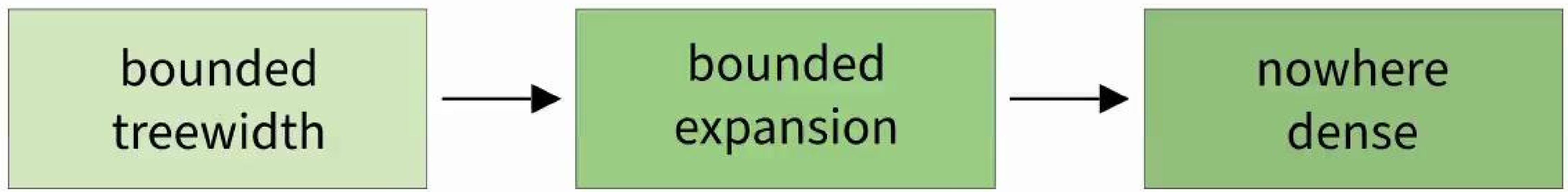
Merge-Width and First-Order Model Checking

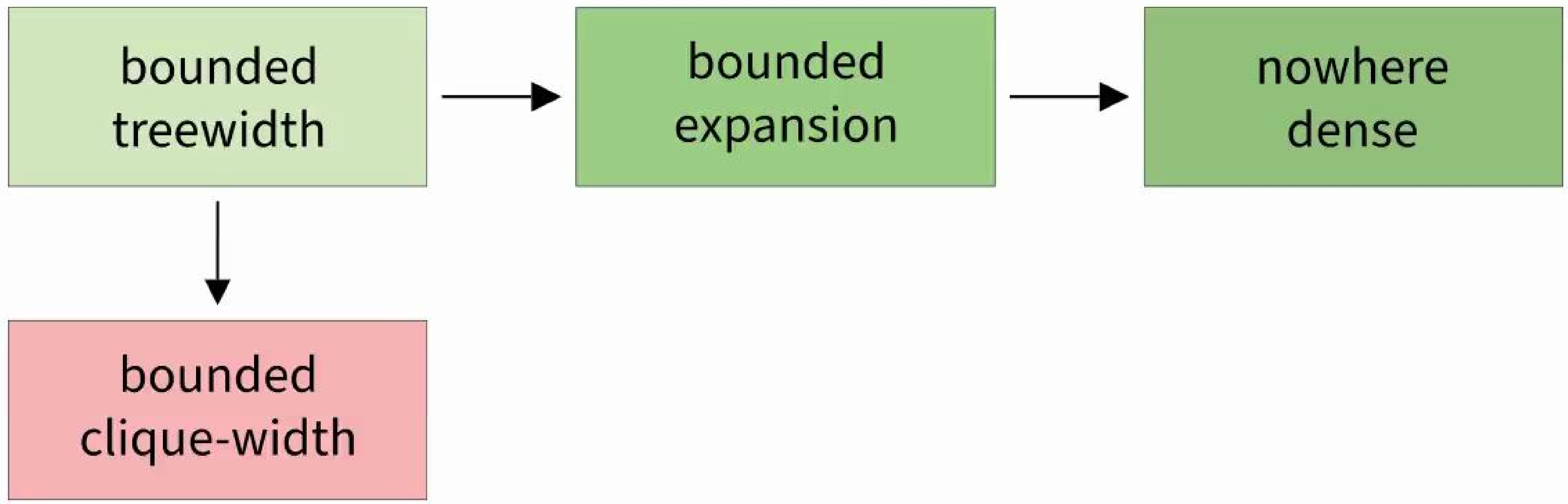
Szymon Toruńczyk, joint work with Jan Dreier

LOGALG’25

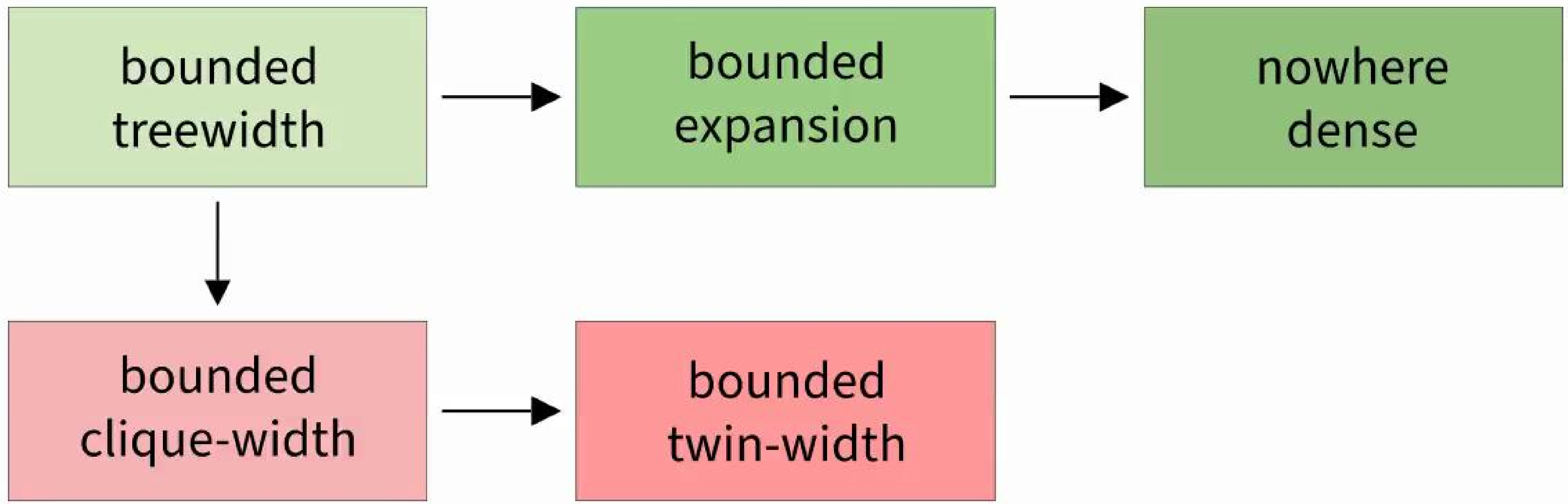
Overview



Overview

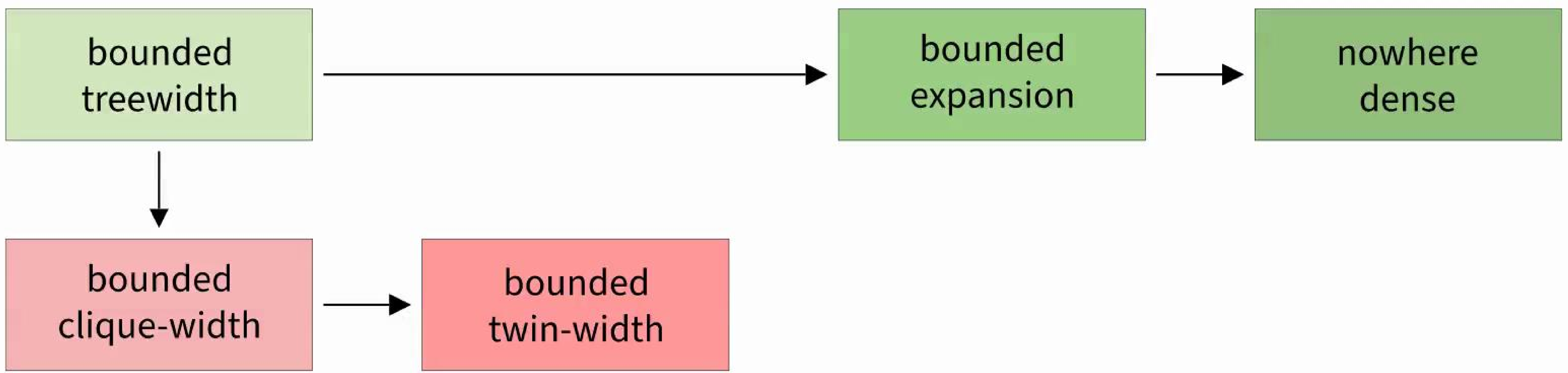

bounded treewidth
bounded expansion
nowhere dense
bounded clique-width

Overview

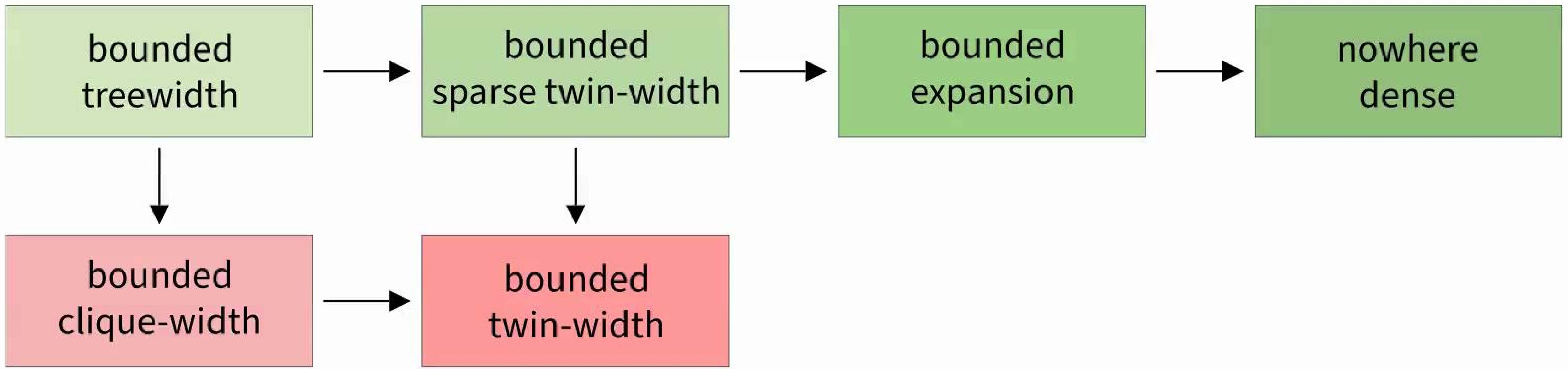


bounded treewidth
bounded expansion
nowhere dense
bounded clique-width
bounded twin-width

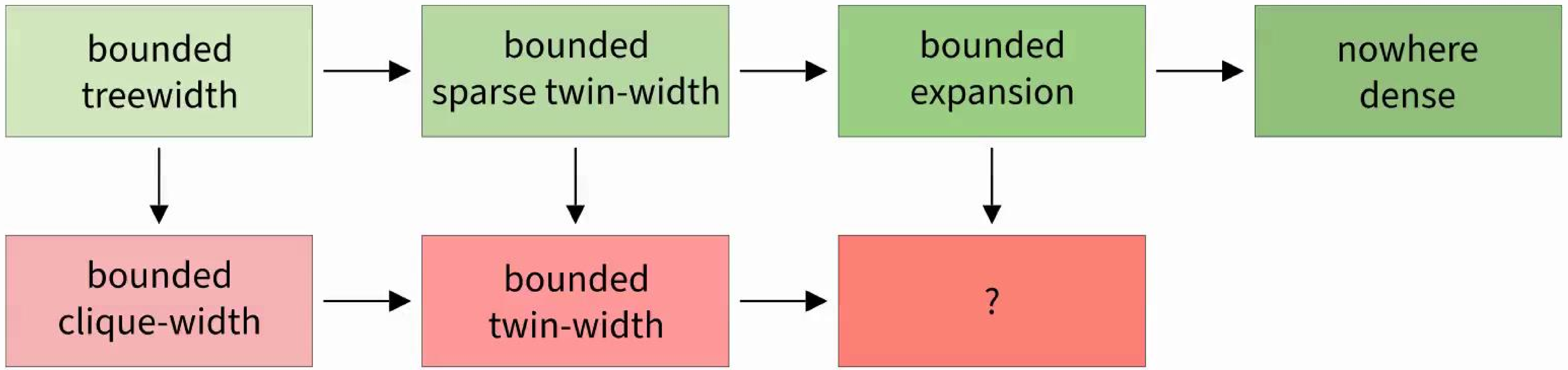
Overview


bounded treewidth
bounded expansion
nowhere dense
bounded clique-width
bounded twin-width

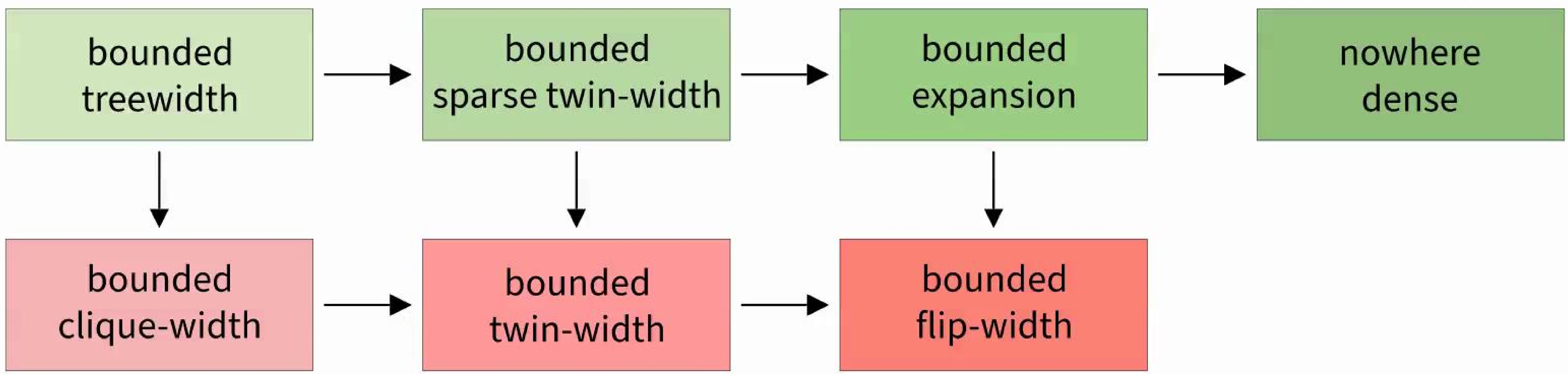
Overview


bounded treewidth
bounded sparse twin-width
bounded expansion
nowhere dense
bounded clique-width
bounded twin-width

Overview

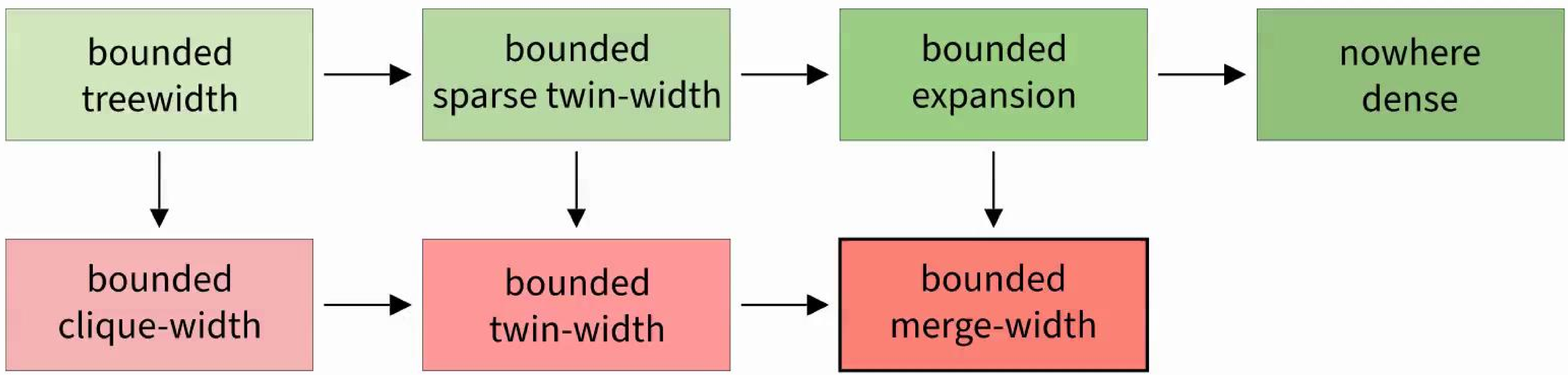

bounded treewidth
bounded sparse twin-width
bounded expansion
nowhere dense
bounded clique-width
bounded twin-width
?

Overview

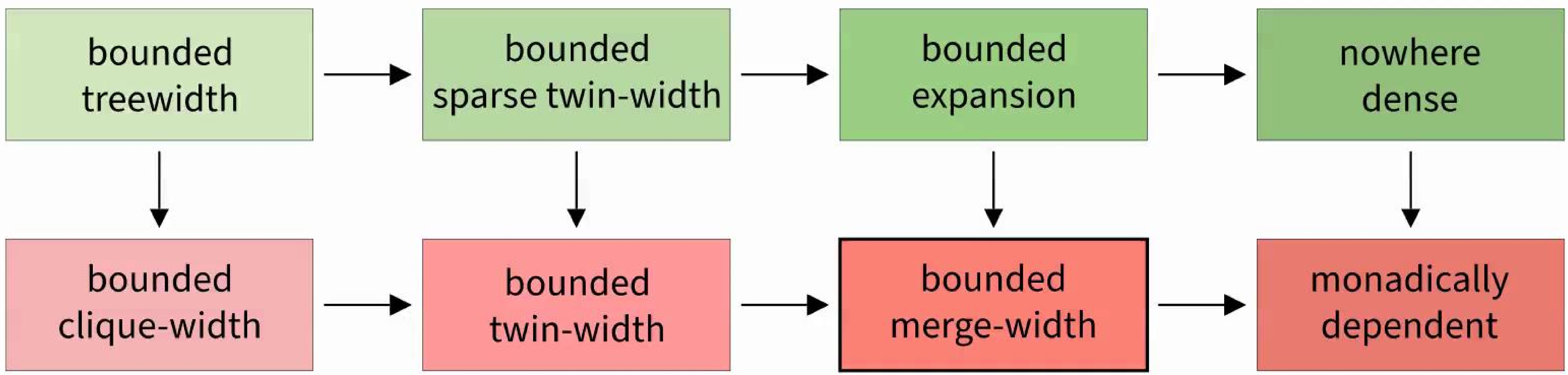


bounded treewidth
bounded sparse twin-width
bounded expansion
nowhere dense
bounded clique-width
bounded twin-width
bounded flip-width

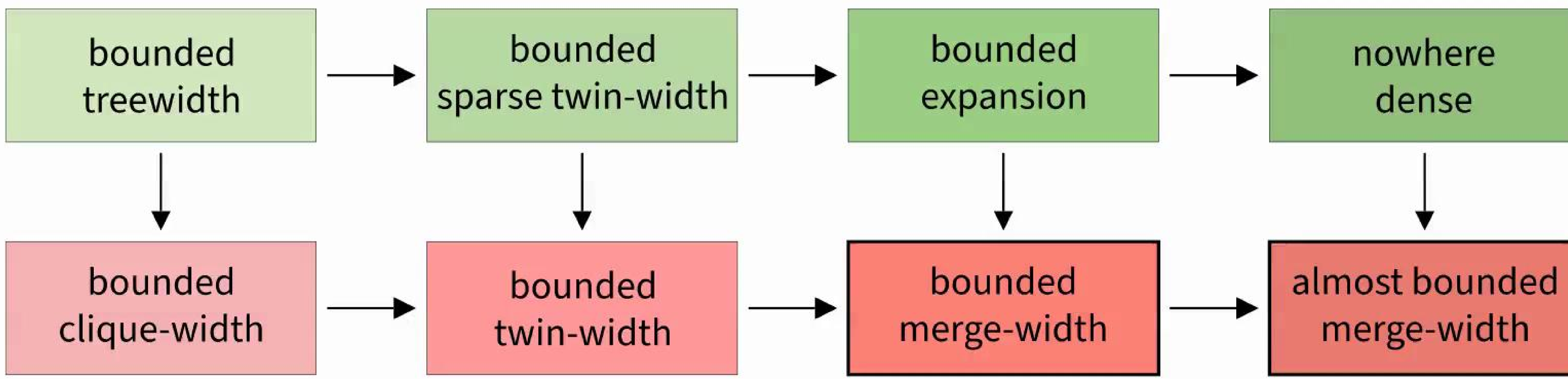
Overview


bounded treewidth
bounded sparse twin-width
bounded expansion
nowhere dense
bounded clique-width
bounded twin-width
bounded merge-width

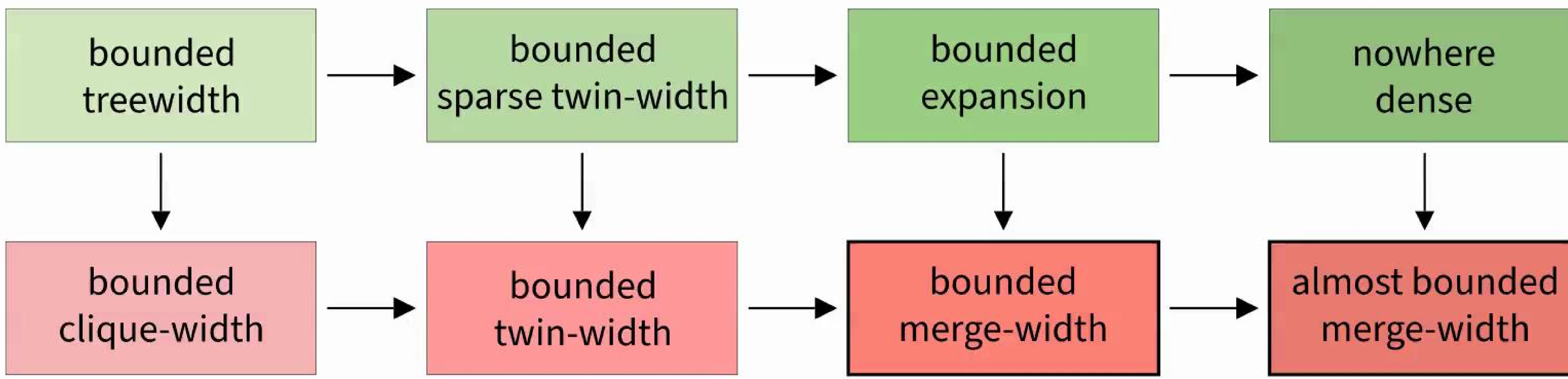
Overview


bounded treewidth
bounded sparse twin-width
bounded expansion
nowhere dense
bounded clique-width
bounded twin-width
bounded merge-width
monadically dependent

