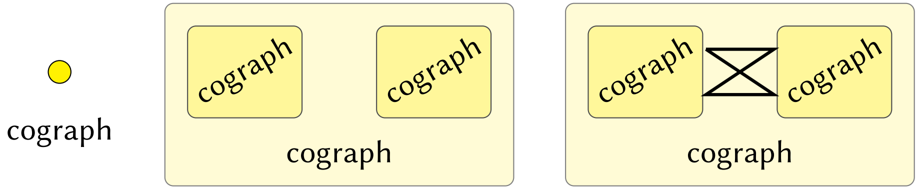
If G has twin-width d and index k, then G can be colored with

$$(2d+4)^{k-1} \text{ colors}$$

so that every color induces a **cograph**.

Cograph $=$ P_4-free graph

Recursive definition:



Partition into cographs

Lemma

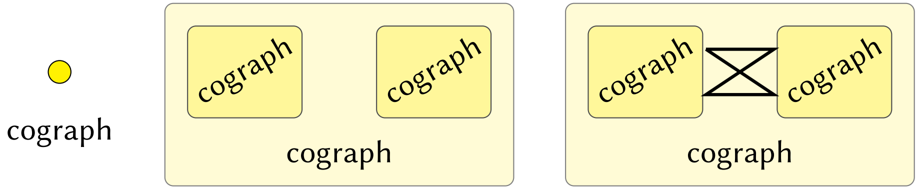
If G has twin-width d and index k, then G can be colored with

$$(2d + 4)^{k-1} \text{ colors}$$

so that every color induces a **cograph**.

Cograph $=$ P_4-free graph

Recursive definition:



Fact: Cographs are **perfect**: $\chi(H) = \omega(H)$ whenever H is a cograph.

Partition into cographs

Lemma

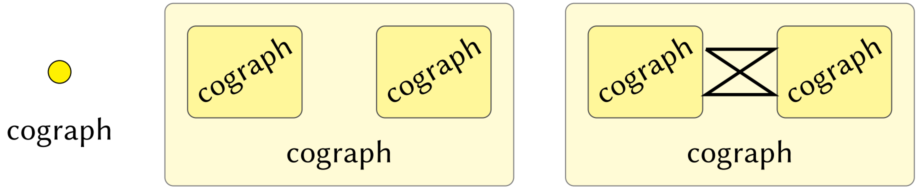
If G has twin-width d and index k, then G can be colored with

$$(2d+4)^{k-1} \text{ colors}$$

so that every color induces a **cograph**.

Cograph $= P_4$-free graph

Recursive definition:



Fact: Cographs are **perfect**: $\chi(H) = \omega(H)$ whenever H is a cograph.

Cor: Under the assumptions of **Lemma**, $\chi(G) \leqslant (2d+4)^{k-1} \cdot \omega(G)$.

Partition into cographs

Lemma

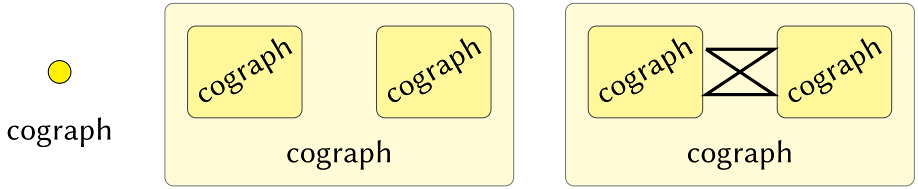
If G has twin-width d and index k, then G can be colored with

$$(2d+4)^{k-1} \text{ colors}$$

so that every color induces a **cograph**.

Cograph $= P_4$-free graph

Recursive definition:



Fact: Cographs are **perfect**: $\chi(H) = \omega(H)$ whenever H is a cograph.

Cor: Under the assumptions of **Lemma**, $\chi(G) \leqslant (2d+4)^{k-1} \cdot \omega(G)$.

Partition into cographs

Lemma

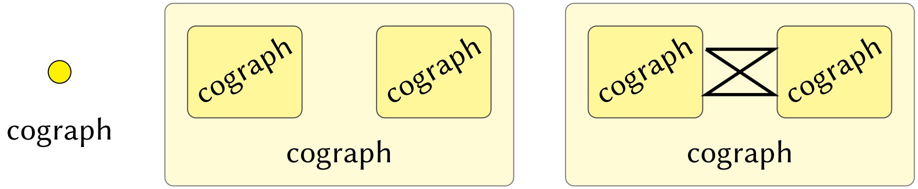
If G has twin-width d and index k, then G can be colored with

$$(2d+4)^{k-1} \text{ colors}$$

so that every color induces a **cograph**.

Cograph $= P_4$-free graph

Recursive definition:



Fact: Cographs are **perfect**: $\chi(H) = \omega(H)$ whenever H is a cograph.

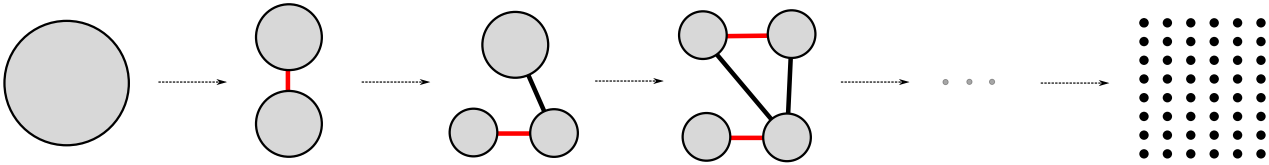
Cor: Under the assumptions of **Lemma**, $\chi(G) \leqslant (2d+4)^{k-1} \cdot \omega(G)$.

Idea: Induction on the index k.

Frozen bubbles

Frozen bubbles

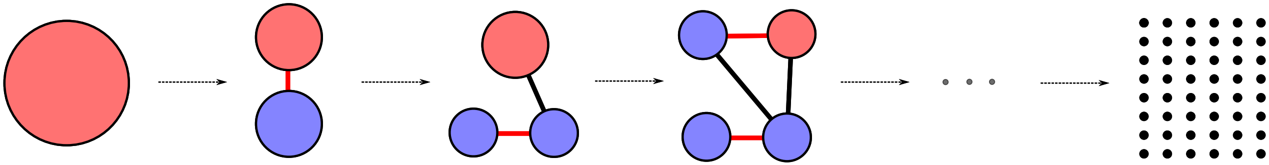
Consider an uncontraction sequence of width d.



$\mathcal{P}_1$ $\mathcal{P}_2$ $\mathcal{P}_3$ $\mathcal{P}_4$ $\mathcal{P}_n$

Frozen bubbles

Consider an uncontraction sequence of width d.