Overview

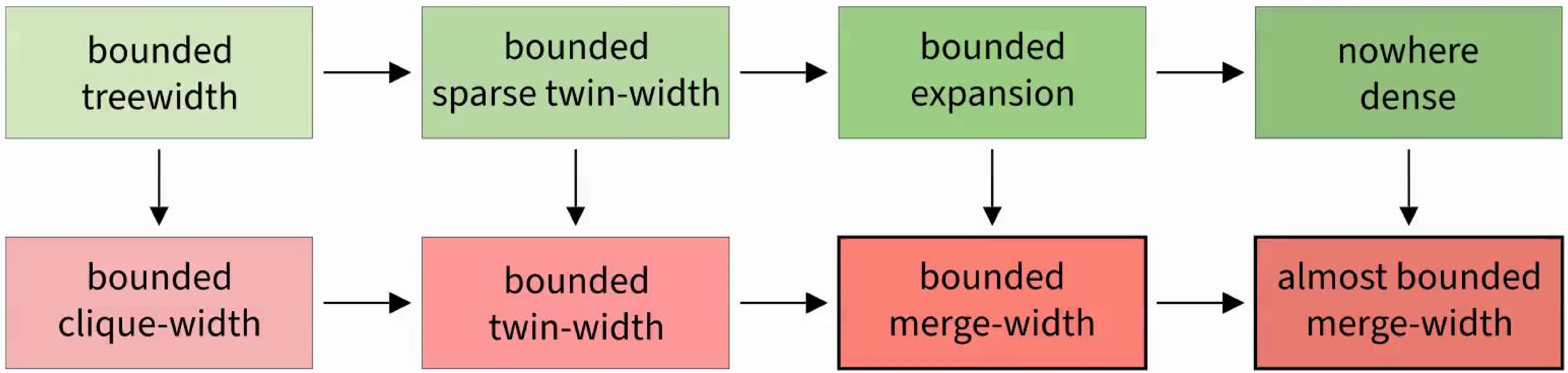

bounded treewidth
bounded sparse twin-width
bounded expansion
nowhere dense
bounded clique-width
bounded twin-width
bounded merge-width
almost bounded merge-width

Overview



Courcelle's theorem. Graph problems definable in monadic-second order logic can be verified in linear time on graphs of bounded tree-width.

Overview



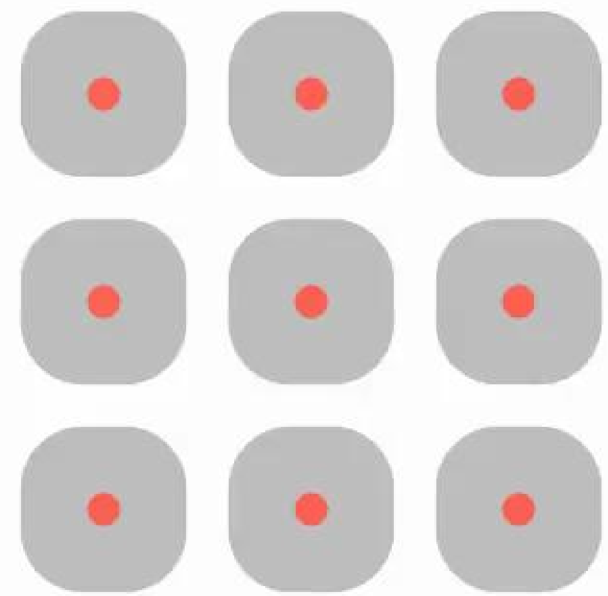
Courcelle's theorem. Graph problems definable in monadic-second order logic can be verified in linear time on graphs of bounded tree-width.

Main result. Graph problems definable in first-order logic can be verified in cubic time on graphs of bounded merge-width, given a decomposition.

V – fixed set of vertices.

V – fixed set of vertices.

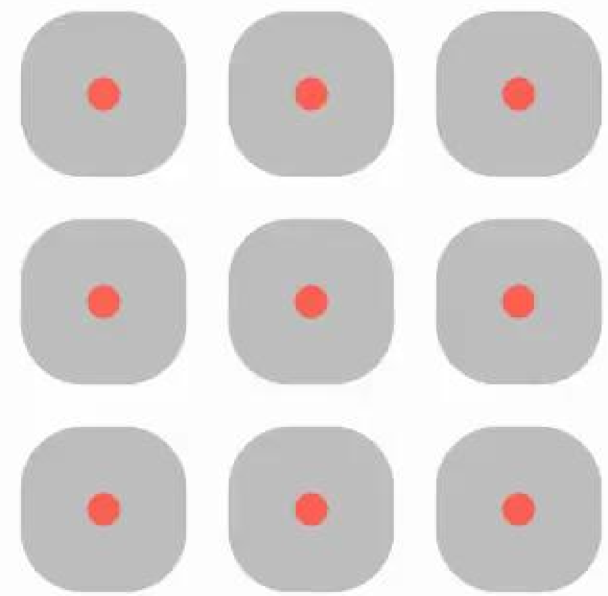
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.



V – fixed set of vertices.

$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges* , initially $E_1 = N_1 = \emptyset$

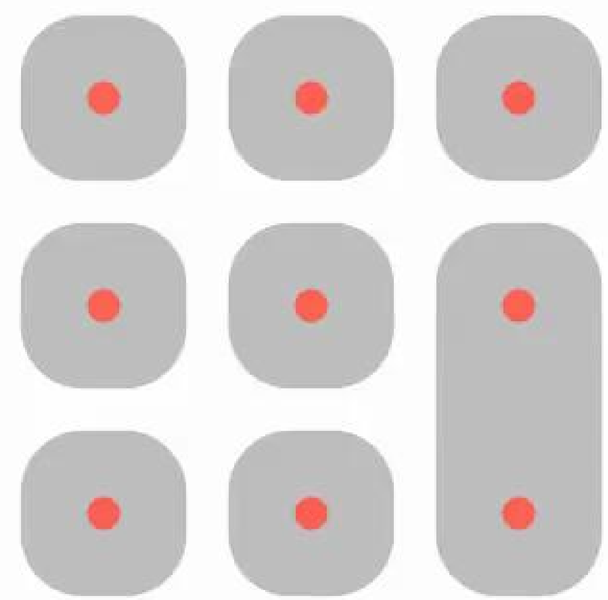


V – fixed set of vertices.

$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges*, initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} \coloneqq \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$



merge

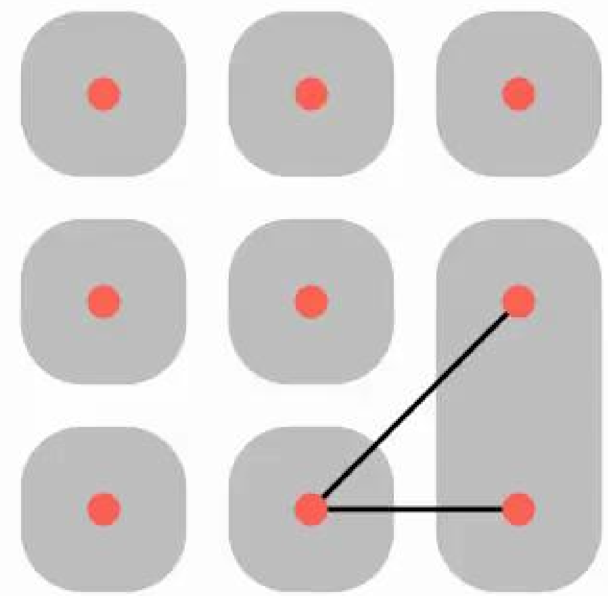
V – fixed set of vertices.

$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges*, initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} \coloneqq \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} \coloneqq E_t \cup (AB - N_t)$



resolve+

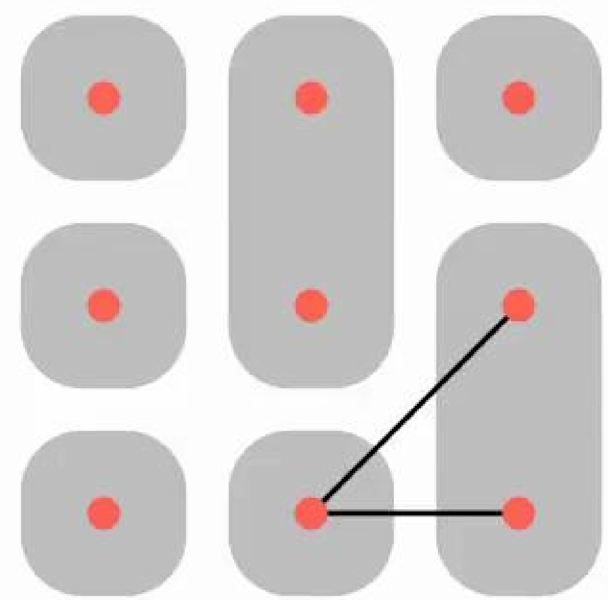
V – fixed set of vertices.

$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges* , initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} \coloneqq \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} \coloneqq E_t \cup (AB - N_t)$



merge

V – fixed set of vertices.

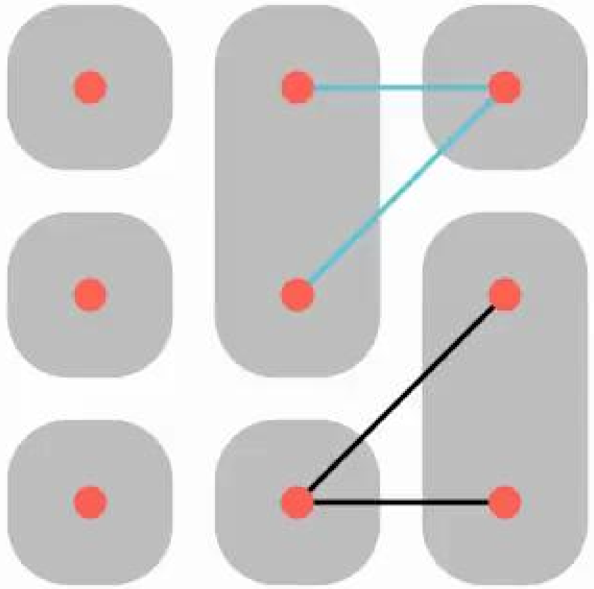
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges* , initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} \coloneqq \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} \coloneqq E_t \cup (AB - N_t)$

- Resolve A, B negatively:
  $N_{t+1} \coloneqq N_t \cup (AB - E_t)$



resolve-

V – fixed set of vertices.

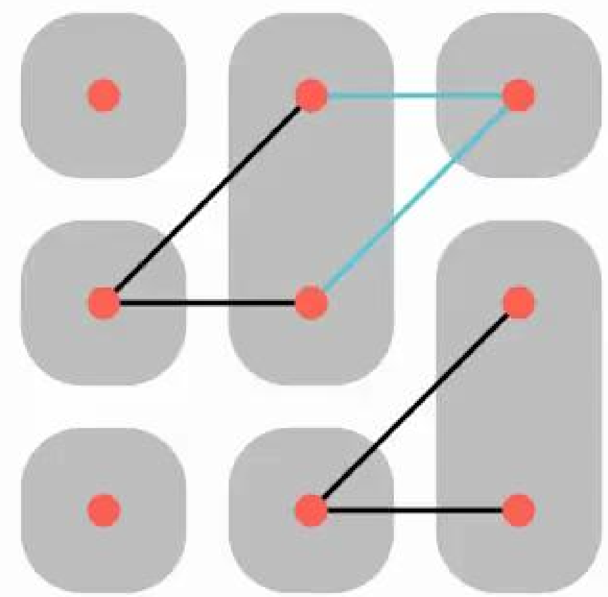
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges*, initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} \coloneqq \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} \coloneqq E_t \cup (AB - N_t)$

- Resolve A, B negatively:
  $N_{t+1} \coloneqq N_t \cup (AB - E_t)$



resolve+

V – fixed set of vertices.

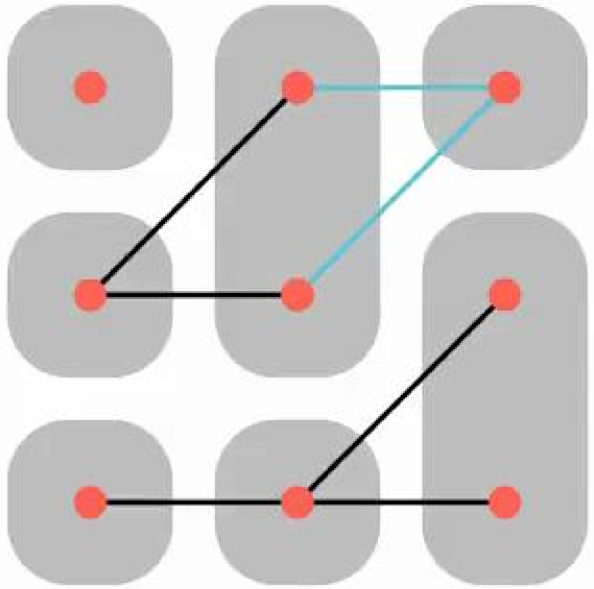
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges* , initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} := \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} := E_t \cup (AB - N_t)$

- Resolve A, B negatively:
  $N_{t+1} := N_t \cup (AB - E_t)$



resolve+

V – fixed set of vertices.

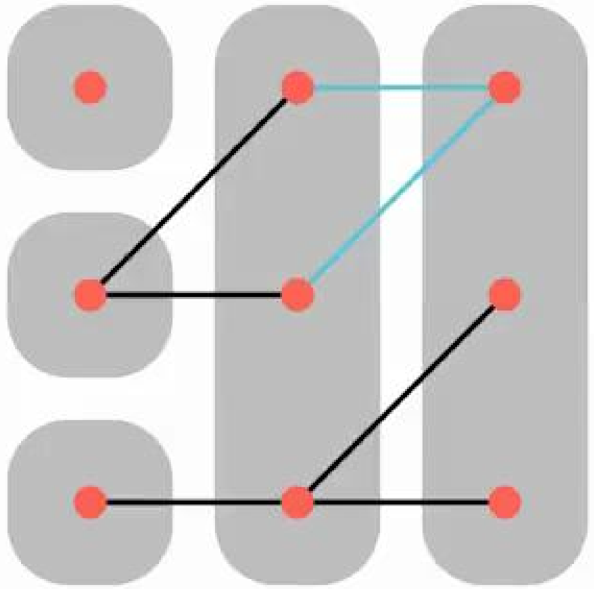
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges*, initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} := \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} := E_t \cup (AB - N_t)$

- Resolve A, B negatively:
  $N_{t+1} := N_t \cup (AB - E_t)$



merge

V – fixed set of vertices.

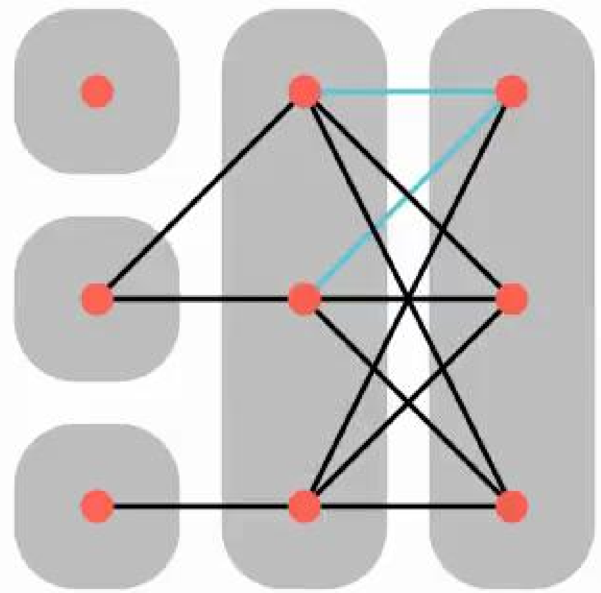
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges*, initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} := \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} := E_t \cup (AB - N_t)$

- Resolve A, B negatively:
  $N_{t+1} := N_t \cup (AB - E_t)$



resolve+

V – fixed set of vertices.

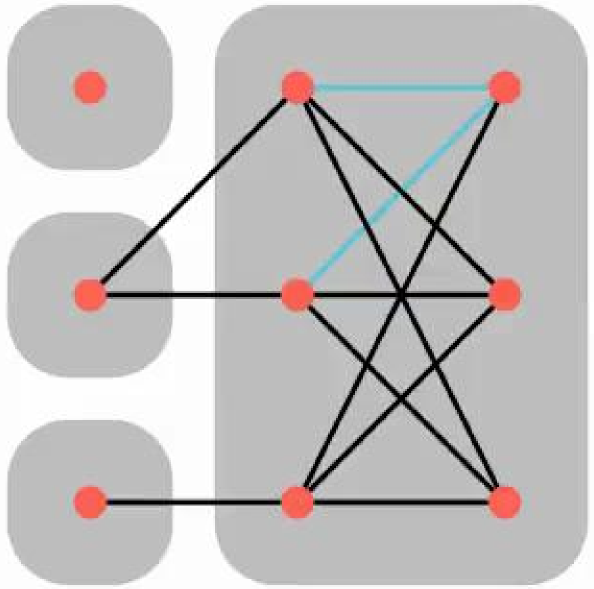
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges*, initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} := \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} := E_t \cup (AB - N_t)$

- Resolve A, B negatively:
  $N_{t+1} := N_t \cup (AB - E_t)$



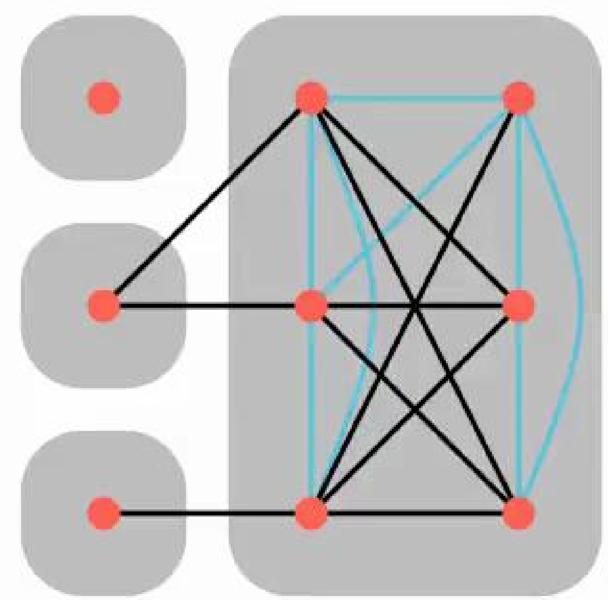
merge

V – fixed set of vertices.

$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges* , initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} := \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$
- Resolve A, B positively:
  $E_{t+1} := E_t \cup (AB - N_t)$
- Resolve A, B negatively:
  $N_{t+1} := N_t \cup (AB - E_t)$



resolve-

V – fixed set of vertices.

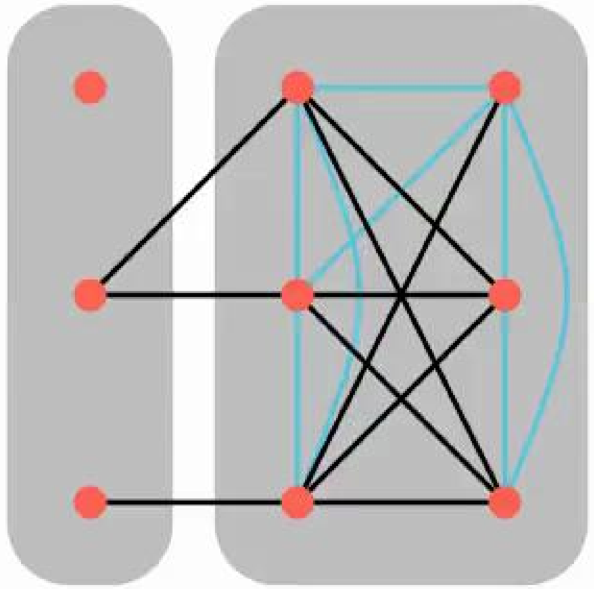
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges* , initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} := \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} := E_t \cup (AB - N_t)$

- Resolve A, B negatively:
  $N_{t+1} := N_t \cup (AB - E_t)$



merge

V – fixed set of vertices.