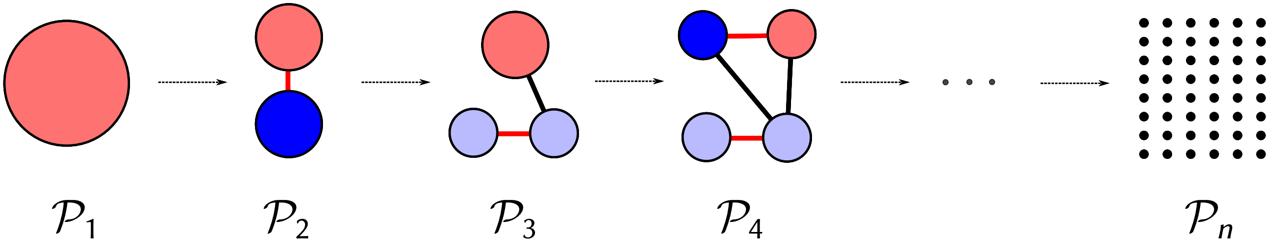


$\mathcal{P}_1$ $\mathcal{P}_2$ $\mathcal{P}_3$ $\mathcal{P}_4$ $\mathcal{P}_n$

A part $A \in \mathcal{P}_t$ is **light** if $\mathrm{index}(G[A]) < k$, and **heavy** otherwise.

Frozen bubbles

Consider an uncontraction sequence of width d.

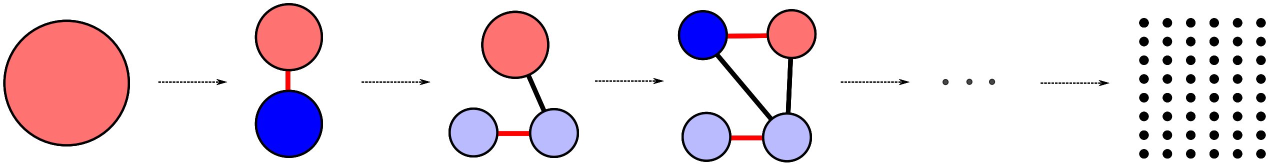


A part $A \in \mathcal{P}_t$ is **light** if $\mathrm{index}(G[A]) < k$, and **heavy** otherwise.

$A \in \mathcal{P}_t$ is **frozen** at time t if A is **light** but the parent $A' \in \mathcal{P}_{t-1}$ is **heavy**.

Frozen bubbles

Consider an uncontraction sequence of width d.



$\mathcal{P}_1$ $\mathcal{P}_2$ $\mathcal{P}_3$ $\mathcal{P}_4$ $\mathcal{P}_n$

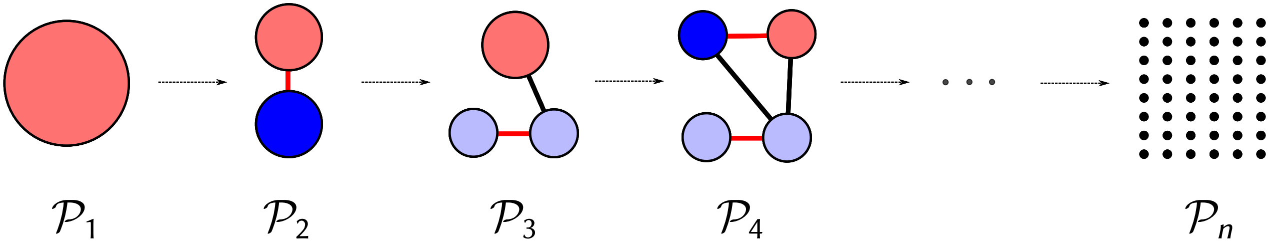
A part $A \in \mathcal{P}_t$ is **light** if $\text{index}(G[A]) < k$, and **heavy** otherwise.

$A \in \mathcal{P}_t$ is **frozen** at time t if A is **light** but the parent $A' \in \mathcal{P}_{t-1}$ is **heavy**.

- $\mathcal{F}_t :=$ parts frozen at time t.

Frozen bubbles

Consider an uncontraction sequence of width d.



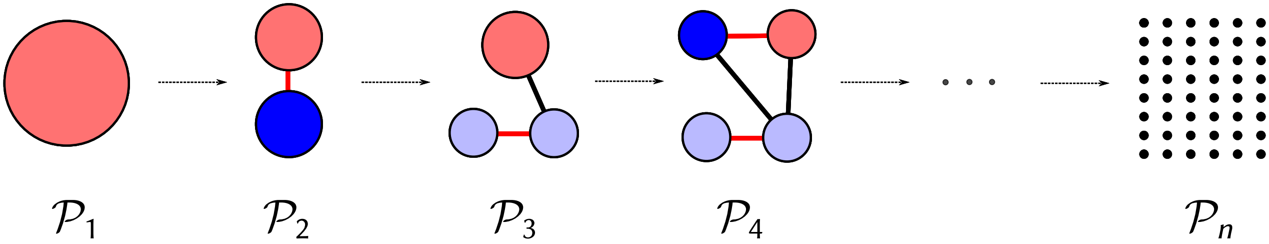
A part $A \in \mathcal{P}_t$ is **light** if $\mathrm{index}(G[A]) < k$, and **heavy** otherwise.

$A \in \mathcal{P}_t$ is **frozen** at time t if A is **light** but the parent $A' \in \mathcal{P}_{t-1}$ is **heavy**.

- $\mathcal{F}_t :=$ parts frozen at time t.
- **Note:** $|\mathcal{F}_t| \leqslant 2$.

Frozen bubbles

Consider an uncontraction sequence of width d.



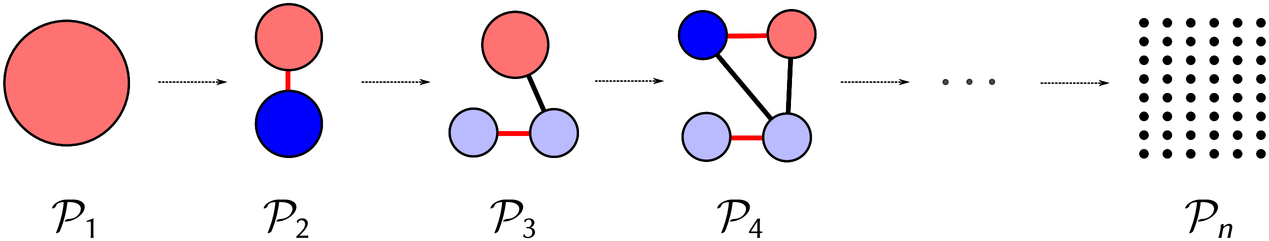
A part $A \in \mathcal{P}_t$ is **light** if $\text{index}(G[A]) < k$, and **heavy** otherwise.