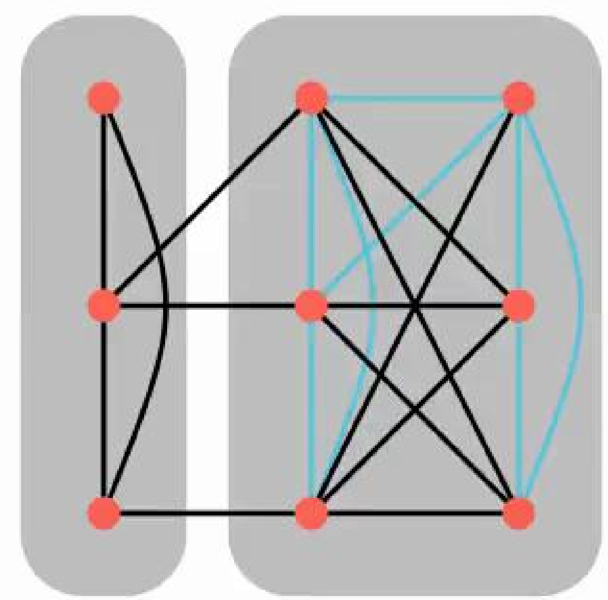
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges* , initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} := \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} := E_t \cup (AB - N_t)$

- Resolve A, B negatively:
  $N_{t+1} := N_t \cup (AB - E_t)$



resolve+

V – fixed set of vertices.

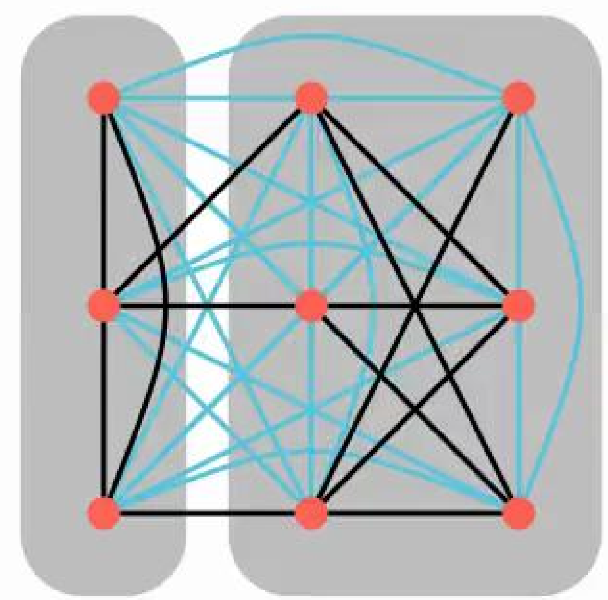
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges*, initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} := \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} := E_t \cup (AB - N_t)$

- Resolve A, B negatively:
  $N_{t+1} := N_t \cup (AB - E_t)$



resolve-

V – fixed set of vertices.

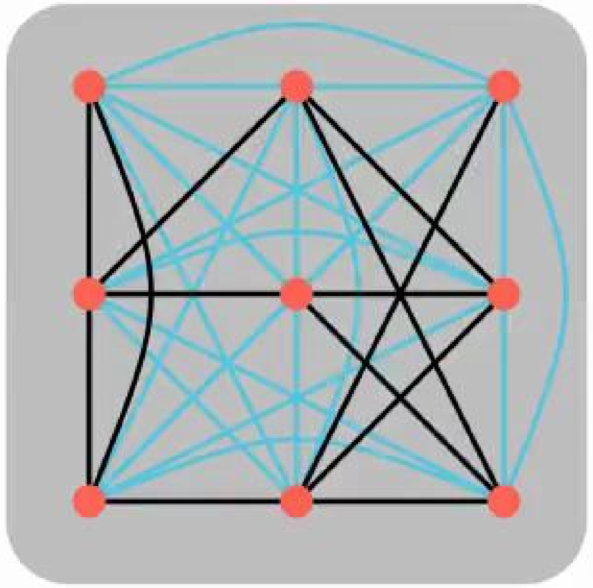
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges* , initially $E_1 = N_1 = \emptyset$

- Merge A, B:
  $\mathcal{P}_{t+1} \coloneqq \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} \coloneqq E_t \cup (AB - N_t)$

- Resolve A, B negatively:
  $N_{t+1} \coloneqq N_t \cup (AB - E_t)$



merge

V – fixed set of vertices.

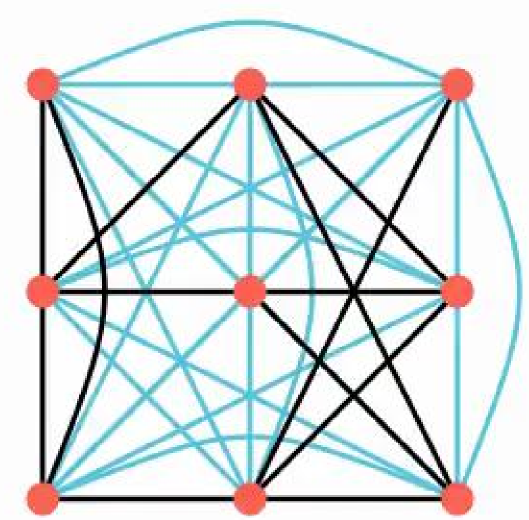
$\mathcal{P}_t$ – partition of V, initially $\mathcal{P}_1 = \{\{v\} : v \in V\}$.

$E_t, N_t \subseteq \binom{V}{2}$ – *edges* and *non-edges* , initially $E_1 = N_1 = \emptyset$

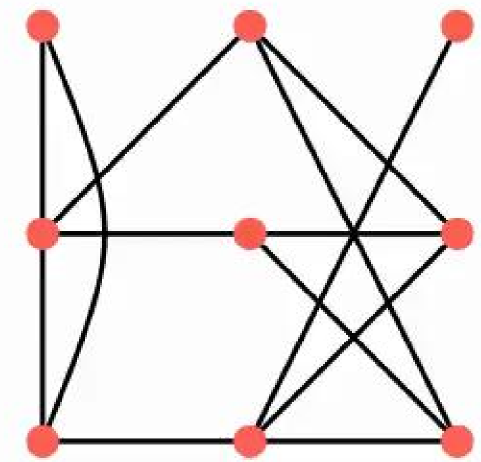
- Merge A, B:
  $\mathcal{P}_{t+1} := \mathcal{P}_t - \{A, B\} \cup \{A \cup B\}$

- Resolve A, B positively:
  $E_{t+1} := E_t \cup (AB - N_t)$

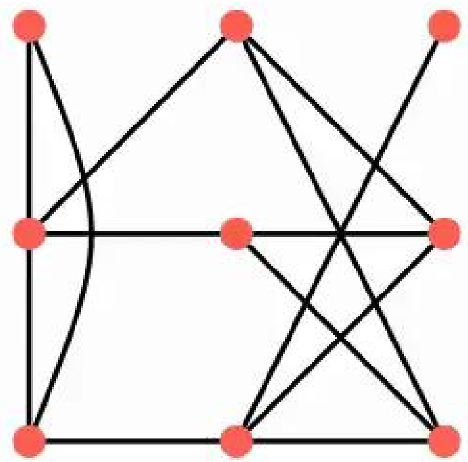
- Resolve A, B negatively:
  $N_{t+1} := N_t \cup (AB - E_t)$



done!

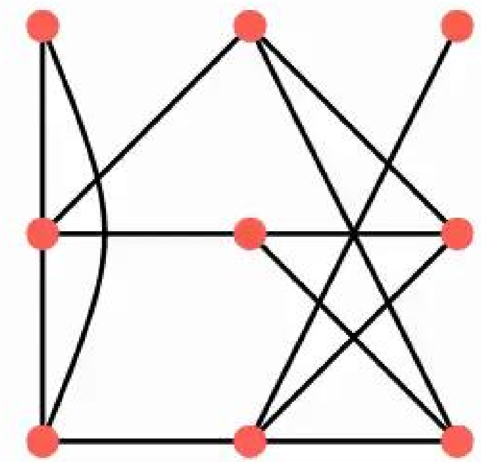


Construction sequence: such a sequence $(\mathcal{P}_1, E_1, N_1), \ldots, (\mathcal{P}_n, E_n, N_n)$.



Construction sequence: such a sequence $(\mathcal{P}_1, E_1, N_1), \ldots, (\mathcal{P}_n, E_n, N_n)$.

Fix $r \in \mathbb{N}$.



Construction sequence: such a sequence $(\mathcal{P}_1, E_1, N_1), \ldots, (\mathcal{P}_n, E_n, N_n)$.

Fix $r \in \mathbb{N}$.

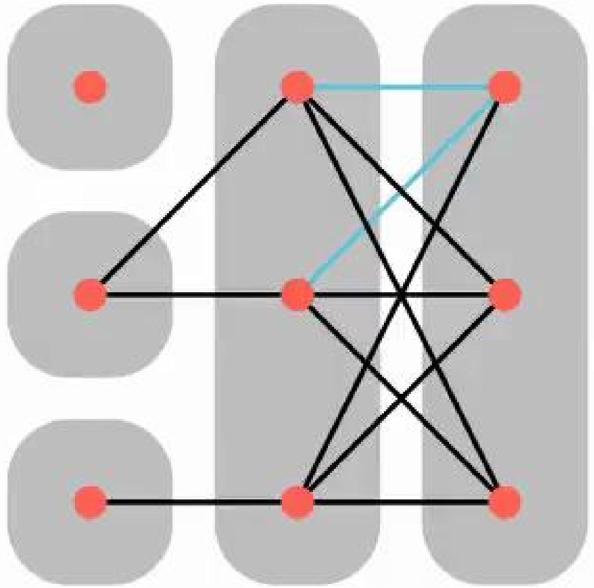
radius-r width of a construction sequence :=

Construction sequence: such a sequence $(\mathcal{P}_1, E_1, N_1), \ldots, (\mathcal{P}_n, E_n, N_n)$.

Fix $r \in \mathbb{N}$.

radius-r width of a construction sequence :=

$$\max_{t=1..n}$$

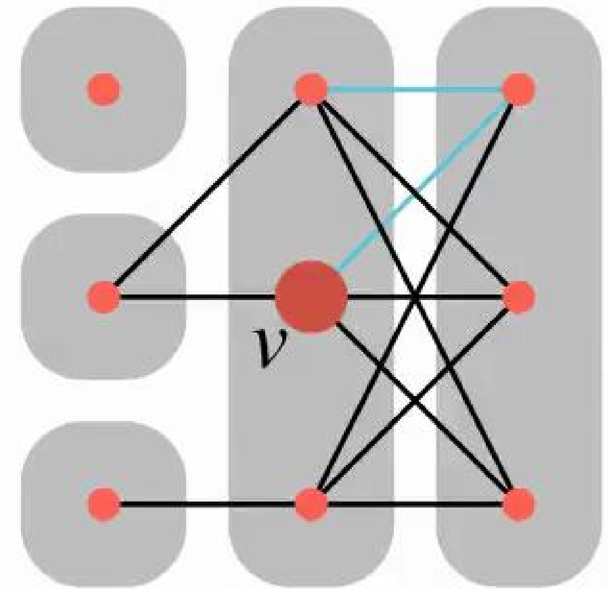


Construction sequence: such a sequence $(\mathcal{P}_1, E_1, N_1), \ldots, (\mathcal{P}_n, E_n, N_n)$.

Fix $r \in \mathbb{N}$.

radius-r width of a construction sequence :=

$$\max_{t=1..n}$$

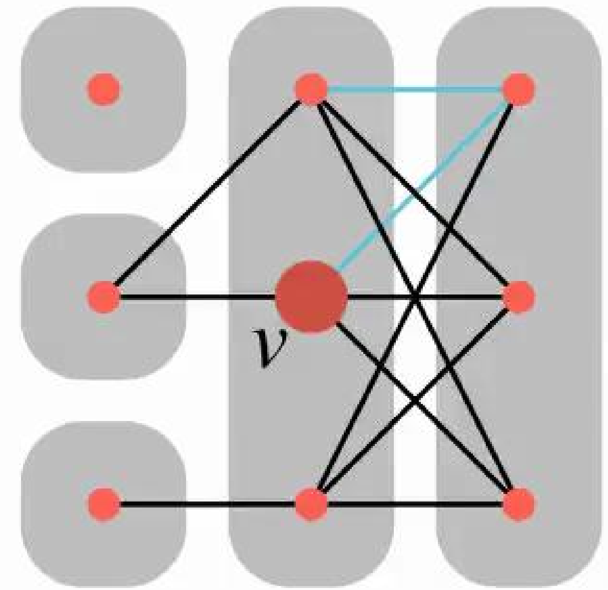


Construction sequence: such a sequence $(\mathcal{P}_1, E_1, N_1), \ldots, (\mathcal{P}_n, E_n, N_n)$.

Fix $r \in \mathbb{N}$.

radius-r width of a construction sequence :=

$$\max_{t=1..n} \quad \max_{v \in V}$$

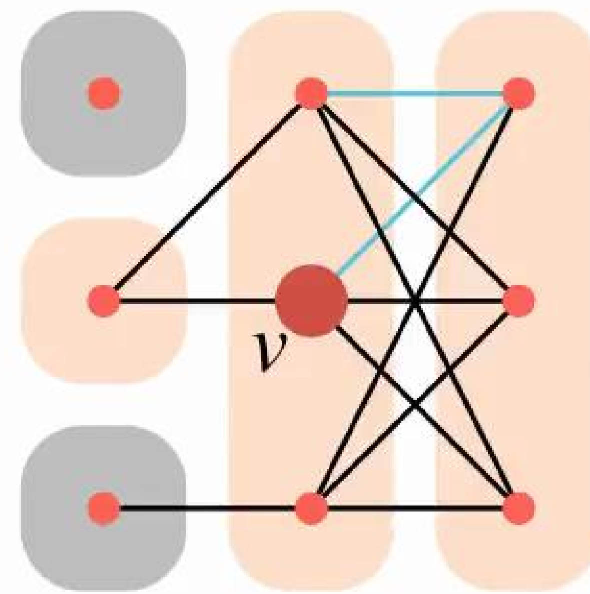


Construction sequence: such a sequence $(\mathcal{P}_1, E_1, N_1), \ldots, (\mathcal{P}_n, E_n, N_n)$.

Fix $r \in \mathbb{N}$.

radius-r width of a construction sequence :=

$$\max_{t=1..n} \quad \max_{v \in V} \quad \#\text{parts of } \mathcal{P}_t \text{ at distance} \leqslant r \text{ from } v \text{ in } (V, E_t \cup N_t)$$



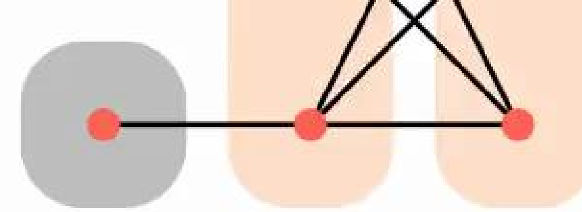
$(r = 2)$

Construction sequence: such a sequence $(\mathcal{P}_1, E_1, N_1), \ldots, (\mathcal{P}_n, E_n, N_n)$.

Fix $r \in \mathbb{N}$.

radius-r width of a construction sequence :=

$$\max_{t=1..n} \quad \max_{v \in V} \quad \#\text{parts of } \mathcal{P}_t \text{ at distance} \leqslant r \text{ from } v \text{ in } (V, E_t \cup N_t)$$



Construction sequence: such a sequence $(\mathcal{P}_1, E_1, N_1), \ldots, (\mathcal{P}_n, E_n, N_n)$.

Fix $r \in \mathbb{N}$.

radius-r width of a construction sequence :=

$$\max_{t=1..n} \quad \max_{v \in V} \quad \#\text{parts of } \mathcal{P}_t \text{ at distance } \leqslant r \text{ from } v \text{ in } (V, E_t \cup N_t)$$

$\underbrace{\text{radius-}r \text{ merge-width of } G}_{\mathrm{mw}_r(G)}$:= min. radius-r width of a constr. sequence of G

Construction sequence: such a sequence $(\mathcal{P}_1, E_1, N_1), \ldots, (\mathcal{P}_n, E_n, N_n)$.

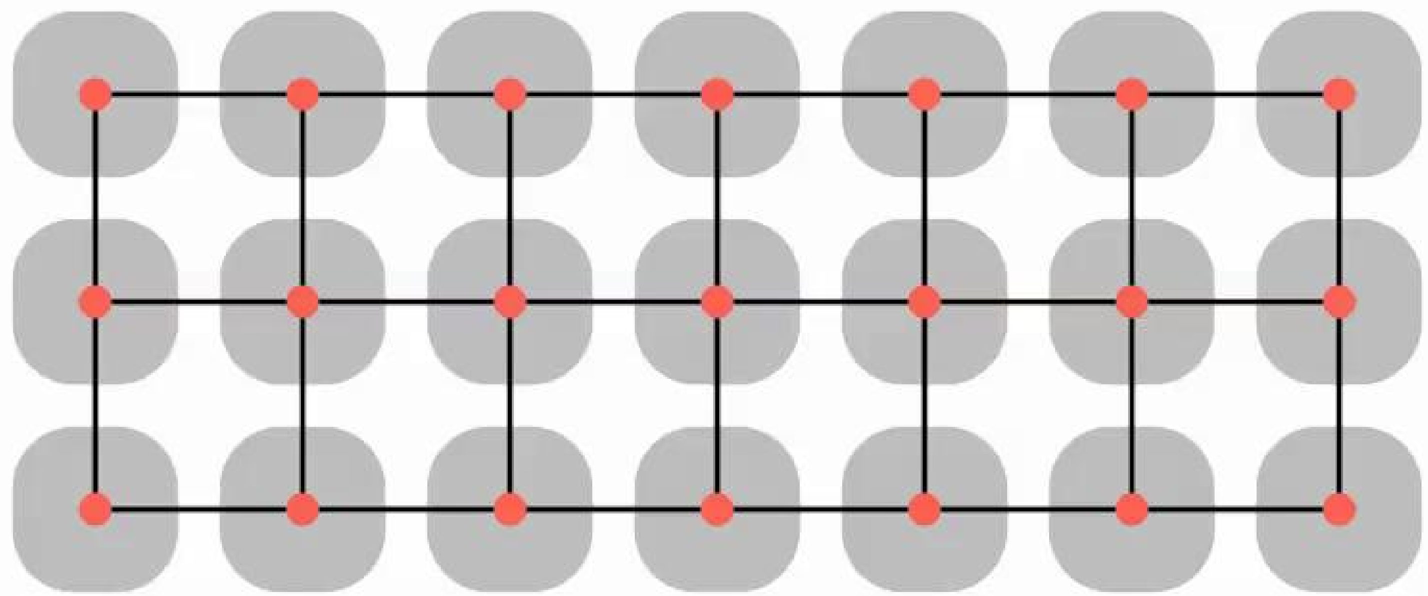
Fix $r \in \mathbb{N}$.

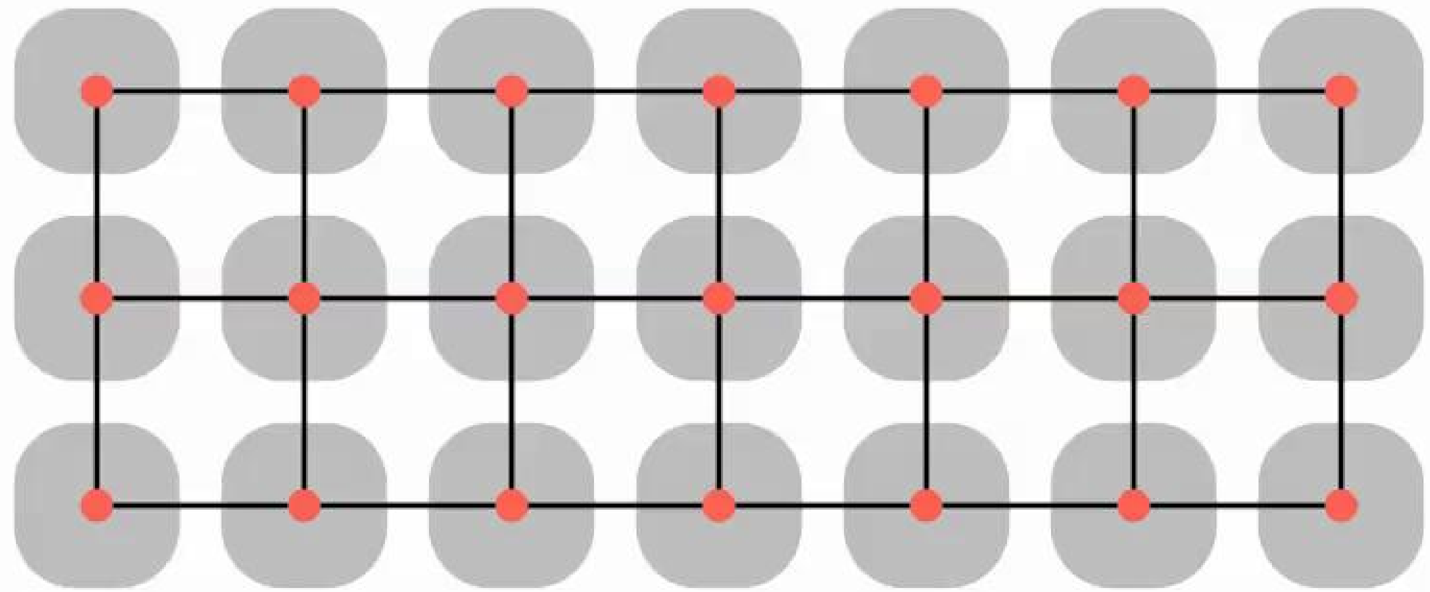
radius-r width of a construction sequence :=

$$\max_{t=1..n} \quad \max_{v \in V} \quad \#\text{parts of } \mathcal{P}_t \text{ at distance } \leqslant r \text{ from } v \text{ in } (V, E_t \cup N_t)$$

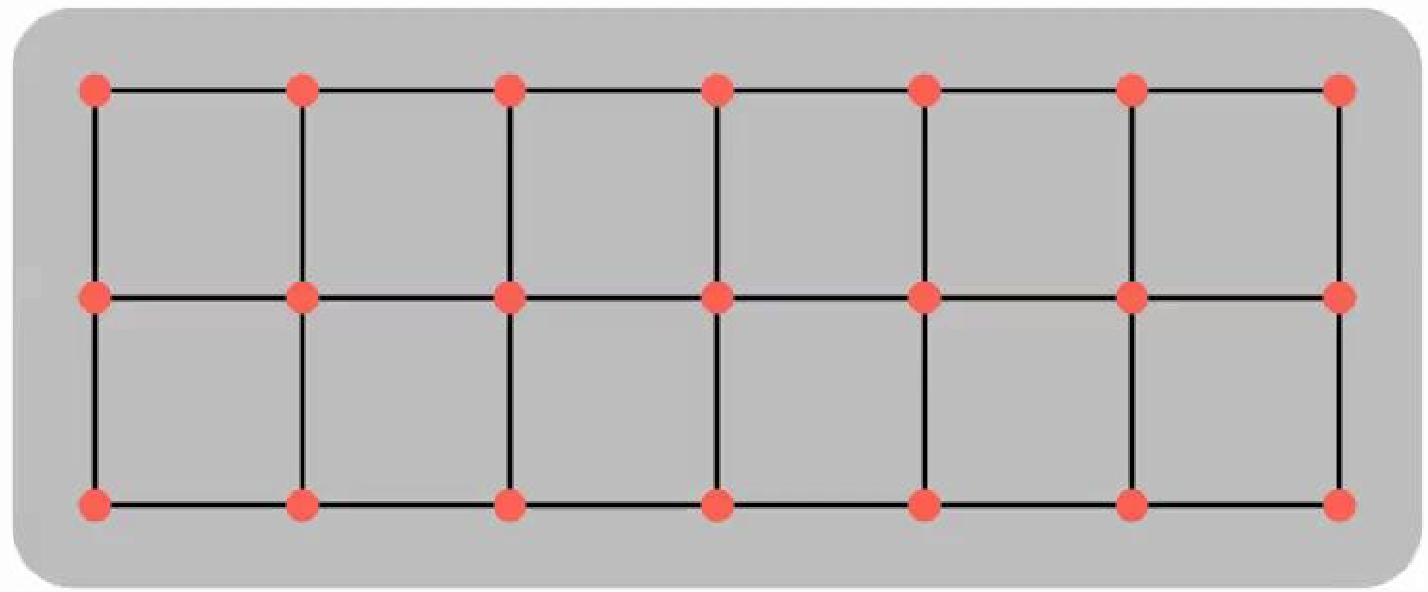
$$\underbrace{\text{radius-}r \text{ merge-width of } G}_{\mathrm{mw}_r(G)} := \text{min. radius-}r\text{ width of a constr. sequence of } G$$

A graph class has *bounded merge-width* if $\mathrm{mw}_r(C) < \infty$ for all $r \in \mathbb{N}$.

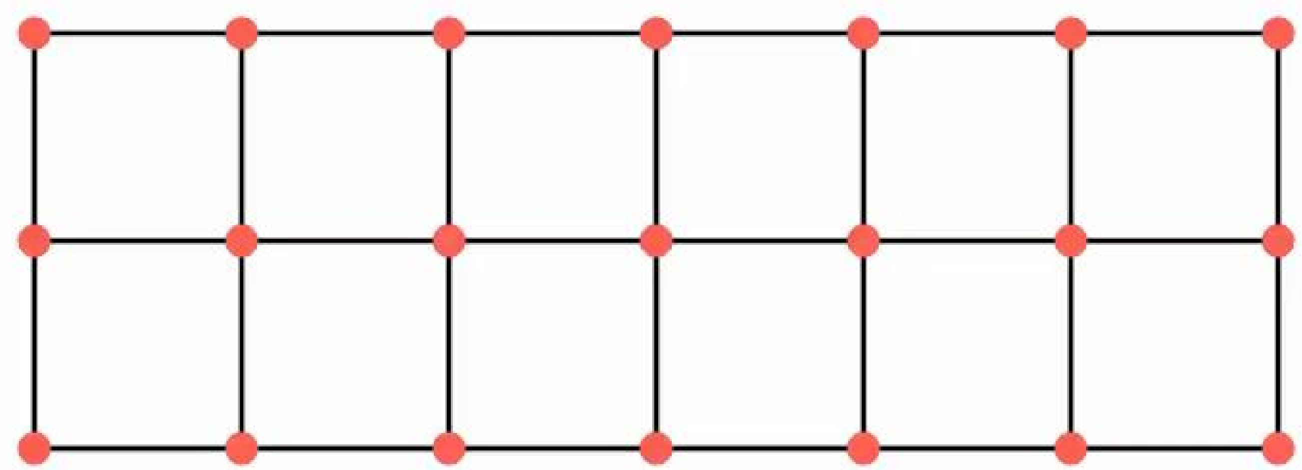




$$\mathrm{mw}_r(G) \leqslant O(\text{maximum-degree}(G))^r$$



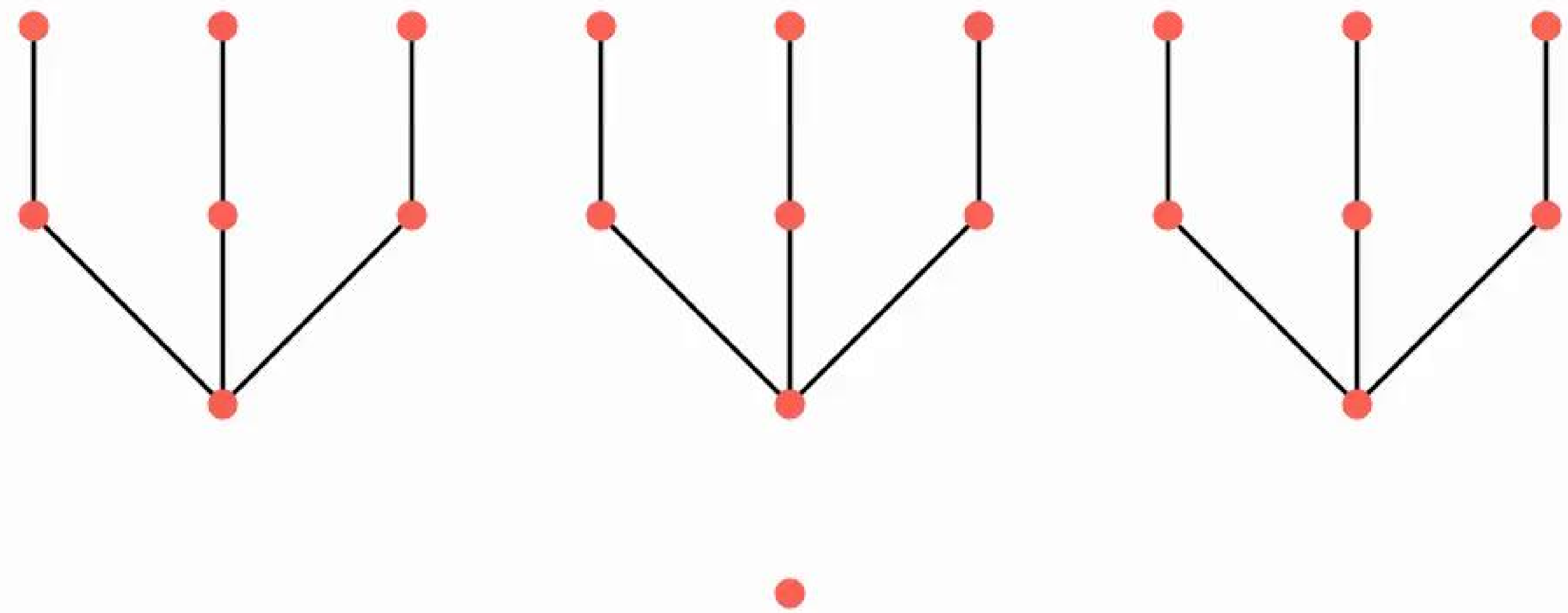
$$\mathrm{mw}_r(G) \leqslant O(\text{maximum-degree}(G))^r$$



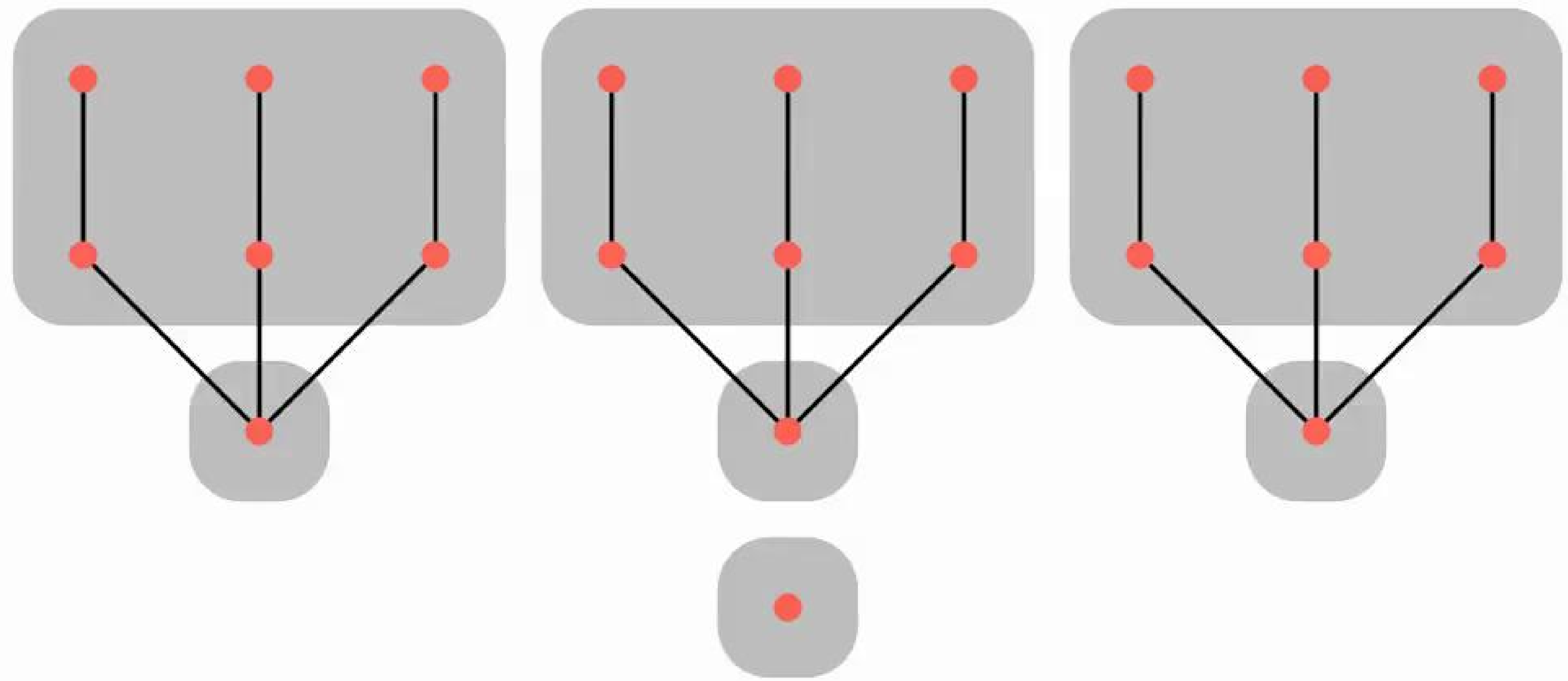
$$\mathrm{mw}_r(G) \leqslant O(\text{maximum-degree}(G))^r$$

Corollary. Every class of bounded maximum degree has bounded merge-width.