$A \in \mathcal{P}_t$ is **frozen** at time t if A is **light** but the parent $A' \in \mathcal{P}_{t-1}$ is **heavy**.

- $\mathcal{F}_t :=$ parts frozen at time t.
- **Note:** $|\mathcal{F}_t| \leqslant 2$.
- $\mathcal{F} := \bigcup_{1 \leqslant t \leqslant n} \mathcal{F}_t$.

Frozen bubbles

Consider an uncontraction sequence of width d.



A part $A \in \mathcal{P}_t$ is **light** if $\mathrm{index}(G[A]) < k$, and **heavy** otherwise.

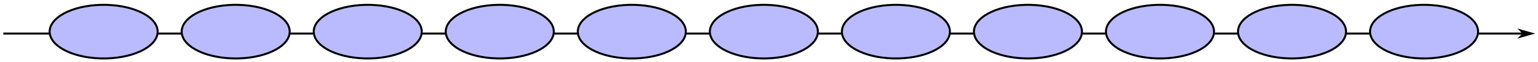
$A \in \mathcal{P}_t$ is **frozen** at time t if A is **light** but the parent $A' \in \mathcal{P}_{t-1}$ is **heavy**.

- $\mathcal{F}_t :=$ parts frozen at time t.
- **Note:** $|\mathcal{F}_t| \leqslant 2$.
- $\mathcal{F} := \bigcup_{1 \leqslant t \leqslant n} \mathcal{F}_t$.
- **Note:** $\mathcal{F}$ is a partition of the vertex set.

Ordering bubbles

Ordering bubbles

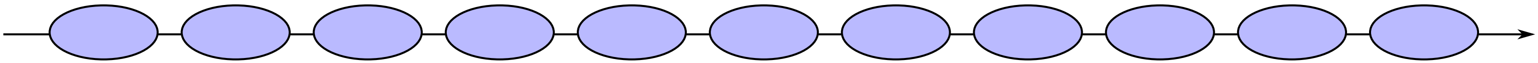
Order $\mathcal{F}$ by the freezing times.



Ordering bubbles

Order $\mathcal{F}$ by the freezing times.

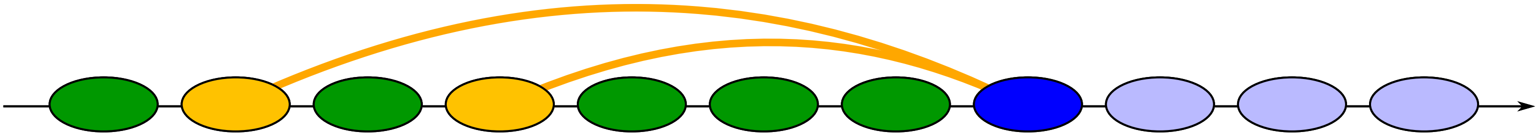
Idea: Adjacency between bubbles has a specific structure.



Ordering bubbles

Order $\mathcal{F}$ by the freezing times.

Idea: Adjacency between bubbles has a specific structure.



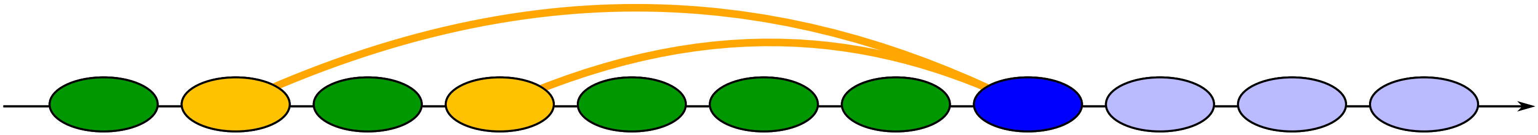
Lemma

For every $B \in \mathcal{F}$ there is a set $\mathcal{E}_B$ of at most $d + 1$ earlier bubbles such that B is homogeneous towards $\bigcup\{A\colon A \prec B, A \notin \mathcal{E}_B\}$.

Ordering bubbles

Order $\mathcal{F}$ by the freezing times.

Idea: Adjacency between bubbles has a specific structure.



Lemma

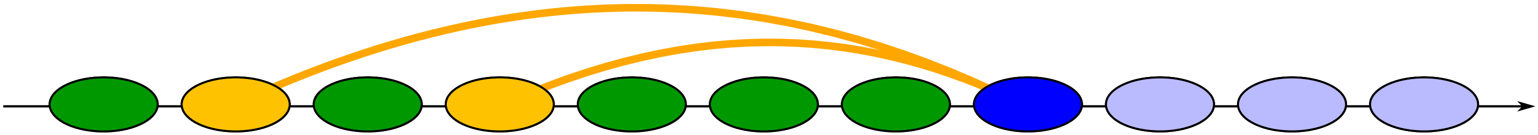
For every $B \in \mathcal{F}$ there is a set $\mathcal{E}_B$ of at most $d+1$ earlier bubbles such that B is homogeneous towards $\bigcup\{A\colon A \prec B, A \notin \mathcal{E}_B\}$.

Proof: look ☞

Ordering bubbles

Order $\mathcal{F}$ by the freezing times.

Idea: Adjacency between bubbles has a specific structure.



Lemma

For every $B \in \mathcal{F}$ there is a set $\mathcal{E}_B$ of at most $d+1$ earlier bubbles such that B is homogeneous towards $\bigcup\{A \colon A \prec B, A \notin \mathcal{E}_B\}$.

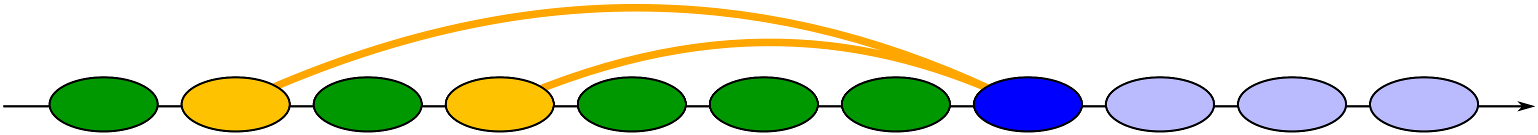
Proof: look ☞

– $B' :=$ parent of B, say $B' \in \mathcal{P}_t$.

Ordering bubbles

Order $\mathcal{F}$ by the freezing times.

Idea: Adjacency between bubbles has a specific structure.



Lemma

For every $B \in \mathcal{F}$ there is a set $\mathcal{E}_B$ of at most $d+1$ earlier bubbles such that B is homogeneous towards $\bigcup\{A\colon A \prec B, A \notin \mathcal{E}_B\}$.

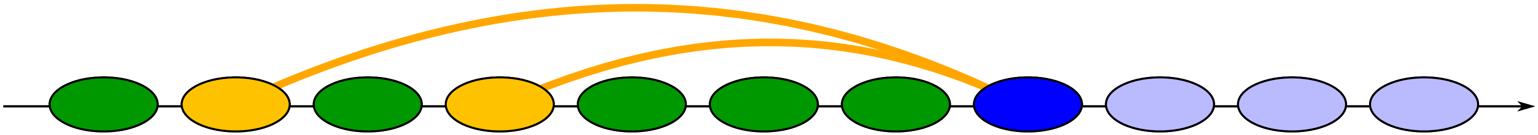
Proof: look ☞

- $B' :=$ parent of B, say $B' \in \mathcal{P}_t$.
- $\mathcal{N} :=$ red neighbors of B' at time t.

Ordering bubbles

Order $\mathcal{F}$ by the freezing times.

Idea: Adjacency between bubbles has a specific structure.



Lemma

For every $B \in \mathcal{F}$ there is a set $\mathcal{E}_B$ of at most $d+1$ earlier bubbles such that B is homogeneous towards $\bigcup\{A \colon A \prec B, A \notin \mathcal{E}_B\}$.

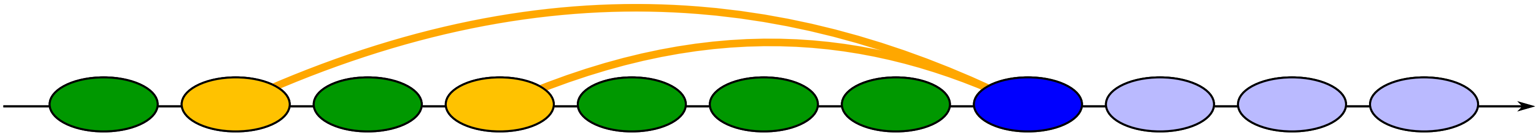
Proof: look ☞

- $B' :=$ parent of B, say $B' \in \mathcal{P}_t$.
- $\mathcal{N} :=$ red neighbors of B' at time t.
- **Note:** Every $u \notin B' \cup \bigcup \mathcal{N}$ is homogeneous towards B'.

Ordering bubbles

Order $\mathcal{F}$ by the freezing times.

Idea: Adjacency between bubbles has a specific structure.



Lemma

For every $B \in \mathcal{F}$ there is a set $\mathcal{E}_B$ of at most $d+1$ earlier bubbles such that B is homogeneous towards $\bigcup\{A\colon A \prec B, A \notin \mathcal{E}_B\}$.

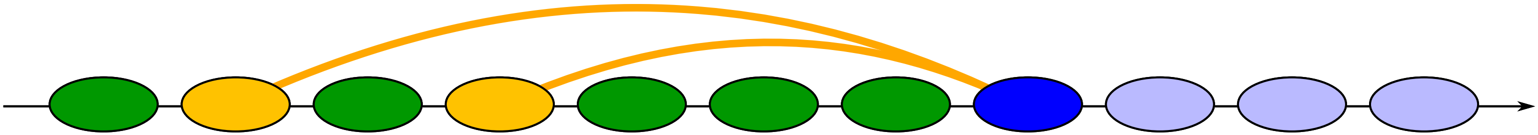
Proof: look ☞

- $B' :=$ parent of B, say $B' \in \mathcal{P}_t$.
- $\mathcal{N} :=$ red neighbors of B' at time t.
- **Note:** Every $u \notin B' \cup \bigcup \mathcal{N}$ is homogeneous towards B'.
- **Note:** B' is **heavy** $\Rightarrow$ All homogeneity of same type.

Ordering bubbles

Order $\mathcal{F}$ by the freezing times.

Idea: Adjacency between bubbles has a specific structure.



Lemma

For every $B \in \mathcal{F}$ there is a set $\mathcal{E}_B$ of at most $d+1$ earlier bubbles such that B is homogeneous towards $\bigcup\{A\colon A \prec B, A \notin \mathcal{E}_B\}$.

Proof: look ☞

- $B' :=$ parent of B, say $B' \in \mathcal{P}_t$.
- $\mathcal{N} :=$ red neighbors of B' at time t.
- **Note:** Every $u \notin B' \cup \bigcup \mathcal{N}$ is homogeneous towards B'.
- **Note:** B' is **heavy** $\Rightarrow$ All homogeneity of same type.
- $\mathcal{E}_B :=$ frozen ancestors of $\mathcal{N}$ and maybe sibling of B. $\square$

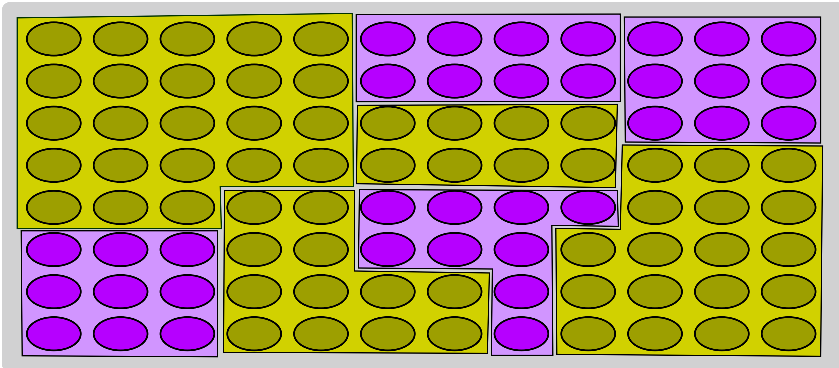
Coloring bubbles

Coloring bubbles

Partition $\mathcal{F}$ into $2 \cdot (d+2)$ groups:

– **Degeneracy** coloring with $d+2$ colors $\rightsquigarrow$ No exceptions within a group.

– **Homogeneity type** $+$ or $-$ $\rightsquigarrow$ Every group of same homogeneity type.

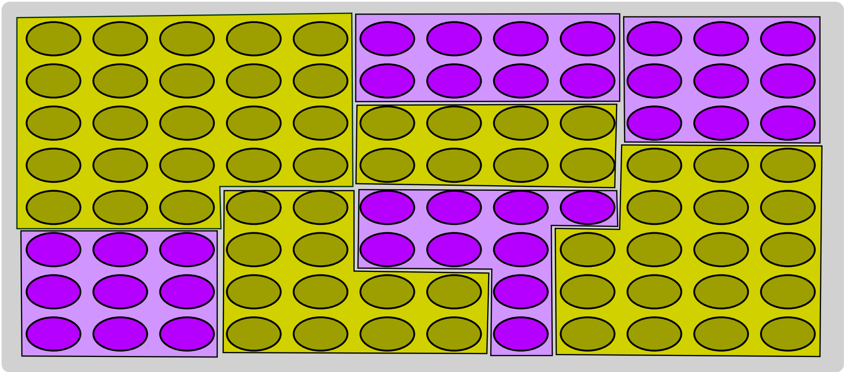


Coloring bubbles

Partition $\mathcal{F}$ into $2 \cdot (d+2)$ groups:

– **Degeneracy** coloring with $d+2$ colors $\rightsquigarrow$ No exceptions within a group.