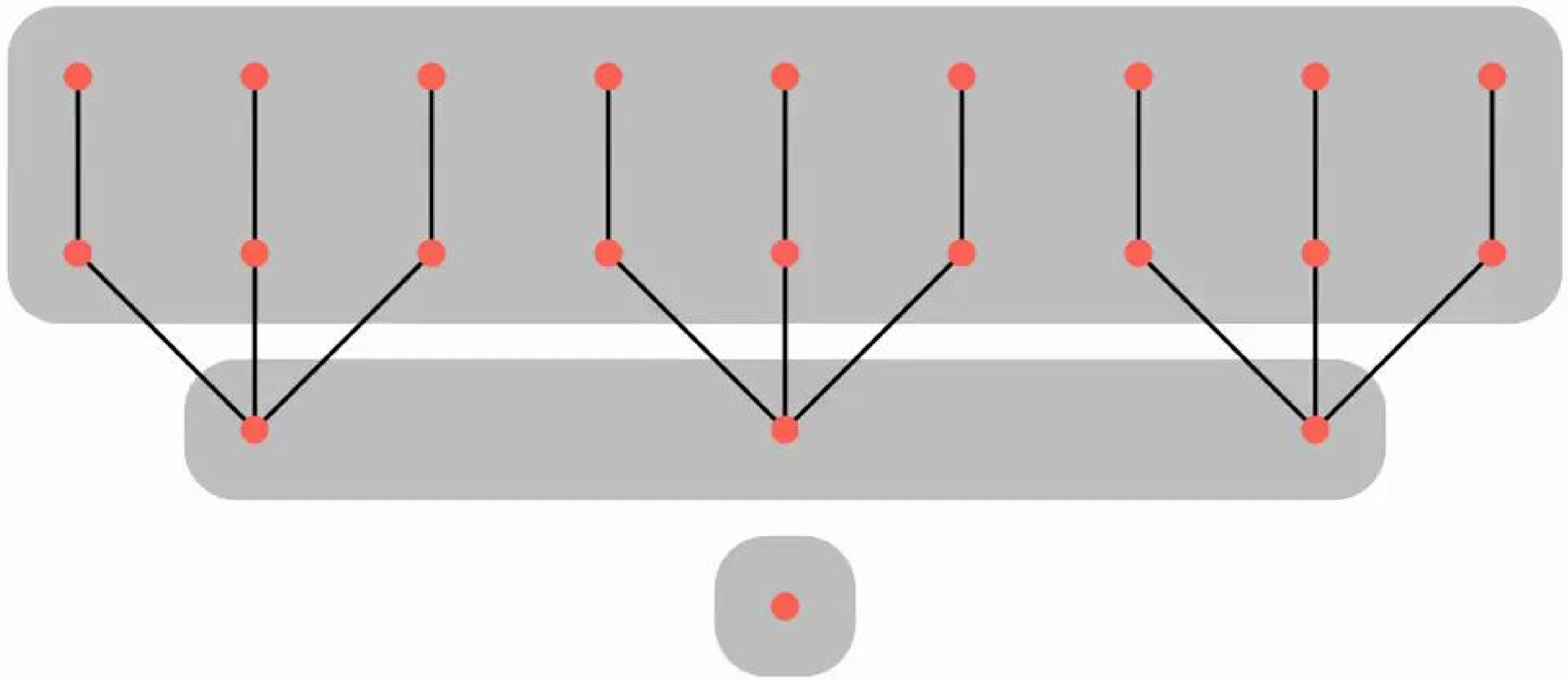
Trees



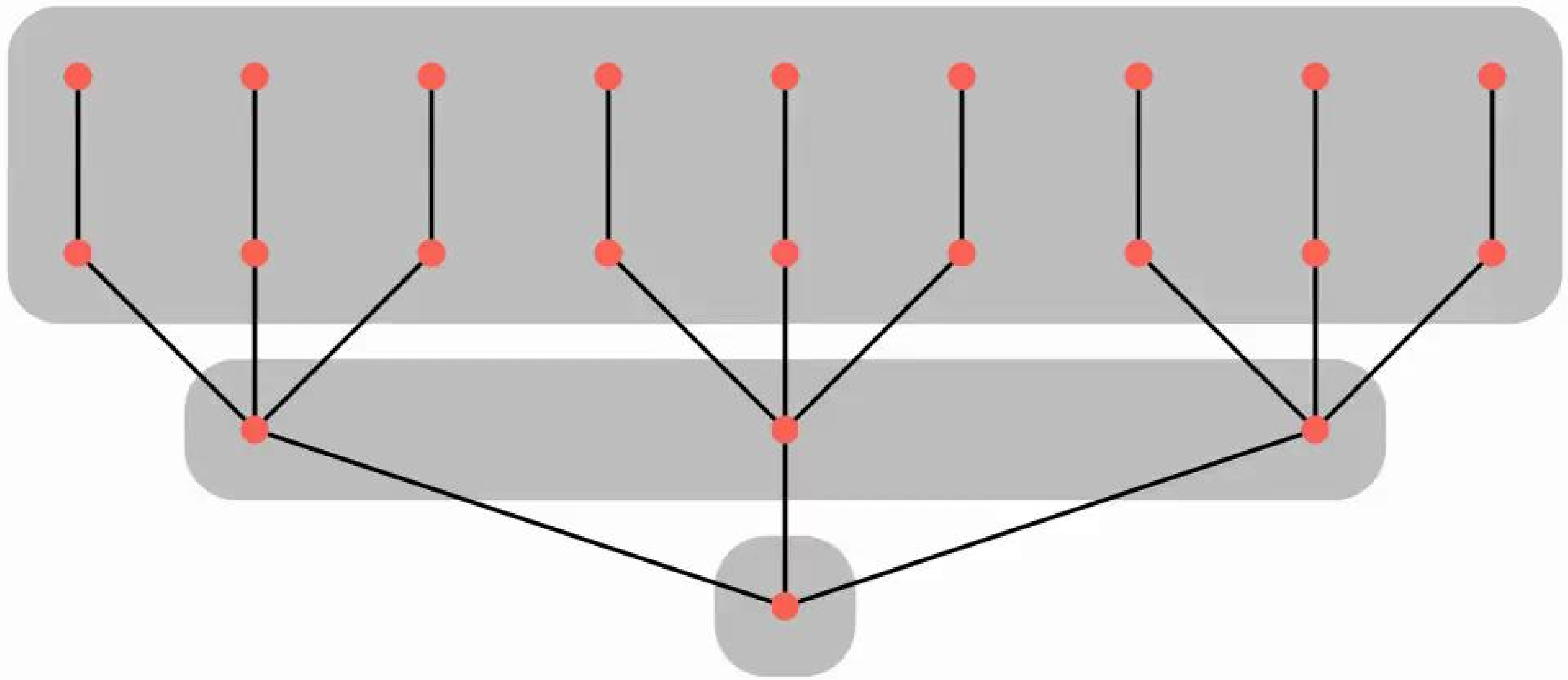
Trees



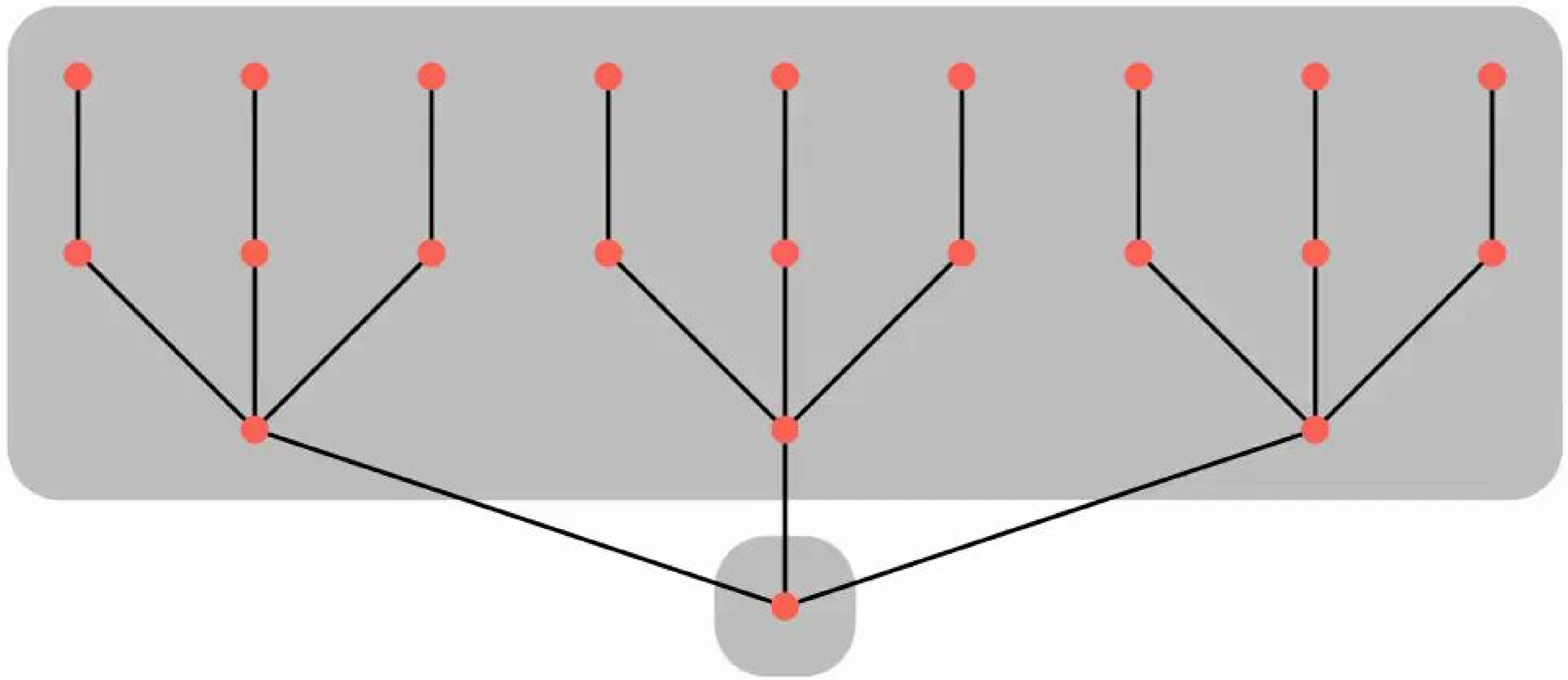
Trees



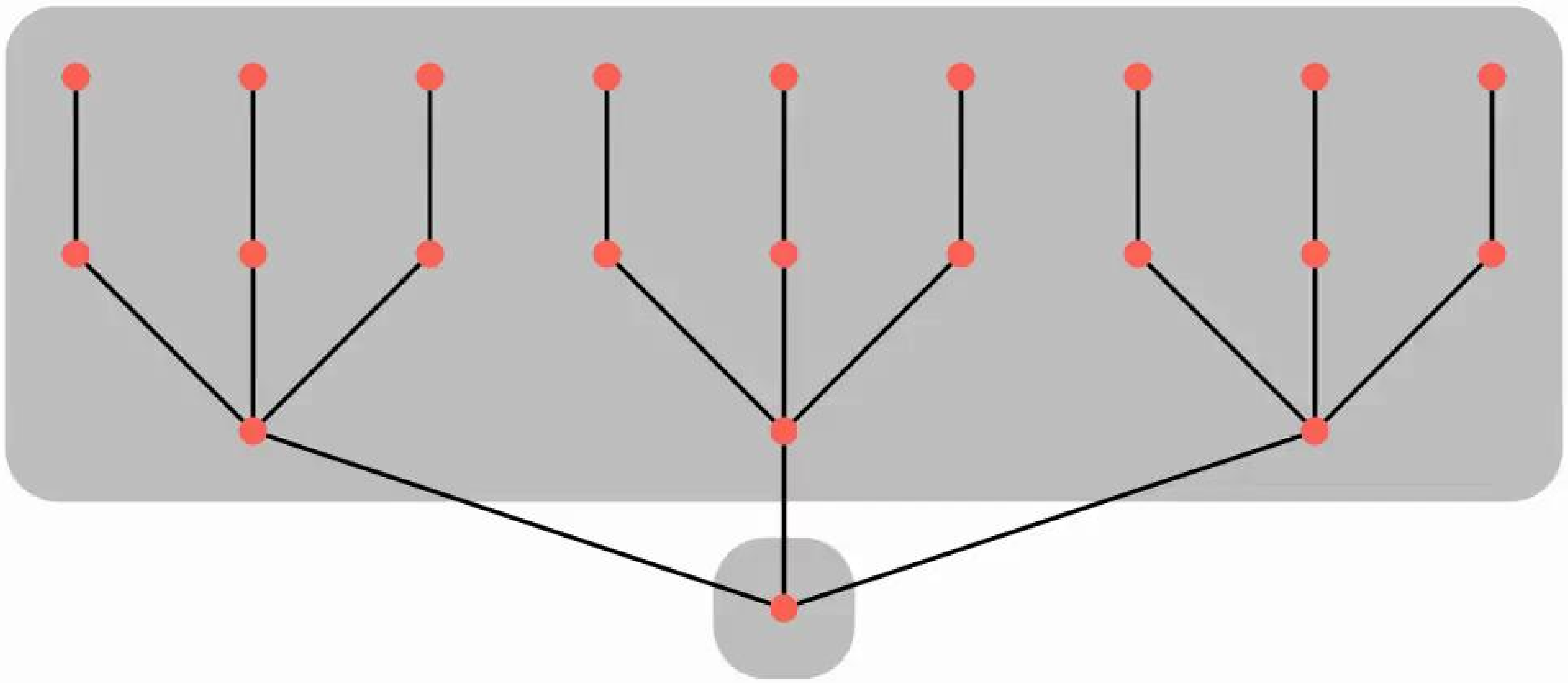
Trees



Trees

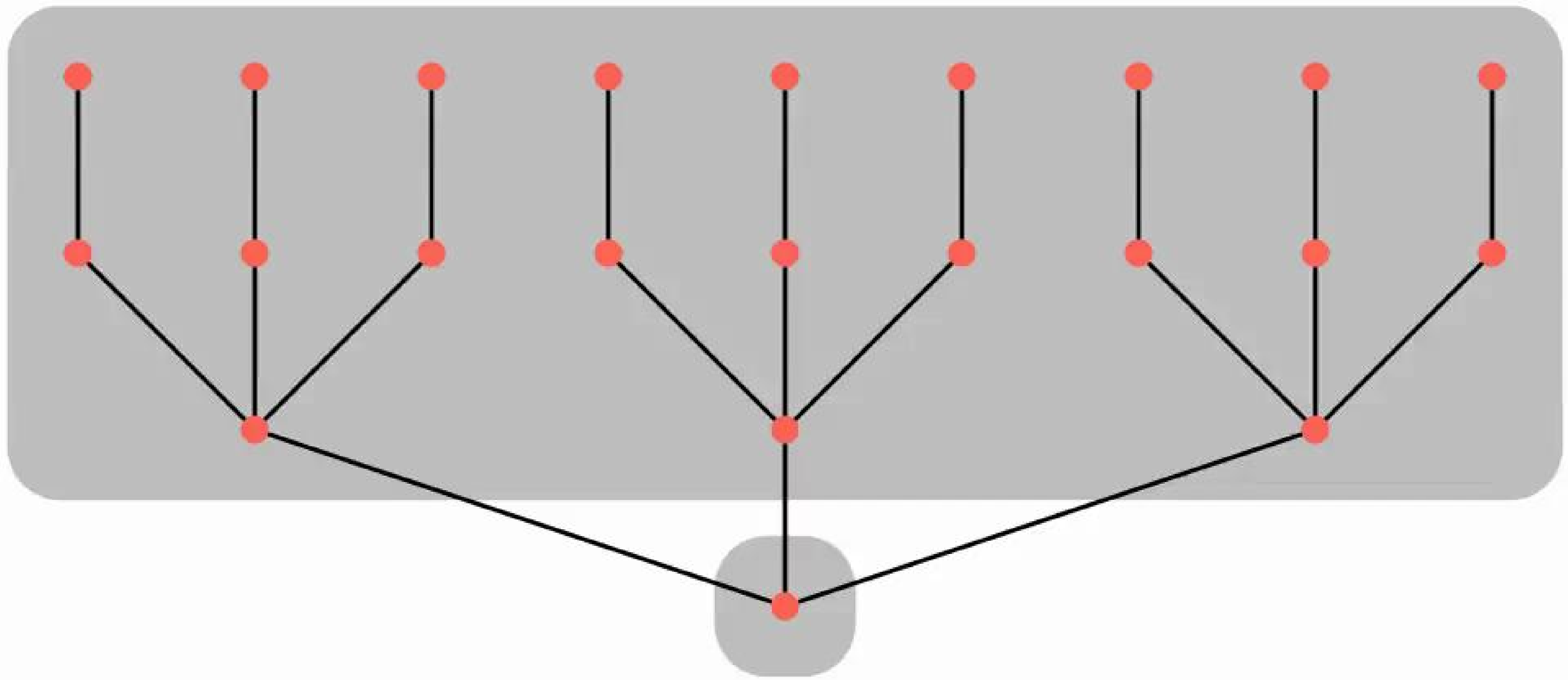


Trees



$$\mathrm{mw}_\infty(\mathit{Trees}) \leqslant 3$$

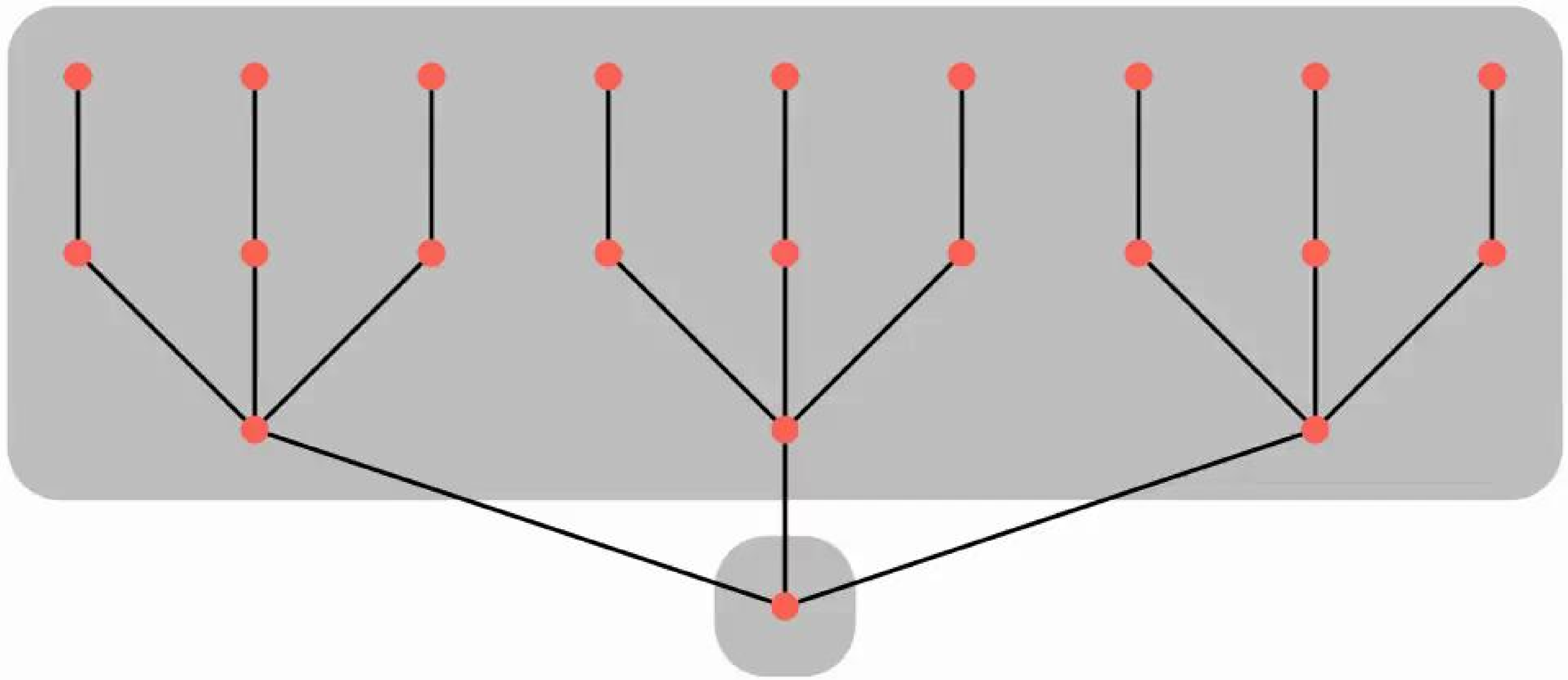
Trees



$$\mathrm{mw}_\infty(\textit{Trees}) \leqslant 3$$

Theorem. $\mathrm{mw}_\infty \approx$ clique-width

Trees



$$\mathrm{mw}_\infty(\mathit{Trees}) \leqslant 3$$

Theorem. $\mathrm{mw}_\infty \approx \text{clique-width}$

Corollary. Every class of bounded clique-width has bounded merge-width.

Twin-width

[Bonnet, Kim, Thomassé, Watrigant, 2021]

$$\text{twin-width}(G) \leqslant k \quad ^*$$

$$\Updownarrow$$

$\exists$ construction sequence such that each *part* neighbors $\leqslant k$ other parts

*differs by ± 1 from original definition

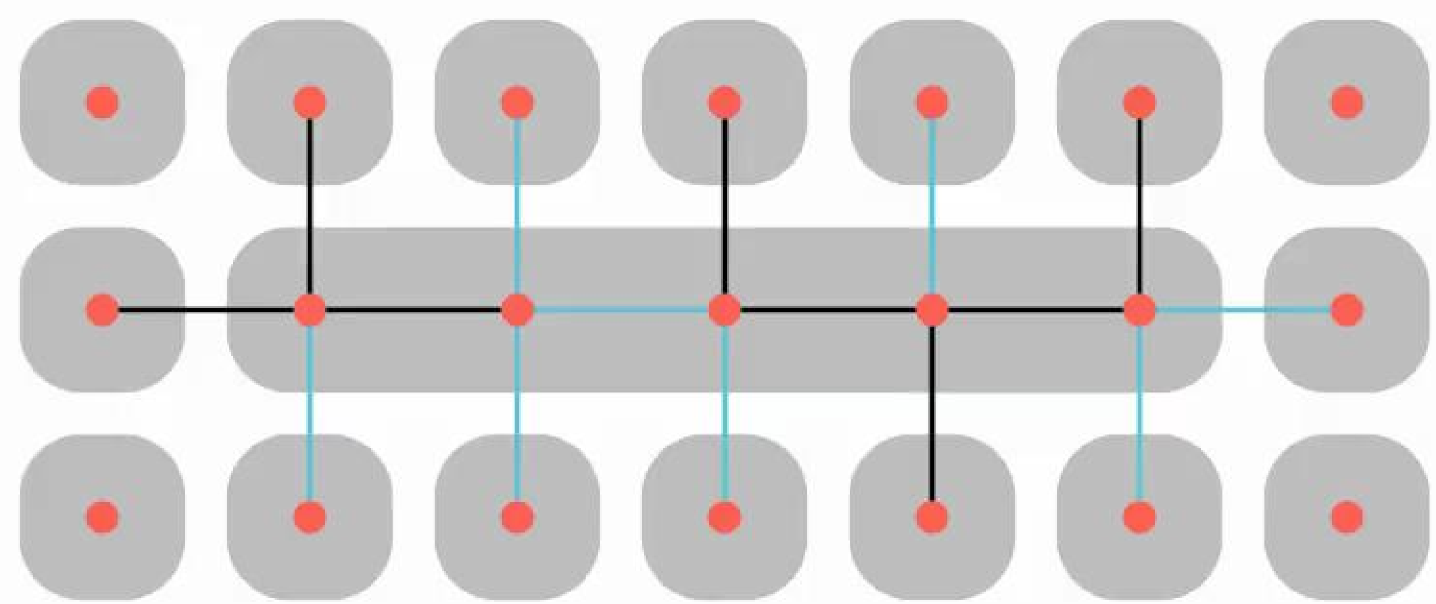
Twin-width

[Bonnet, Kim, Thomassé, Watrigant, 2021]

$$\text{twin-width}(G) \leqslant k \quad ^*$$

$$\Updownarrow$$

$\exists$ construction sequence such that each *part* neighbors $\leqslant k$ other parts



*differs by ± 1 from original definition

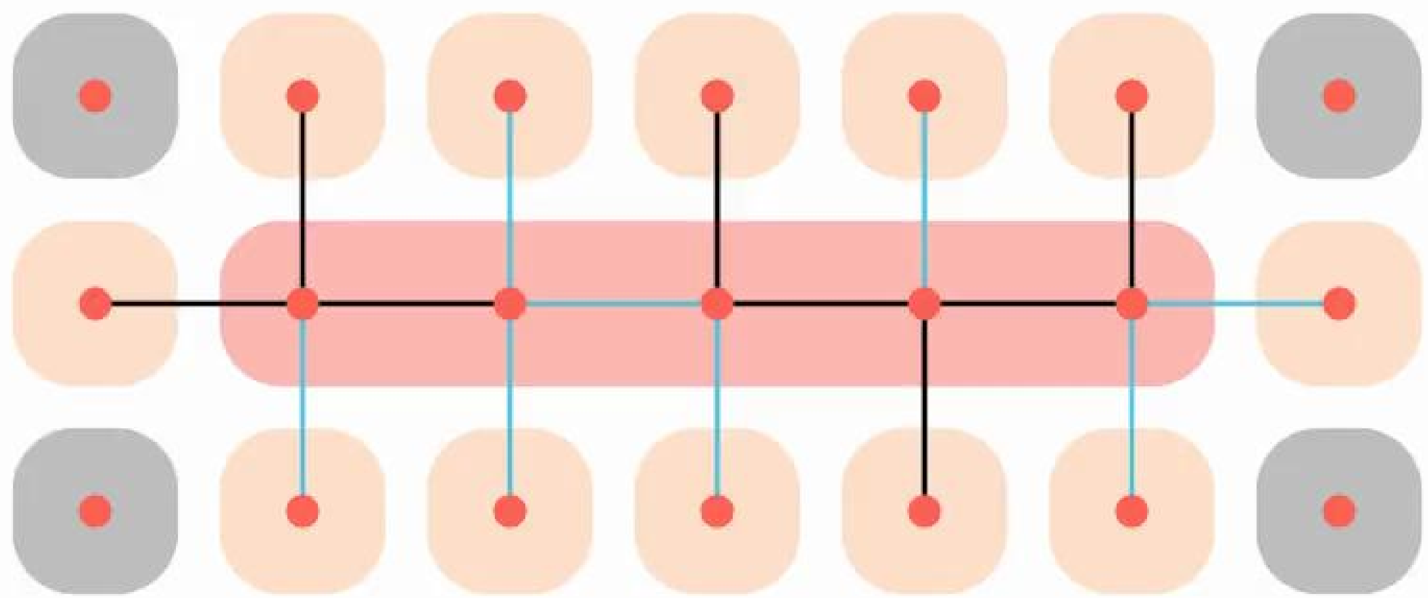
Twin-width

[Bonnet, Kim, Thomassé, Watrigant, 2021]

$$\text{twin-width}(G) \leqslant k \quad ^{*}$$

$$\Updownarrow$$

$\exists$ construction sequence such that each *part* neighbors $\leqslant k$ other parts



*differs by ± 1 from original definition

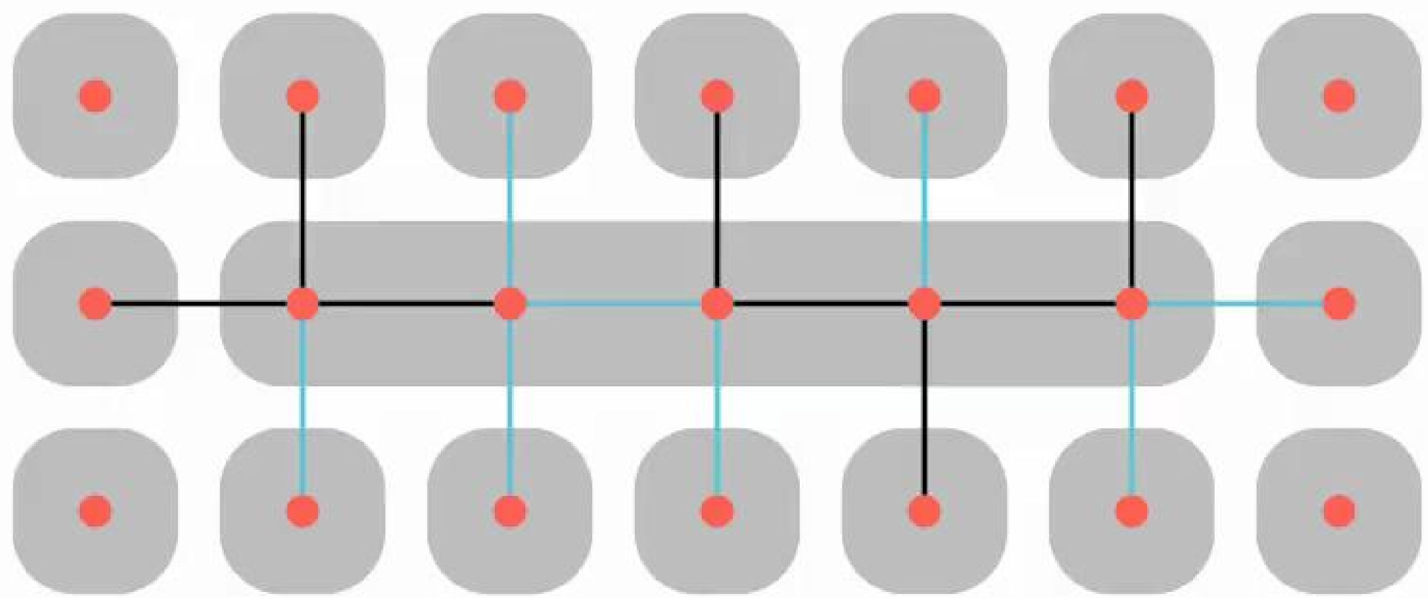
Twin-width

[Bonnet, Kim, Thomassé, Watrigant, 2021]

$$\text{twin-width}(G) \leqslant k \quad ^*$$

$$\Updownarrow$$

$\exists$ construction sequence such that each *part* neighbors $\leqslant k$ other parts