– **Homogeneity type** $+$ or $-$ $\rightsquigarrow$ Every group of same homogeneity type.



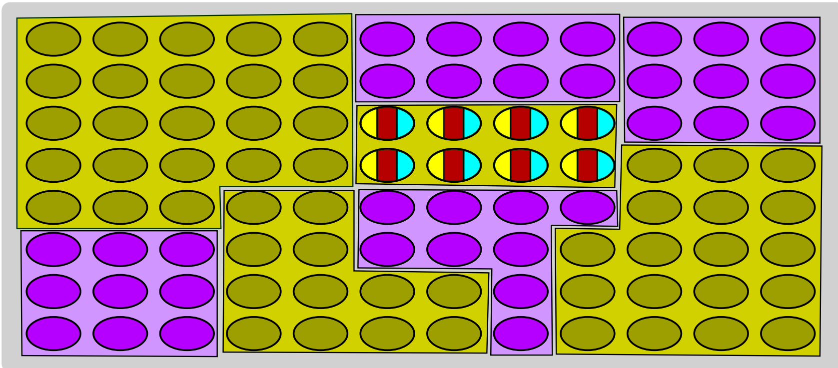
Within every group, bubbles pairwise **complete** or pairwise **anticomplete**.

Coloring bubbles

Partition $\mathcal{F}$ into $2 \cdot (d+2)$ groups:

– **Degeneracy** coloring with $d+2$ colors $\rightsquigarrow$ No exceptions within a group.

– **Homogeneity type** $+$ or $-$ $\rightsquigarrow$ Every group of same homogeneity type.



Within every group, bubbles pairwise **complete** or pairwise **anticomplete**.

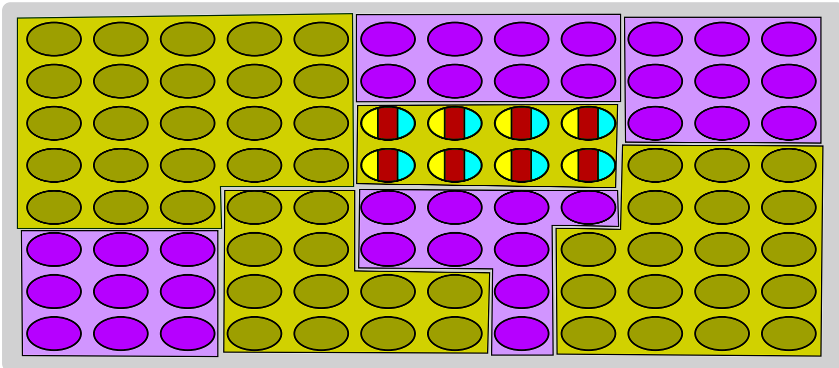
Apply induction on each $B \in \mathcal{F} \rightsquigarrow$ Cograph coloring with $f(k-1)$ colors.

Coloring bubbles

Partition $\mathcal{F}$ into $2 \cdot (d+2)$ groups:

– **Degeneracy** coloring with $d+2$ colors $\rightsquigarrow$ No exceptions within a group.

– **Homogeneity type** $+$ or $-$ $\rightsquigarrow$ Every group of same homogeneity type.



Within every group, bubbles pairwise **complete** or pairwise **anticomplete**.

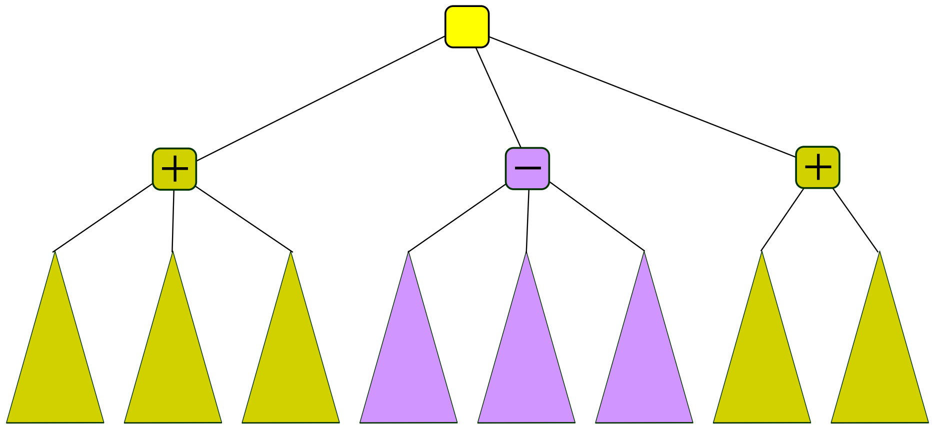
Apply induction on each $B \in \mathcal{F} \rightsquigarrow$ Cograph coloring with $f(k-1)$ colors.

Use $2d+4$ palettes of size $f(k-1) \rightsquigarrow (2d+4) \cdot f(k-1)$ colors in total. $\square$

Decomposition

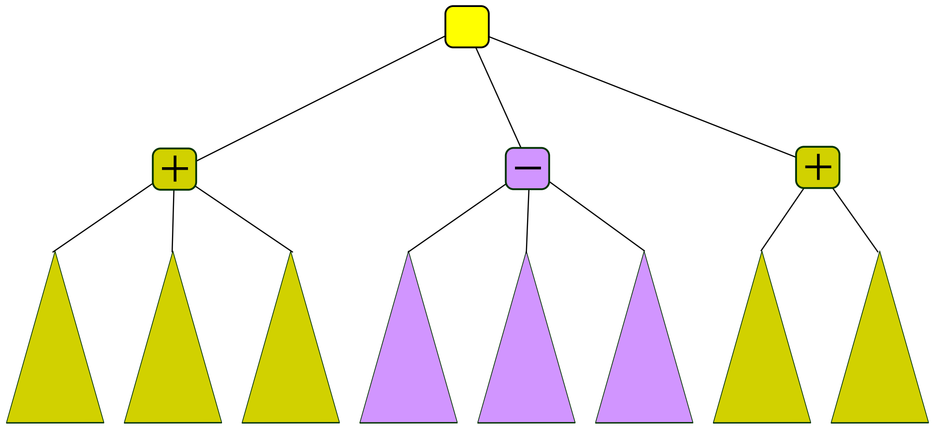
Decomposition

We got sort of a decomposition:



Decomposition

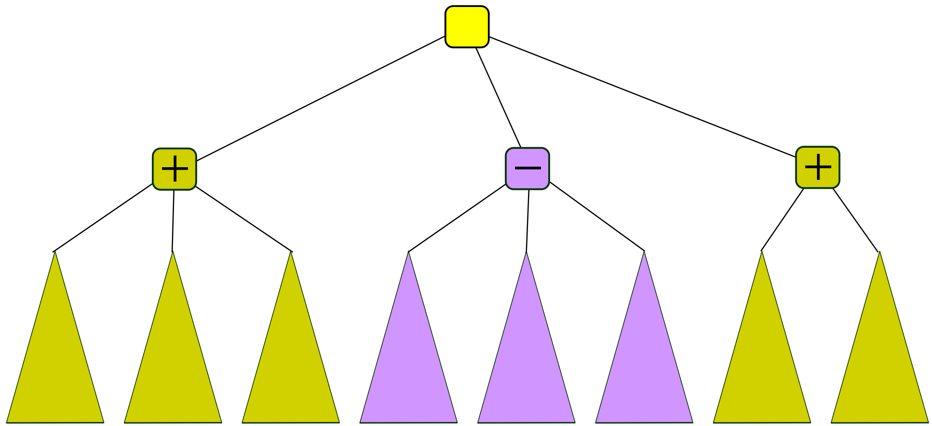
We got sort of a decomposition:



Problem: We don't control edges between groups.

Decomposition

We got sort of a decomposition:

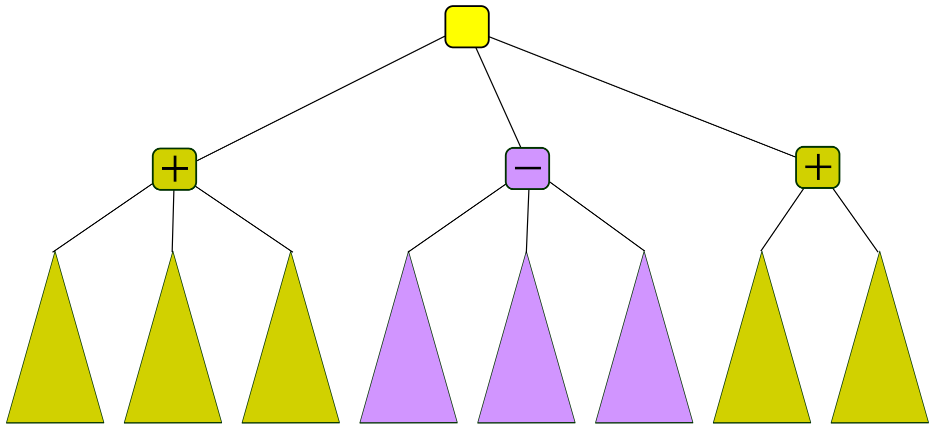


Problem: We don't control edges between groups.

Idea: Induct on **pairs** of bubbles.

Decomposition

We got sort of a decomposition:



Problem: We don't control edges between groups.

Idea: Induct on **pairs** of bubbles.

– Pair of bubbles A, B is simpler if $\mathrm{index}(G[A, B]) < k$.

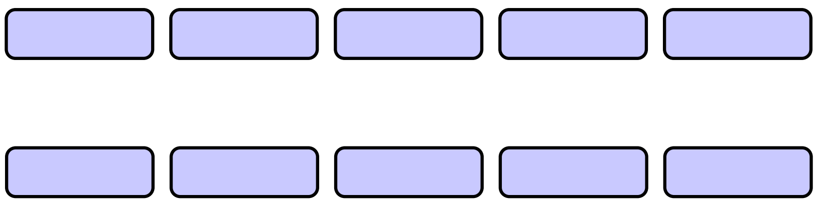
General case

General case

Lemma

Suppose G is a **bipartite** graph of bipartite twin-width d and index k.

Then one can partition $V(G)$ into $\mathcal{F}$ respecting sides so that:



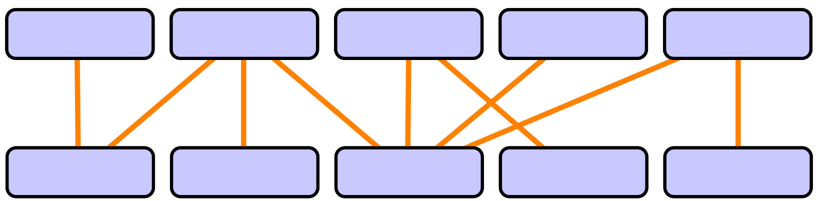
General case

Lemma

Suppose G is a **bipartite** graph of bipartite twin-width d and index k.

Then one can partition $V(G)$ into $\mathcal{F}$ respecting sides so that:

– On $\mathcal{F}$ there is an **exception graph** H.



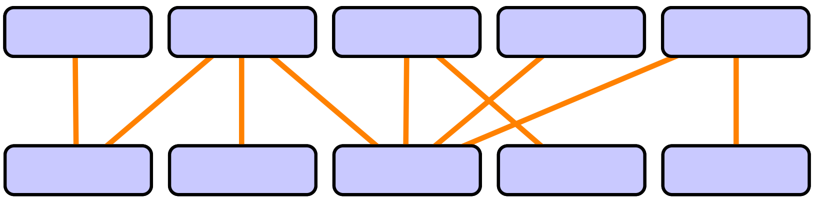
General case

Lemma

Suppose G is a **bipartite** graph of bipartite twin-width d and index k.

Then one can partition $V(G)$ into $\mathcal{F}$ respecting sides so that:

- On $\mathcal{F}$ there is an **exception graph** H.
- H has **star chromatic number** bounded by $p = p(d, k)$.



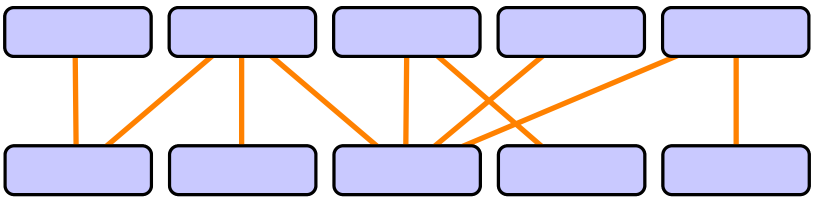
General case

Lemma

Suppose G is a **bipartite** graph of bipartite twin-width d and index k.