*differs by ± 1 from original definition

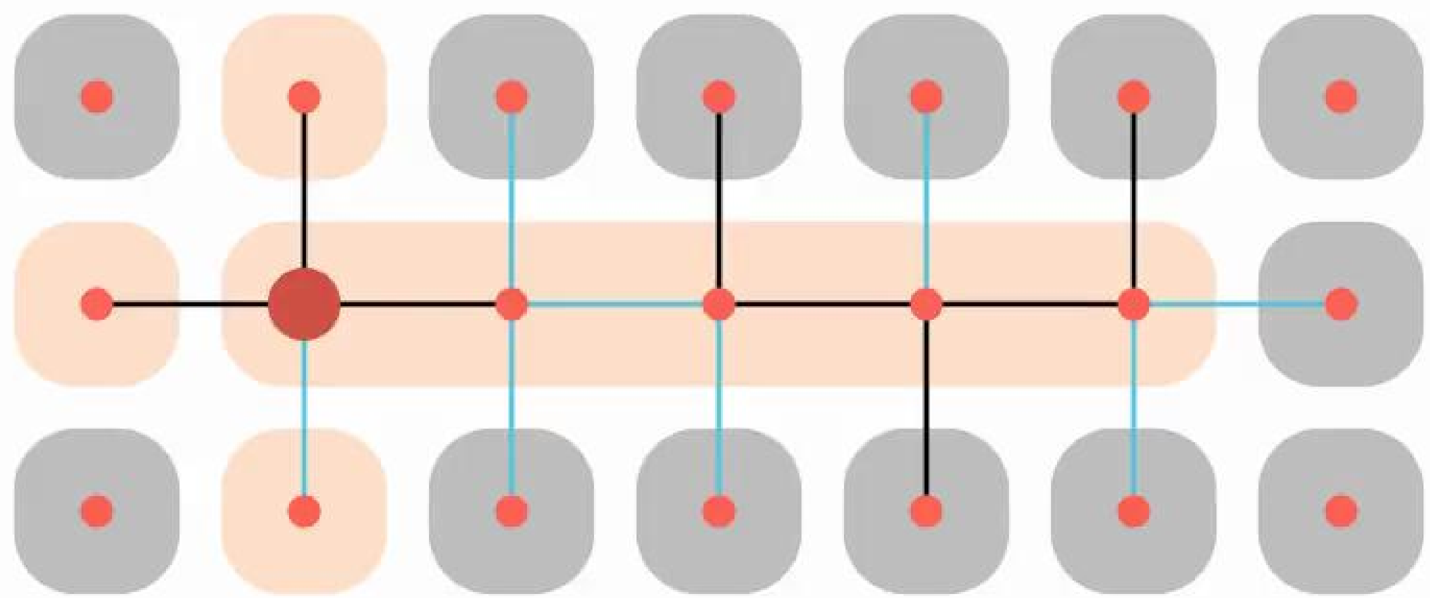
Twin-width

[Bonnet, Kim, Thomassé, Watrigant, 2021]

$$\text{twin-width}(G) \leqslant k \quad ^*$$

$$\Updownarrow$$

$\exists$ construction sequence such that each *part* neighbors $\leqslant k$ other parts



*differs by ± 1 from original definition

Twin-width

[Bonnet, Kim, Thomassé, Watrigant, 2021]

$$\text{twin-width}(G) \leqslant k \quad ^*$$

$$\Updownarrow$$

$\exists$ construction sequence such that each *part* neighbors $\leqslant k$ other parts

*differs by ± 1 from original definition

Twin-width

[Bonnet, Kim, Thomassé, Watrigant, 2021]

$$\text{twin-width}(G) \leqslant k \quad {}^*$$

$$\Updownarrow$$

$\exists$ construction sequence such that each *part* neighbors $\leqslant k$ other parts

In this case, $\text{mw}_r(G) \leqslant O(k^r)$

* differs by ± 1 from original definition

Twin-width

[Bonnet, Kim, Thomassé, Watrigant, 2021]

$$\text{twin-width}(G) \leqslant k \quad ^*$$

$$\Updownarrow$$

$\exists$ construction sequence such that each *part* neighbors $\leqslant k$ other parts

In this case, $\mathrm{mw}_r(G) \leqslant O(k^r)$

Corollary. Every class of bounded twin-width has bounded merge-width.

*differs by ± 1 from original definition

Twin-width

[Bonnet, Kim, Thomassé, Watrigant, 2021]

$$\text{twin-width}(G) \leqslant k \quad ^*$$

$$\Updownarrow$$

$\exists$ construction sequence such that each *part* neighbors $\leqslant k$ other parts

In this case, $\mathrm{mw}_r(G) \leqslant O(k^r)$

Corollary. Every class of bounded twin-width has bounded merge-width.

This includes:

- every proper minor-closed graph class,

*differs by ± 1 from original definition

Twin-width

[Bonnet, Kim, Thomassé, Watrigant, 2021]

$$\text{twin-width}(G) \leqslant k \quad ^*$$

$$\Updownarrow$$

$\exists$ construction sequence such that each *part* neighbors $\leqslant k$ other parts

In this case, $\mathrm{mw}_r(G) \leqslant O(k^r)$

Corollary. Every class of bounded twin-width has bounded merge-width.

This includes:

- every proper minor-closed graph class,
- every proper hereditary class of permutation graphs.

*differs by ± 1 from original definition

Degeneracy

$$\mathrm{degeneracy}(G) \leqslant d$$

$$\Updownarrow$$

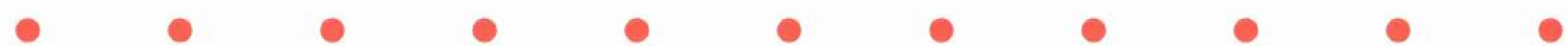
$\exists$ total order on $V(G)$ such that each vertex has $\leqslant d$ neighbors before it

Degeneracy

$$\mathrm{degeneracy}(G) \leqslant d$$

$$\Updownarrow$$

$\exists$ total order on $V(G)$ such that each vertex has $\leqslant d$ neighbors before it

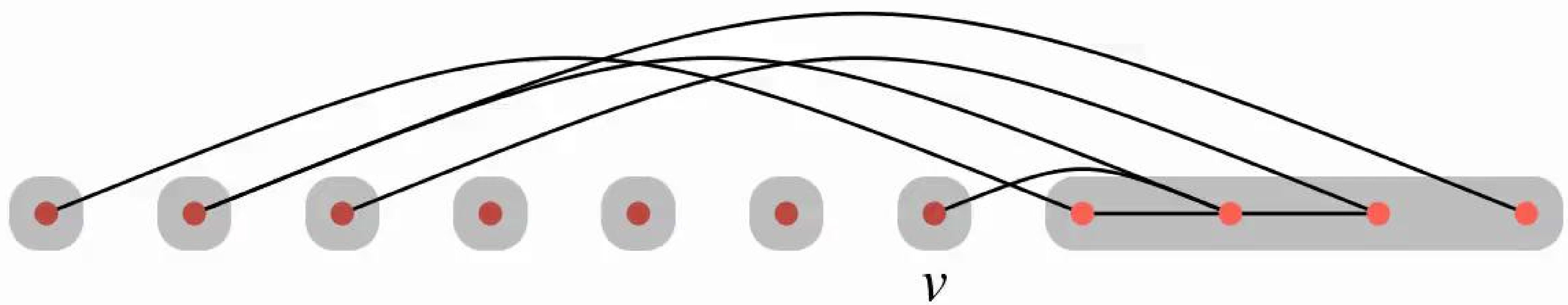


Degeneracy

$$\text{degeneracy}(G) \leqslant d$$

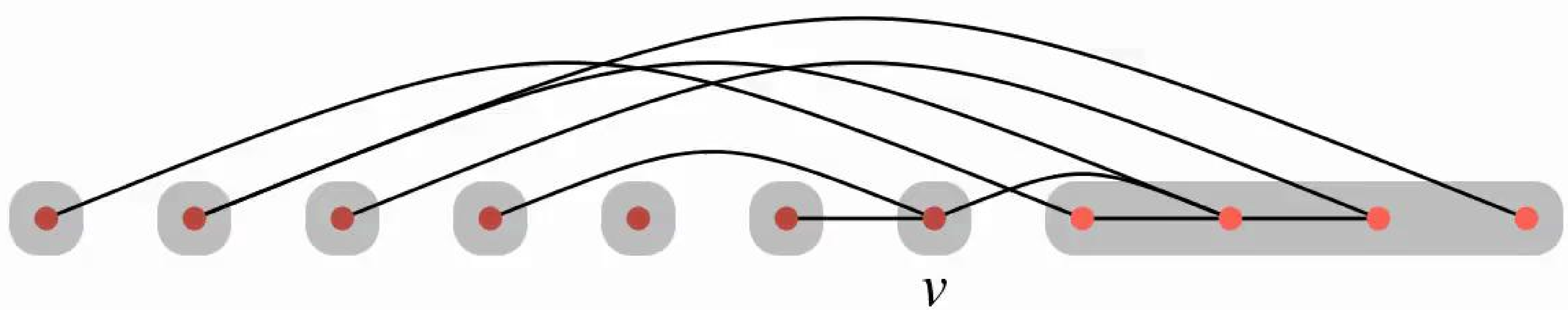
$$\Updownarrow$$

$\exists$ total order on $V(G)$ such that each vertex has $\leqslant d$ neighbors before it



Degeneracy

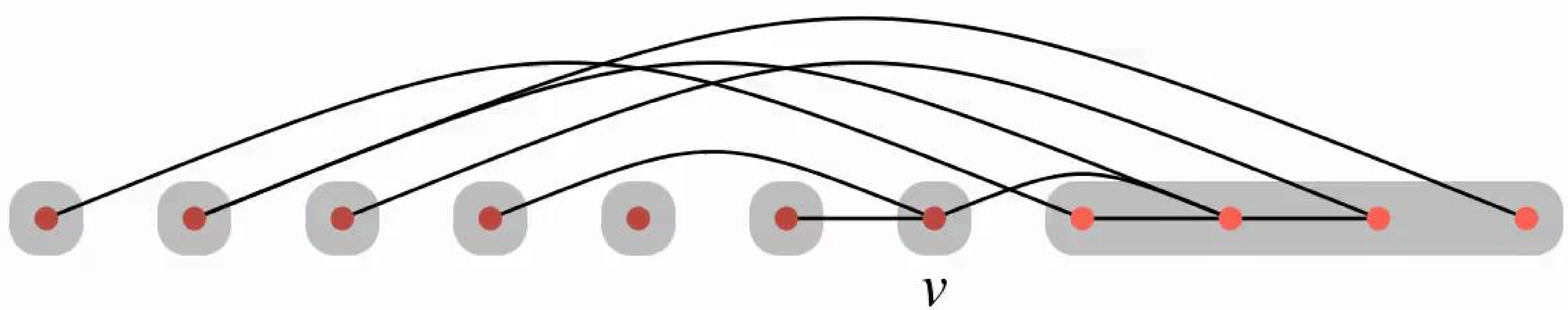
$$\text{degeneracy}(G) \leqslant d$$
$$\Updownarrow$$
$\exists$ total order on $V(G)$ such that each vertex has $\leqslant d$ neighbors before it



resolve $\{v\}$ with $\leqslant d$ parts to the left

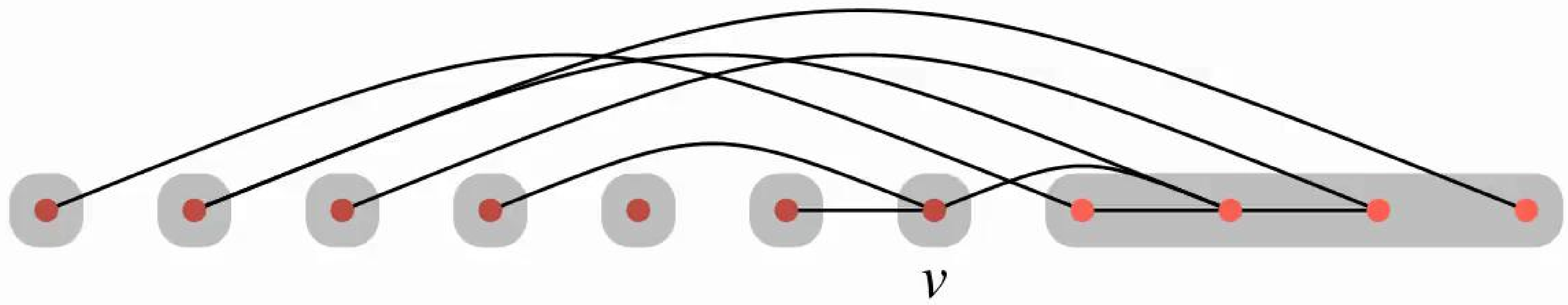
Degeneracy

$$\text{degeneracy}(G) \leqslant d$$
$$\Updownarrow$$
$\exists$ total order on $V(G)$ such that each vertex has $\leqslant d$ neighbors before it



$$\text{mw}_1(G) \leqslant 2 + \text{degeneracy}(G)$$

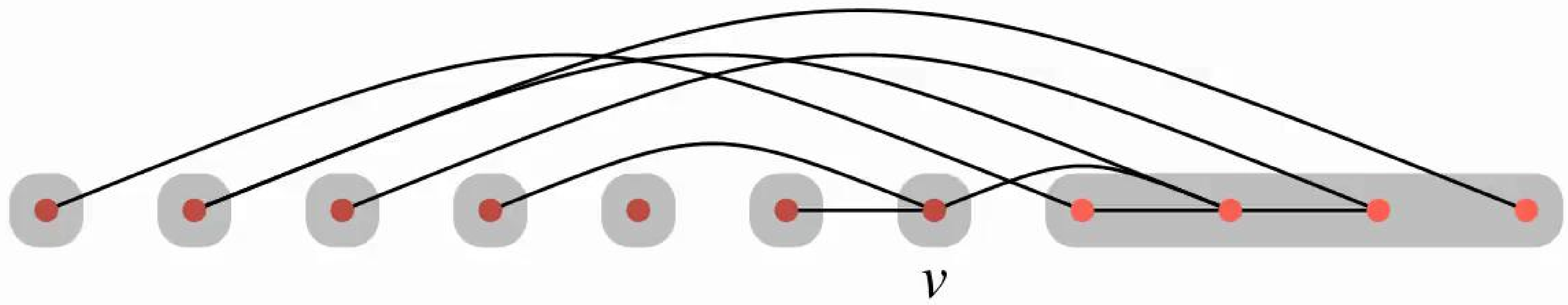
$\exists$ total order on $V(G)$ such that each vertex has $\leqslant d$ neighbors before it



$$\mathrm{mw}_1(G) \leqslant 2 + \mathrm{degeneracy}(G)$$

Theorem. Every class of *bounded expansion* has bounded merge-width.

∃ total order on $V(G)$ such that each vertex has $\leqslant d$ neighbors before it



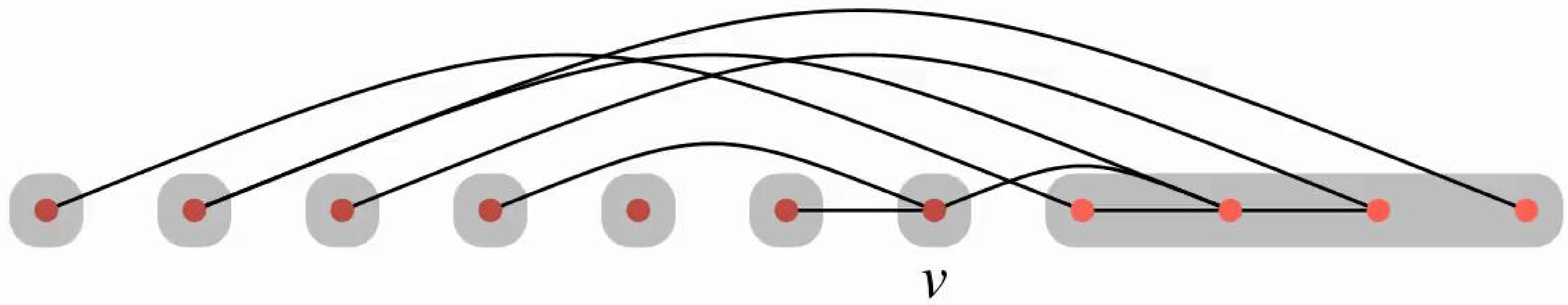
$$\mathrm{mw}_1(G) \leqslant 2 + \mathrm{degeneracy}(G)$$

Theorem. Every class of *bounded expansion* has bounded merge-width.

This includes:

- every graph class which excludes a fixed graph as a minor,

$\exists$ total order on $V(G)$ such that each vertex has $\leqslant d$ neighbors before it

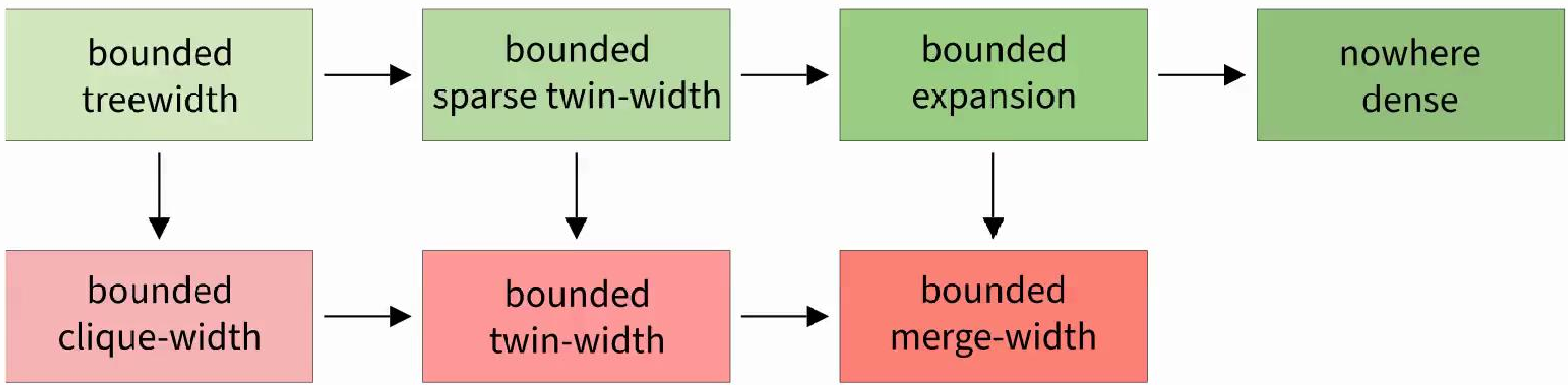


$$\mathrm{mw}_1(G) \leqslant 2 + \mathrm{degeneracy}(G)$$

Theorem. Every class of *bounded expansion* has bounded merge-width.

This includes:

- every graph class which excludes a fixed graph as a minor,
- or as a topological minor.


bounded treewidth
bounded sparse twin-width
bounded expansion
nowhere dense
bounded clique-width
bounded twin-width
bounded merge-width

First-Order Model Checking

fpt algorithm for FO-definable properties

Input: a graph G and first-order sentence φ

First-Order Model Checking

fpt algorithm for FO-definable properties

Input: a graph G and first-order sentence φ

Decide: does φ hold in G?

First-Order Model Checking

fpt algorithm for FO-definable properties

Input: a graph G and first-order sentence φ

Decide: does φ hold in G?

Examples:

- does G contain a clique of size k?

First-Order Model Checking

fpt algorithm for FO-definable properties

Input: a graph G and first-order sentence φ

Decide: does φ hold in G?

Examples:

- does G contain a clique of size k?
- does G contain an independent set of size k?

First-Order Model Checking

fpt algorithm for FO-definable properties

Input: a graph G and first-order sentence φ

Decide: does φ hold in G?

Examples:

- does G contain a clique of size k?
- does G contain an independent set of size k?
- does G contain a dominating set of size k?

Input: a graph G and first-order sentence φ

Decide: does φ hold in G?

Examples:

- does G contain a clique of size k?
- does G contain an independent set of size k?
- does G contain a dominating set of size k?

A graph class C is *tractable* if model checking is fixed-parameter tractable (fpt) on C – can be decided in time

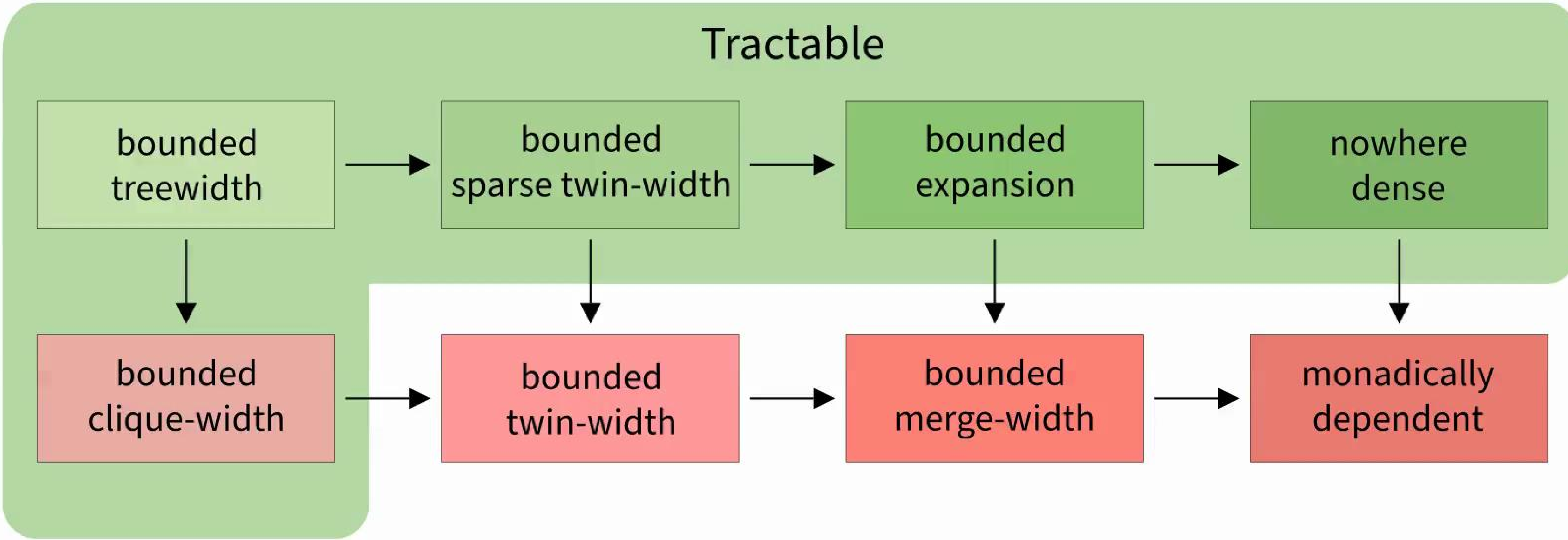
$$f(\varphi) \cdot |V(G)|^c \qquad \text{for } G \in C$$

A graph class C is *tractable* if model checking is fixed-parameter tractable (fpt) on C – can be decided in time

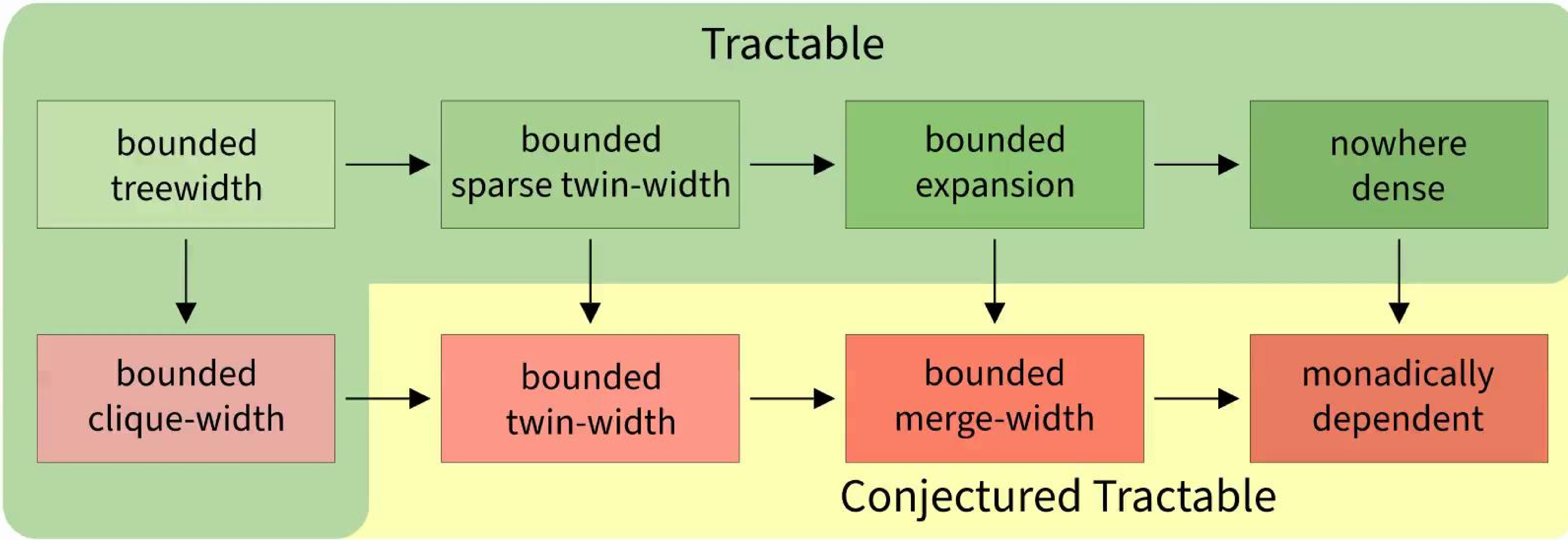
$$f(\varphi) \cdot |V(G)|^c \qquad \text{for } G \in C$$

Main question: which graph classes C are tractable?

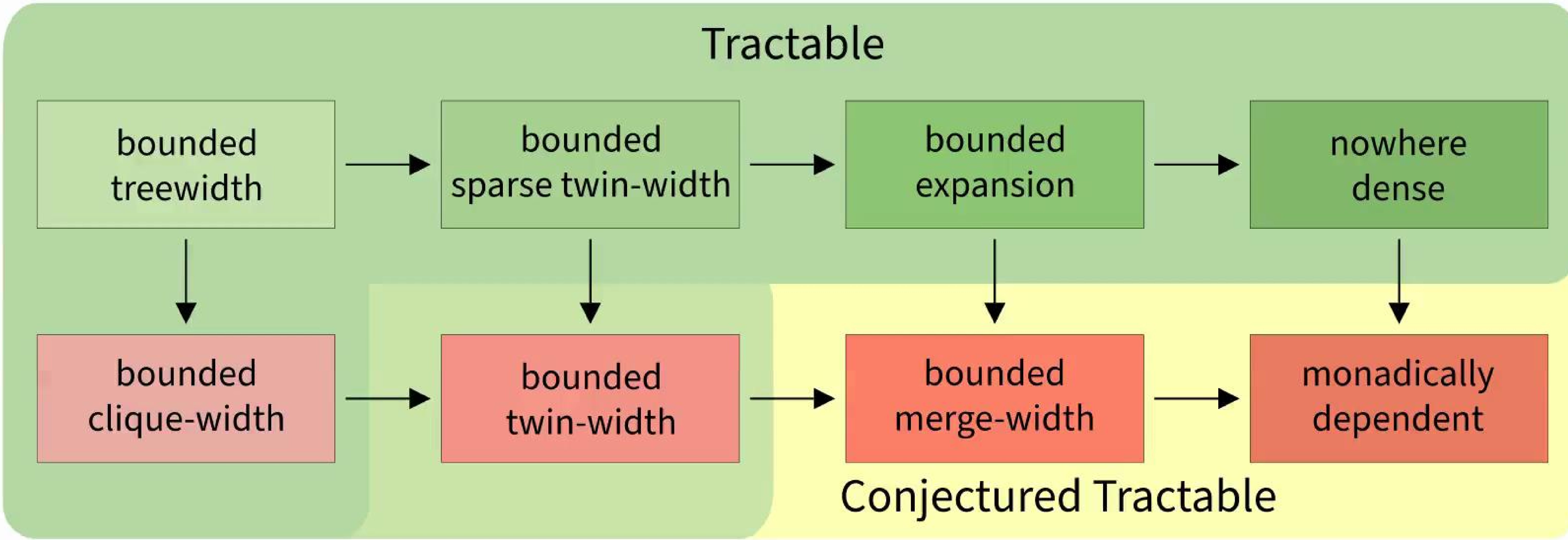
Main question: which graph classes C are tractable?



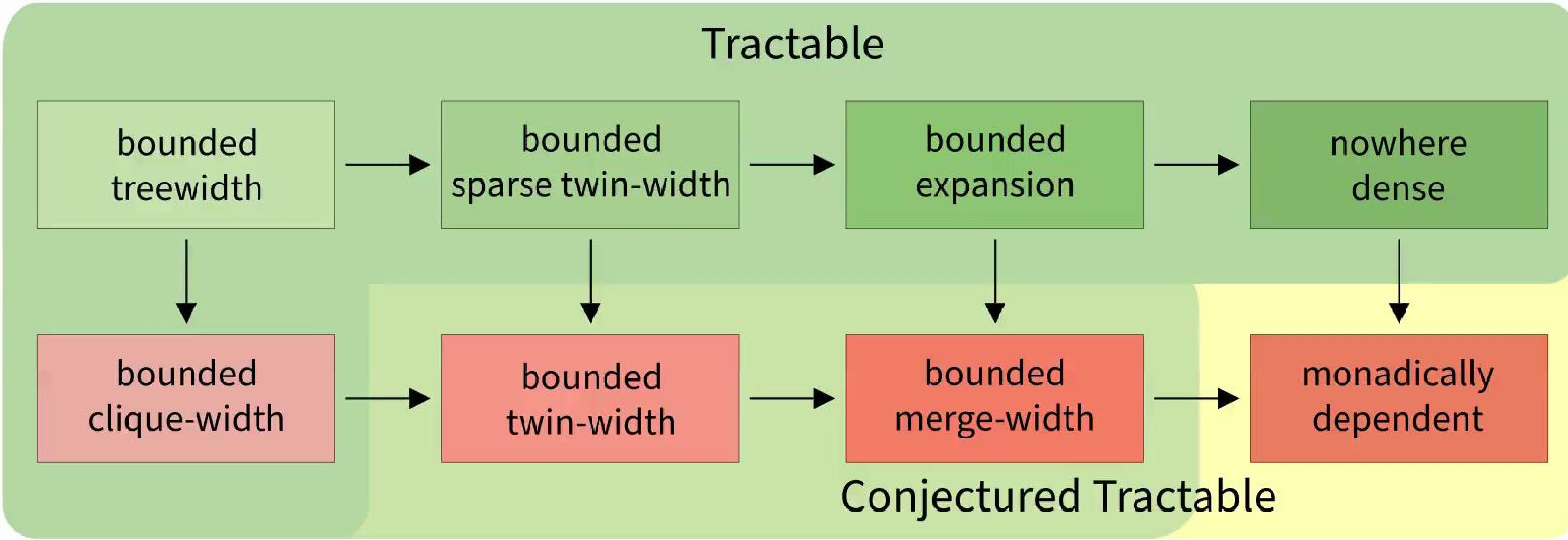
Main question: which graph classes C are tractable?



Main question: which graph classes C are tractable?



Main question: which graph classes C are tractable?



Conjectured Tractable

Main result. "Graph classes of bounded merge-width are tractable, *assuming a construction sequence is provided on input*":

Main result. “Graph classes of bounded merge-width are tractable, *assuming a construction sequence is provided on input*”:

Theorem. There is an algorithm which given:

Main result. "Graph classes of bounded merge-width are tractable, *assuming a construction sequence is provided on input*":

Theorem. There is an algorithm which given:

- a first-order sentence φ of quantifier depth q,

Main result. "Graph classes of bounded merge-width are tractable, *assuming a construction sequence is provided on input*":

Theorem. There is an algorithm which given:

- a first-order sentence φ of quantifier depth q,
- a graph G with a construction sequence C,

Main result. “Graph classes of bounded merge-width are tractable, *assuming a construction sequence is provided on input*”:

Theorem. There is an algorithm which given:

- a first-order sentence φ of quantifier depth q,
- a graph G with a construction sequence C,

decides whether G satisfies φ in time

$$f(q,w) \cdot |V(G)|^3,$$

for w=radius-r width of C and $r = 2^{O(q^2)}$

Proof sketch

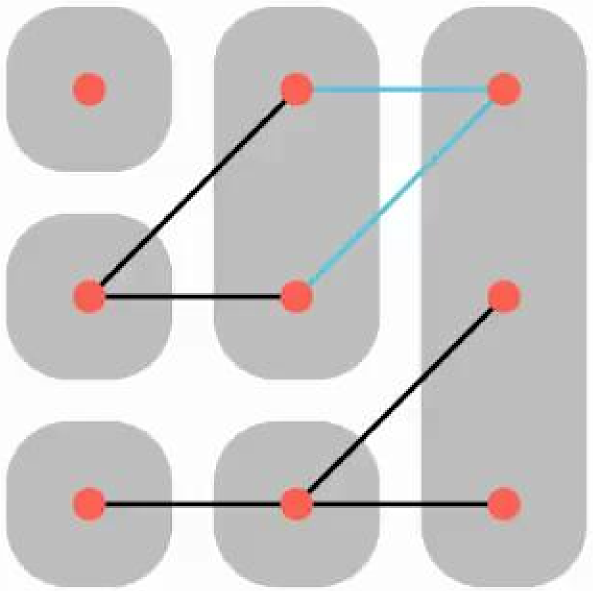
Proof sketch

View a construction sequence as a sequence of structures $G_1, \ldots, G_m$

Proof sketch

View a construction sequence as a sequence of structures $G_1, \ldots, G_m$

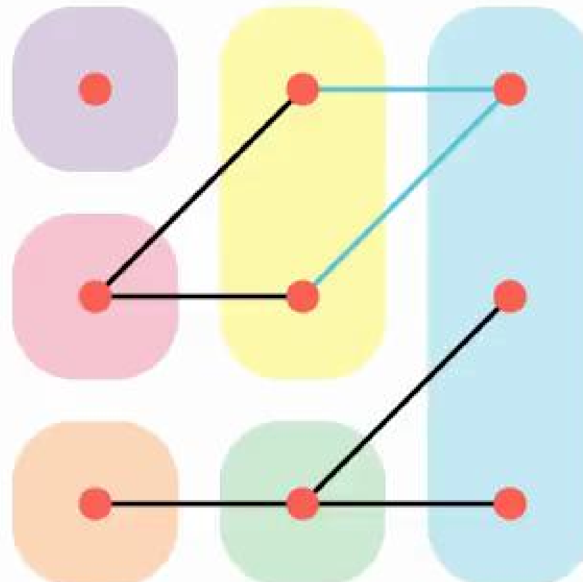
G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.



Proof sketch

View a construction sequence as a sequence of structures $G_1, \ldots, G_m$

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.

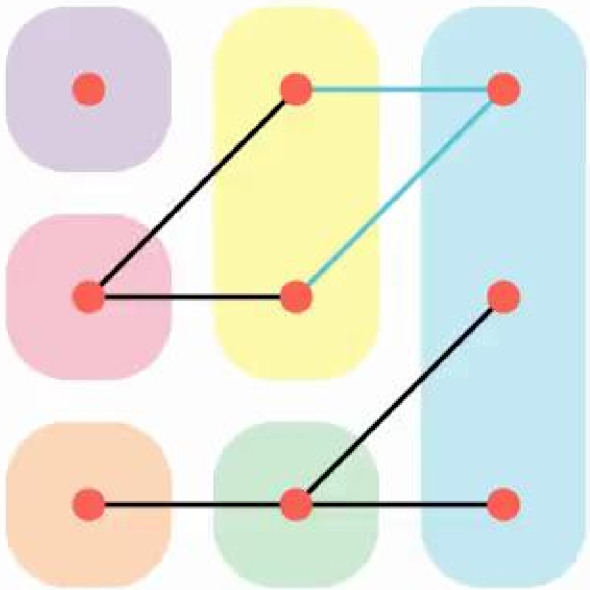


Proof sketch

View a construction sequence as a sequence of structures $G_1, \ldots, G_m$

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.

- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$

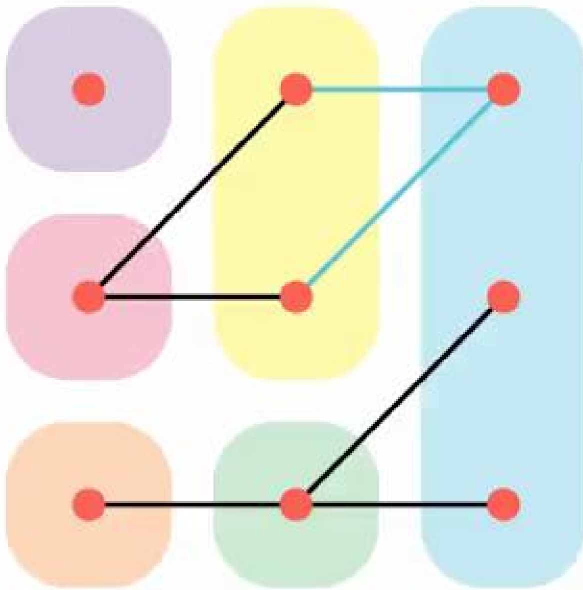


Proof sketch

View a construction sequence as a sequence of structures $G_1, \ldots, G_m$

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.

- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$
- $\#\{\text{unary predicates in an } r\text{-ball}\} \leq \text{radius-}r \text{ width}$

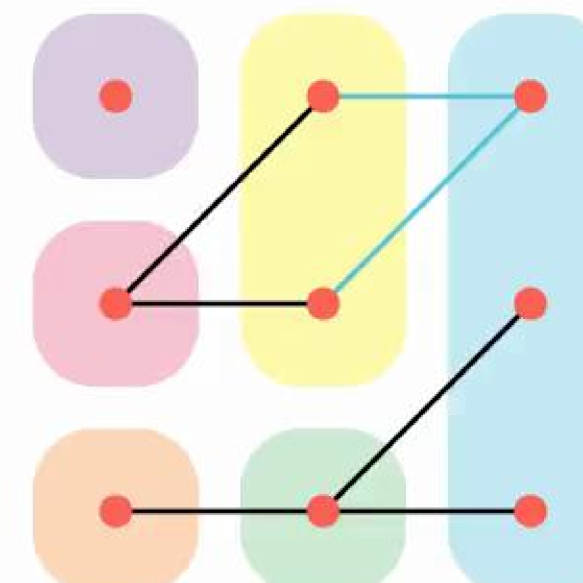


Proof sketch

View a construction sequence as a sequence of structures $G_1, \ldots, G_m$

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.

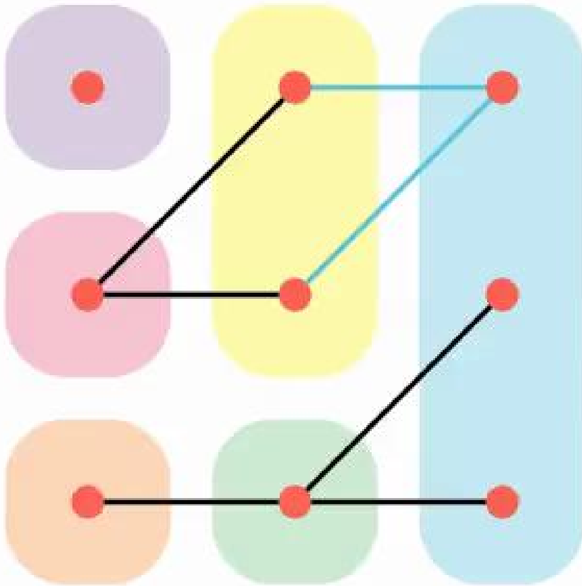
- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$
- $\#\{\text{unary predicates in an } r\text{-ball}\} \le \text{radius-}r \text{ width}$



Locality Theorem

Fix a structure A, vertex $v \in V(A)$, $q \in \mathbb{N}$.

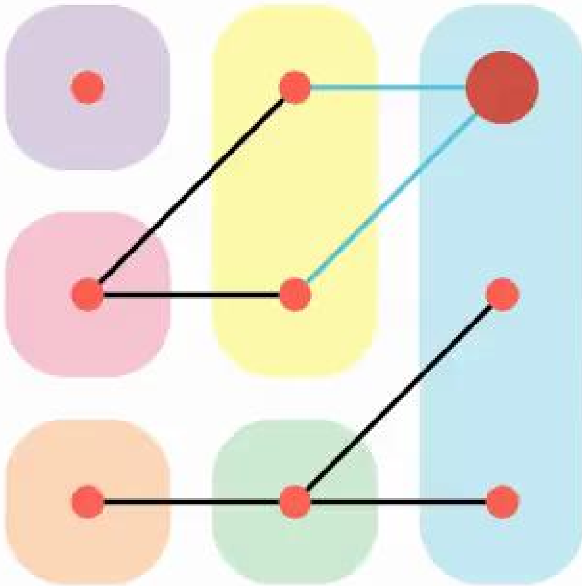
Locality Theorem



Fix a structure A, vertex $v \in V(A)$, $q \in \mathbb{N}$.

$\mathrm{tp}_q(A, v) := \{$formulas $\varphi(x)$ of depth q with $A \models \varphi(v)\}$

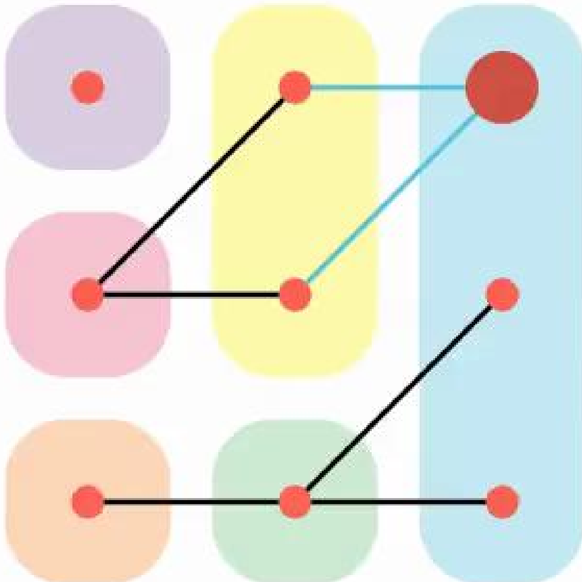
Locality Theorem



Fix a structure A, vertex $v \in V(A)$, $q \in \mathbb{N}$.