Then one can partition $V(G)$ into $\mathcal{F}$ respecting sides so that:

- On $\mathcal{F}$ there is an **exception graph** H.
- H has **star chromatic number** bounded by $p = p(d, k)$.
- Each star in each induced star forest of the above has index $< k$.



General case

Lemma

Suppose G is a **bipartite** graph of bipartite twin-width d and index k.

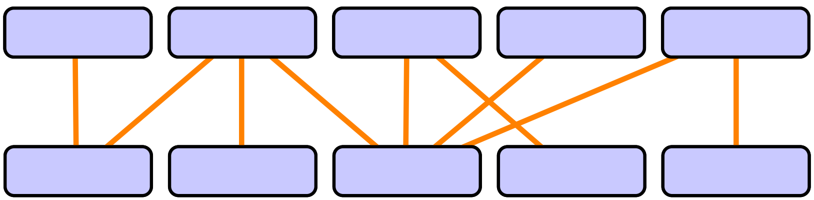
Then one can partition $V(G)$ into $\mathcal{F}$ respecting sides so that:

– On $\mathcal{F}$ there is an **exception graph** H.

– H has **star chromatic number** bounded by $p = p(d, k)$.

– Each star in each induced star forest of the above has index $< k$.

– All non-exceptional pairs of $A, B \in \mathcal{F}$ are homogeneous.



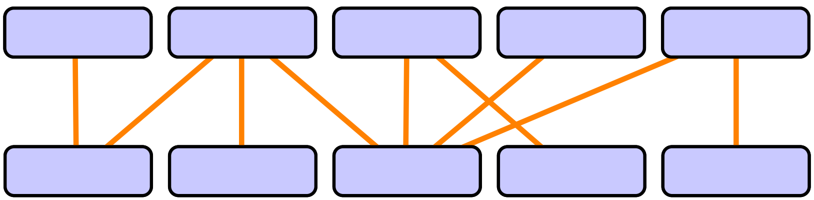
General case

Lemma

Suppose G is a **bipartite** graph of bipartite twin-width d and index k.

Then one can partition $V(G)$ into $\mathcal{F}$ respecting sides so that:

- On $\mathcal{F}$ there is an **exception graph** H.
- H has **star chromatic number** bounded by $p = p(d, k)$.
- Each star in each induced star forest of the above has index $< k$.
- All non-exceptional pairs of $A, B \in \mathcal{F}$ are homogeneous.
- Complete pairs $AB \notin E(H)$ can be cleared using $q = q(d, k)$ **flips**.



Glimpse into the proof

Glimpse into the proof

Freezing condition:

$A \in \mathcal{P}_t$ gets frozen if $\text{index}(G[A, B]) < k$ for all $B \in \mathcal{P}_t$ on the other side.

Glimpse into the proof

Freezing condition:

$A \in \mathcal{P}_t$ gets frozen if $\mathrm{index}(G[A, B]) < k$ for all $B \in \mathcal{P}_t$ on the other side.

This gives rise to the **exception graph** H.

Glimpse into the proof

Freezing condition:

$A \in \mathcal{P}_t$ gets frozen if $\mathrm{index}(G[A, B]) < k$ for all $B \in \mathcal{P}_t$ on the other side.

This gives rise to the **exception graph** H.

– H has bounded wcol_2.

Glimpse into the proof

Freezing condition:

$A \in \mathcal{P}_t$ gets frozen if $\text{index}(G[A, B]) < k$ for all $B \in \mathcal{P}_t$ on the other side.

This gives rise to the **exception graph** H.

– H has bounded wcol_2.

– **Ergo:** H has bounded **star chromatic number**.

Glimpse into the proof

Freezing condition:

$A \in \mathcal{P}_t$ gets frozen if $\mathrm{index}(G[A,B]) < k$ for all $B \in \mathcal{P}_t$ on the other side.

This gives rise to the **exception graph** H.

- H has bounded wcol_2.
- **Ergo:** H has bounded **star chromatic number**.
- A bit of work with bounding the index of stars.

Glimpse into the proof

Freezing condition:

$A \in \mathcal{P}_t$ gets frozen if $\mathrm{index}(G[A, B]) < k$ for all $B \in \mathcal{P}_t$ on the other side.

This gives rise to the **exception graph** H.

- H has bounded wcol_2.
- **Ergo:** H has bounded **star chromatic number**.
- A bit of work with bounding the index of stars.

More work with flipping away the complete pairs. $\square$

Glimpse into the proof

Freezing condition:

$A \in \mathcal{P}_t$ gets frozen if $\mathrm{index}(G[A, B]) < k$ for all $B \in \mathcal{P}_t$ on the other side.

This gives rise to the **exception graph** H.

- H has bounded wcol_2.
- **Ergo:** H has bounded **star chromatic number**.
- A bit of work with bounding the index of stars.

More work with flipping away the complete pairs. $\square$

Different **freezing conditions** give different **structure** between bubbles.

Glimpse into the proof

Freezing condition:

$A \in \mathcal{P}_t$ gets frozen if $\mathrm{index}(G[A, B]) < k$ for all $B \in \mathcal{P}_t$ on the other side.

This gives rise to the **exception graph** H.

- H has bounded wcol_2.
- **Ergo:** H has bounded **star chromatic number**.
- A bit of work with bounding the index of stars.

More work with flipping away the complete pairs. □

Different **freezing conditions** give different **structure** between bubbles.

- χ-boundedness $\rightsquigarrow$ freeze when ω drops $\rightsquigarrow$ quotient is sparse.

Glimpse into the proof

Freezing condition:

$A \in \mathcal{P}_t$ gets frozen if $\text{index}(G[A, B]) < k$ for all $B \in \mathcal{P}_t$ on the other side.

This gives rise to the **exception graph** H.

- H has bounded wcol_2.
- **Ergo:** H has bounded **star chromatic number**.
- A bit of work with bounding the index of stars.

More work with flipping away the complete pairs. $\square$

Different **freezing conditions** give different **structure** between bubbles.

- χ-boundedness $\rightsquigarrow$ freeze when ω drops $\rightsquigarrow$ quotient is sparse.
- qpoly χ-boundedness $\rightsquigarrow$ freeze when ω drops by 1%.

Glimpse into the proof

Freezing condition:

$A \in \mathcal{P}_t$ gets frozen if $\text{index}(G[A, B]) < k$ for all $B \in \mathcal{P}_t$ on the other side.

This gives rise to the **exception graph** H.

- H has bounded wcol_2.
- **Ergo:** H has bounded **star chromatic number**.
- A bit of work with bounding the index of stars.

More work with flipping away the complete pairs. $\square$

Different **freezing conditions** give different **structure** between bubbles.

- χ-boundedness $\rightsquigarrow$ freeze when ω drops $\rightsquigarrow$ quotient is sparse.
- qpoly χ-boundedness $\rightsquigarrow$ freeze when ω drops by 1%.
- neighborhood covers $\rightsquigarrow$ freeze when there is a universal vertex.

Sketch of main proof

Goal: Find **sparse** G' of bnd twin-width from which G can be transduced.

Sketch of main proof

Goal: Find **sparse** G' of bnd twin-width from which G can be transduced.

Wlog we can work with bipartite graphs.

Sketch of main proof

Goal: Find **sparse** G' of bnd twin-width from which G can be transduced.

Wlog we can work with bipartite graphs.

Apply **Lemma**, recurse on all stars in each star forest.

Sketch of main proof

Goal: Find **sparse** G' of bnd twin-width from which G can be transduced.

Wlog we can work with bipartite graphs.

Apply **Lemma**, recurse on all stars in each star forest.

Each application encoded by unary predicates and **equivalence relations**.

Sketch of main proof

Goal: Find **sparse** G' of bnd twin-width from which G can be transduced.

Wlog we can work with bipartite graphs.

Apply **Lemma**, recurse on all stars in each star forest.
Each application encoded by unary predicates and **equivalence relations**.

Final: Structure D consisting of $t = t(d, k)$ unary predicates and equivalence relations from which G can be interpreted.

Sketch of main proof

Goal: Find **sparse** G' of bnd twin-width from which G can be transduced.

Wlog we can work with bipartite graphs.

Apply **Lemma**, recurse on all stars in each star forest.

Each application encoded by unary predicates and **equivalence relations**.

Final: Structure D consisting of $t = t(d, k)$ unary predicates and equivalence relations from which G can be interpreted.

D can be represented as a **sparse** graph G' from which G can be transduced.

Sketch of main proof

Goal: Find **sparse** G' of bnd twin-width from which G can be transduced.

Wlog we can work with bipartite graphs.

Apply **Lemma**, recurse on all stars in each star forest.

Each application encoded by unary predicates and **equivalence relations**.

Final: Structure D consisting of $t = t(d, k)$ unary predicates and equivalence relations from which G can be interpreted.

D can be represented as a **sparse** graph G' from which G can be transduced.

– Just replace each **equivalence relation** with a **star forest**.

Sketch of main proof

Goal: Find **sparse** G' of bnd twin-width from which G can be transduced.

Wlog we can work with bipartite graphs.

Apply **Lemma**, recurse on all stars in each star forest.

Each application encoded by unary predicates and **equivalence relations**.

Final: Structure D consisting of $t = t(d, k)$ unary predicates and equivalence relations from which G can be interpreted.

D can be represented as a **sparse** graph G' from which G can be transduced.

– Just replace each **equivalence relation** with a **star forest**.

Issue: Why does G' have bounded twin-width?

Sketch of main proof

Goal: Find **sparse** G' of bnd twin-width from which G can be transduced.

Wlog we can work with bipartite graphs.

Apply **Lemma**, recurse on all stars in each star forest.

Each application encoded by unary predicates and **equivalence relations**.

Final: Structure D consisting of $t = t(d, k)$ unary predicates and equivalence relations from which G can be interpreted.

D can be represented as a **sparse** graph G' from which G can be transduced.

– Just replace each **equivalence relation** with a **star forest**.

Issue: Why does G' have bounded twin-width?

– G' can be transduced from $(G, \leqslant)$, where $\leqslant$ witnesses bnd tww of G.

Open problems

Open problems

str bnd expansion $\stackrel{?}{=}$ stable bnd flip-width

str nowhere dense $\stackrel{?}{=}$ mon stable

Open problems

str bnd expansion $\stackrel{?}{=}$ stable bnd flip-width

str nowhere dense $\stackrel{?}{=}$ mon stable

Def: For a property Π, let $\widehat{\Pi}$ be the largest transduction ideal such that

$$\Pi = \widehat{\Pi} \cap \text{weakly sparse.}$$

Open problems

str bnd expansion $\stackrel{?}{=}$ **stable bnd flip-width**

str nowhere dense $\stackrel{?}{=}$ **mon stable**

Def: For a property Π, let $\widehat{\Pi}$ be the largest transduction ideal such that

$$\Pi = \widehat{\Pi} \cap \text{weakly sparse.}$$

Does this exactly map the **sparse** column to the **dependent** column?

Open problems

str bnd expansion $\stackrel{?}{=}$ **stable bnd flip-width**

str nowhere dense $\stackrel{?}{=}$ **mon stable**

Def: For a property Π, let $\widehat{\Pi}$ be the largest transduction ideal such that

$$\Pi = \widehat{\Pi} \cap \text{weakly sparse.}$$

Does this exactly map the **sparse** column to the **dependent** column?

Conjecture: $\mathscr{C}$ has **unbounded cliquewidth**

$\Leftrightarrow$ $\mathscr{C}$ transduces a class that contains a subdivision of every **wall**.

Open problems

str bnd expansion $\stackrel{?}{=}$ **stable bnd flip-width**

str nowhere dense $\stackrel{?}{=}$ **mon stable**

Def: For a property Π, let $\widehat{\Pi}$ be the largest transduction ideal such that

$$\Pi = \widehat{\Pi} \cap \text{weakly sparse}.$$

Does this exactly map the **sparse** column to the **dependent** column?

Conjecture: $\mathscr{C}$ has **unbounded cliquewidth**

$\Leftrightarrow$ $\mathscr{C}$ transduces a class that contains a subdivision of every **wall**.

Conjecture: $\mathscr{C}$ has **unbounded linear cliquewidth**

$\Leftrightarrow$ $\mathscr{C}$ transduces a class that contains a subdivision of every **binary tree**.

Open problems

str bnd expansion $\stackrel{?}{=}$ **stable bnd flip-width**

str nowhere dense $\stackrel{?}{=}$ **mon stable**

Def: For a property Π, let $\widehat{\Pi}$ be the largest transduction ideal such that

$$\Pi = \widehat{\Pi} \cap \text{weakly sparse.}$$

Does this exactly map the **sparse** column to the **dependent** column?

Conjecture: $\mathscr{C}$ has **unbounded cliquewidth**

$\Leftrightarrow$ $\mathscr{C}$ transduces a class that contains a subdivision of every **wall**.

Conjecture: $\mathscr{C}$ has **unbounded linear cliquewidth**

$\Leftrightarrow$ $\mathscr{C}$ transduces a class that contains a subdivision of every **binary tree**.

Theorem (OdMPS'23)

$\mathscr{C}$ has **unbounded shrubdepth** $\Leftrightarrow$ $\mathscr{C}$ transduces the class of all **paths**.