$\mathrm{tp}_q(A, v) := \{$formulas $\varphi(x)$ of depth q with $A \models \varphi(v)\}$

Locality Theorem

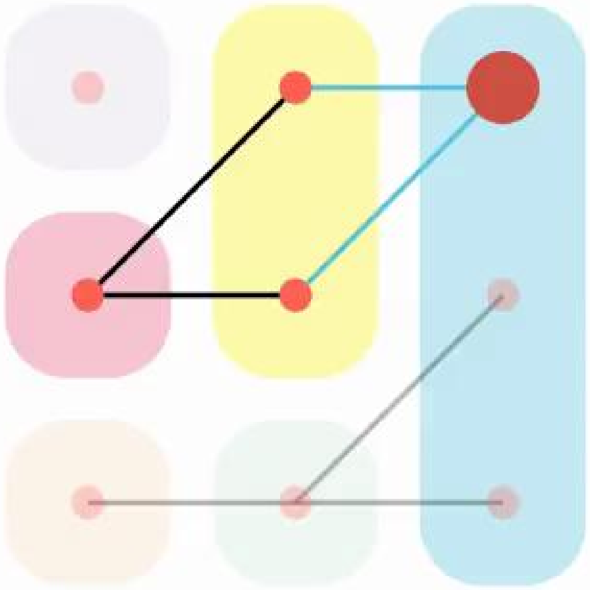


Fix a structure A, vertex $v \in V(A)$, $q \in \mathbb{N}$.

$\mathrm{tp}_q(A, v) := \{$formulas $\varphi(x)$ of depth q with $A \models \varphi(v)\}$

$\mathrm{ltp}_q(A, v) := \{$"local" formulas $\varphi(x)$ of depth q with $G_t \models \varphi(v)\}$
– quantifiers range over r-ball around v in $\mathrm{Gaifman}(A)$, for some $r := r(q)$

Locality Theorem

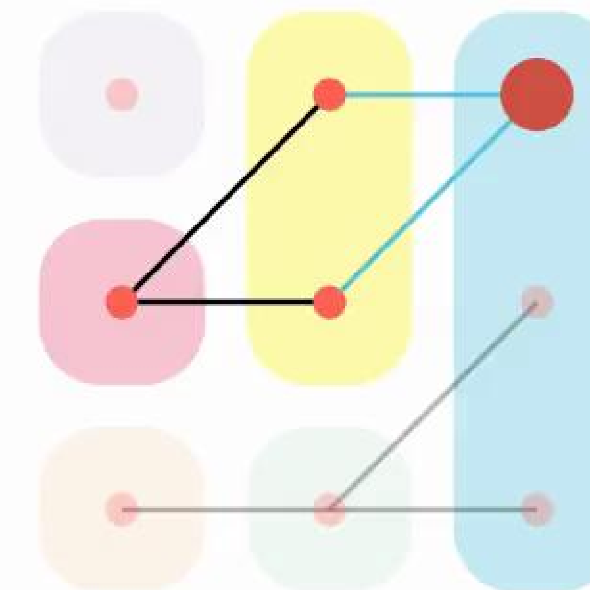


Fix a structure A, vertex $v \in V(A)$, $q \in \mathbb{N}$.

$\mathrm{tp}_q(A, v) := \{$formulas $\varphi(x)$ of depth q with $A \models \varphi(v)\}$

$\mathrm{ltp}_q(A, v) := \{$"local" formulas $\varphi(x)$ of depth q with $G_t \models \varphi(v)\}$
– quantifiers range over r-ball around v in $\mathrm{Gaifman}(A)$, for some $r := r(q)$

Locality Theorem



Fix a structure A, vertex $v \in V(A)$, $q \in \mathbb{N}$.

$\mathrm{tp}_q(A, v) := \{$formulas $\varphi(x)$ of depth q with $A \models \varphi(v)\}$

$\mathrm{ltp}_q(A, v) := \{$"local" formulas $\varphi(x)$ of depth q with $G_t \models \varphi(v)\}$
– quantifiers range over r-ball around v in $\mathrm{Gaifman}(A)$, for some $r := r(q)$

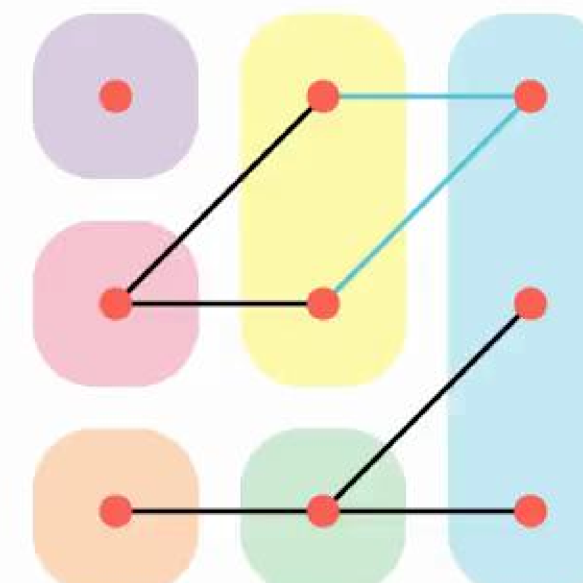
Locality Theorem. $\mathrm{tp}_q(A, v)$ *is determined by* $\mathrm{ltp}_q(A, v)$ *(and global sentences).*

Proof sketch

View a construction sequence as a sequence of structures $G_1, \ldots, G_m$

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.

- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$
- $\#\{\text{unary predicates in an } r\text{-ball}\} \leq \text{radius-}r \text{ width}$



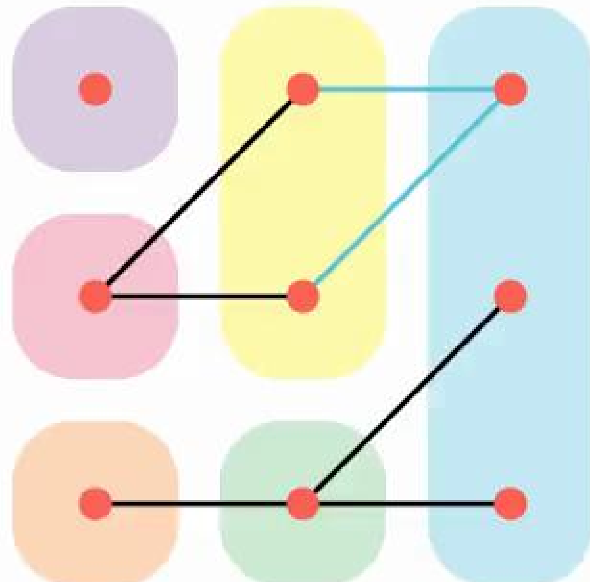
Proof sketch

View a construction sequence as a sequence of structures $G_1, \ldots, G_m$

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.

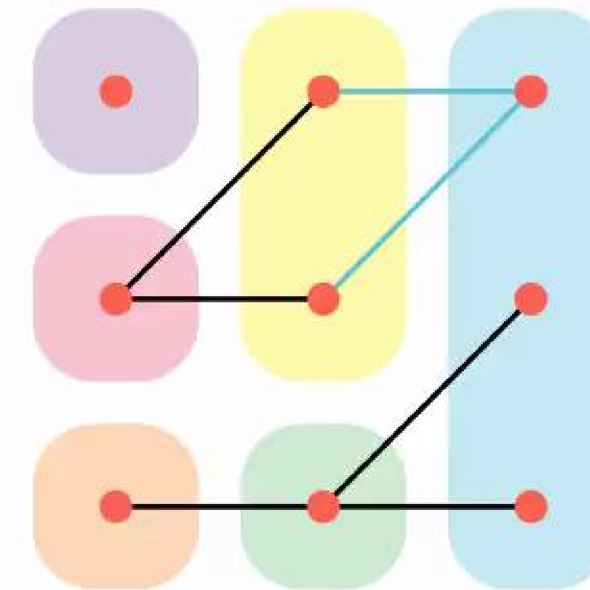
- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$
- $\#\{\text{unary predicates in an } r\text{-ball}\} \leq \text{radius-}r \text{ width}$

$\rightarrow$ each $\mathrm{ltp}_q(G_t, v)$ has size bounded in terms of mw_r



G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.

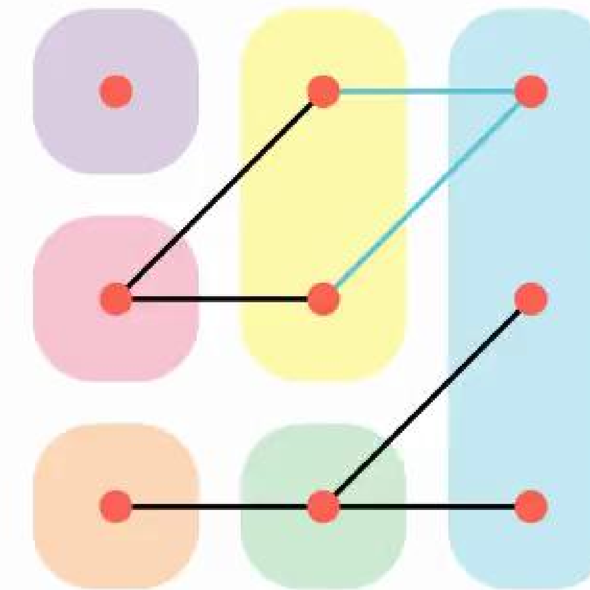
- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$
- $\#\{\text{unary predicates in an } r\text{-ball}\} \leq \text{radius-}r \text{ width}$



$\rightarrow$ each $\mathrm{ltp}_q(G_t, v)$ has size bounded in terms of mw_r

Key Lemma. *Fix $t \in \{1, \ldots, m\}$. For all $v \in V$, $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{ltp}_q(G_{t+1}, v)$.*

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.



- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$
- $\#\{\text{unary predicates in an } r\text{-ball}\} \leq \text{radius-}r \text{ width}$

$\rightarrow$ each $\mathrm{ltp}_q(G_t, v)$ has size bounded in terms of mw_r

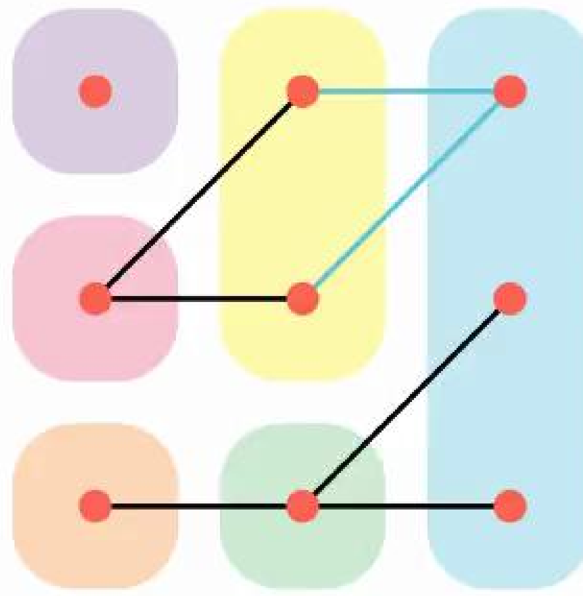
Key Lemma. *Fix $t \in \{1, \ldots, m\}$. For all $v \in V$, $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{ltp}_q(G_{t+1}, v)$.*

Proof.

- $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{tp}_q(G_t, v)$ by locality theorem

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.

- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$
- $\#\{\text{unary predicates in an } r\text{-ball}\} \leq \text{radius-}r \text{ width}$



$\rightarrow$ each $\mathrm{ltp}_q(G_t, v)$ has size bounded in terms of mw_r

Key Lemma. *Fix $t \in \{1, \ldots, m\}$. For all $v \in V$, $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{ltp}_q(G_{t+1}, v)$.*

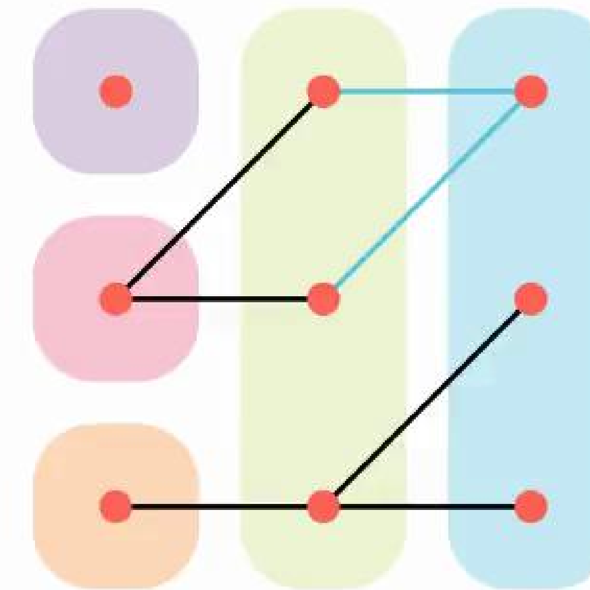
Proof.

- $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{tp}_q(G_t, v)$ by locality theorem
- $\mathrm{tp}_q(G_t, v)$ determines $\mathrm{tp}_q(G_{t+1}, v)$ by formula rewriting:

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.

- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$
- $\#\{\text{unary predicates in an } r\text{-ball}\} \leq \text{radius-}r \text{ width}$

$\rightarrow$ each $\mathrm{ltp}_q(G_t, v)$ has size bounded in terms of mw_r



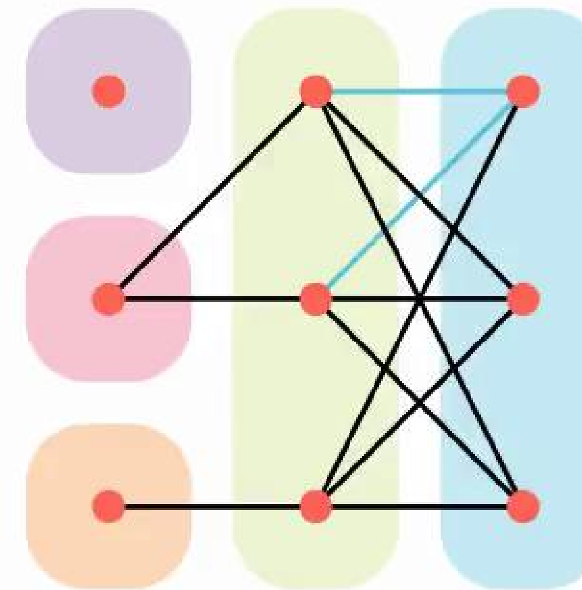
Key Lemma. *Fix $t \in \{1, \ldots, m\}$. For all $v \in V$, $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{ltp}_q(G_{t+1}, v)$.*

Proof.

- $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{tp}_q(G_t, v)$ by locality theorem
- $\mathrm{tp}_q(G_t, v)$ determines $\mathrm{tp}_q(G_{t+1}, v)$ by formula rewriting:

  $\mathrm{merge}(P, Q)$: $\qquad (P \cup Q)(x) \Leftrightarrow P(x) \vee Q(x)$

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.



- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$
- $\#\{\text{unary predicates in an } r\text{-ball}\} \leq \text{radius-}r \text{ width}$

$\rightarrow$ each $\mathrm{ltp}_q(G_t, v)$ has size bounded in terms of mw_r

Key Lemma. *Fix $t \in \{1, \ldots, m\}$. For all $v \in V$, $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{ltp}_q(G_{t+1}, v)$.*

Proof.

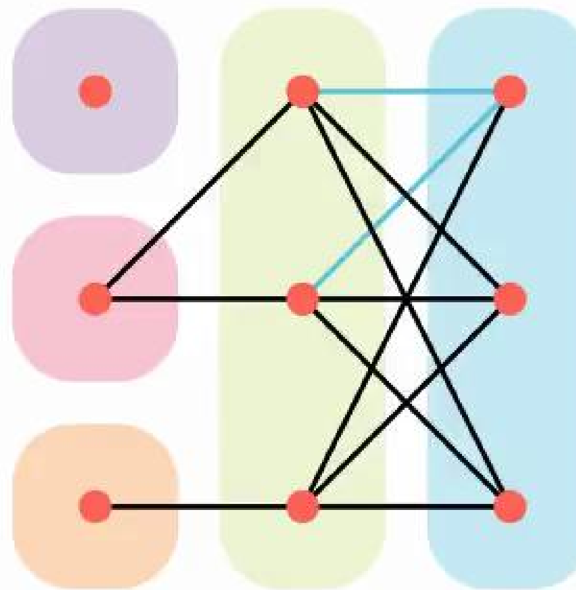
- $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{tp}_q(G_t, v)$ by locality theorem
- $\mathrm{tp}_q(G_t, v)$ determines $\mathrm{tp}_q(G_{t+1}, v)$ by formula rewriting:

$$\mathrm{merge}(P, Q): \quad (P \cup Q)(x) \Leftrightarrow P(x) \vee Q(x)$$

$$\mathrm{resolve}_+(P, Q): \quad E_{t+1}(x, y) \Leftrightarrow E_t(x, y) \vee (P(x) \wedge Q(y) \wedge \neg N_t(x, y)).$$

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.

- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$
- $\#\{\text{unary predicates in an } r\text{-ball}\} \le \text{radius-}r \text{ width}$



$\rightarrow$ each $\mathrm{ltp}_q(G_t, v)$ has size bounded in terms of mw_r

Key Lemma. *Fix $t \in \{1, \dots, m\}$. For all $v \in V$, $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{ltp}_q(G_{t+1}, v)$.*

Proof.

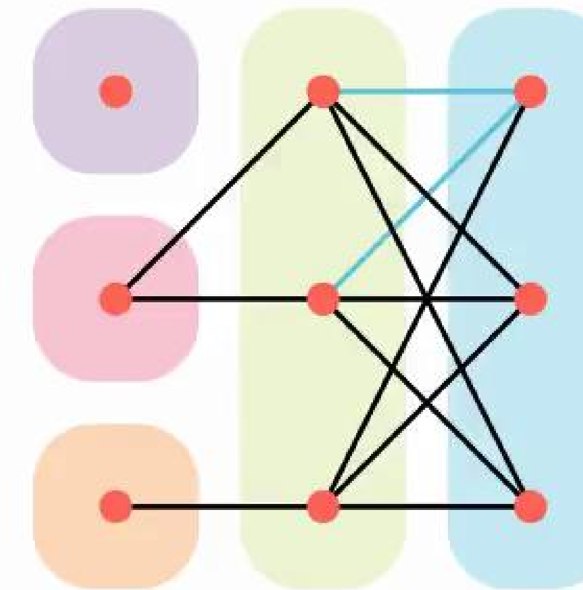
- $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{tp}_q(G_t, v)$ by locality theorem
- $\mathrm{tp}_q(G_t, v)$ determines $\mathrm{tp}_q(G_{t+1}, v)$ by formula rewriting:

$$\mathrm{merge}(P,Q): \qquad (P \cup Q)(x) \Leftrightarrow P(x) \vee Q(x)$$
$$\mathrm{resolve}_+(P,Q): \qquad E_{t+1}(x,y) \Leftrightarrow E_t(x,y) \vee (P(x) \wedge Q(y) \wedge \neg N_t(x,y)).$$

- $\mathrm{tp}_q(G_{t+1}, v)$ determines $\mathrm{ltp}_q(G_{t+1}, v)$. ■

G_t: binary relations E_t, N_t,
unary relation $P(\cdot)$ for each $P \in \mathcal{P}_t$.

- $\mathrm{Gaifman}(G_t) = (V, E_t \cup N_t)$
- $\#\{\text{unary predicates in an } r\text{-ball}\} \le \text{radius-}r \text{ width}$



$\to$ each $\mathrm{ltp}_q(G_t, v)$ has size bounded in terms of mw_r

Key Lemma. *Fix $t \in \{1, \ldots, m\}$. For all $v \in V$, $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{ltp}_q(G_{t+1}, v)$.*

Proof.

- $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{tp}_q(G_t, v)$ by locality theorem
- $\mathrm{tp}_q(G_t, v)$ determines $\mathrm{tp}_q(G_{t+1}, v)$ by formula rewriting:

$$\mathrm{merge}(P,Q): \quad (P \cup Q)(x) \Leftrightarrow P(x) \vee Q(x)$$

$$\mathrm{resolve}_+(P,Q): \quad E_{t+1}(x,y) \Leftrightarrow E_t(x,y) \vee (P(x) \wedge Q(y) \wedge \neg N_t(x,y)).$$

- $\mathrm{tp}_q(G_{t+1}, v)$ determines $\mathrm{ltp}_q(G_{t+1}, v)$. ■

Remark. *This can be computed in fpt time.*

Key Lemma. *Fix* $t \in \{1, \ldots, m\}$. *For all* $v \in V$, $\mathrm{ltp}_q(G_t, v)$ *determines* $\mathrm{ltp}_q(G_{t+1}, v)$.

Algorithm.

Key Lemma. *Fix* $t \in \{1, \ldots, m\}$*. For all* $v \in V$, $\mathrm{ltp}_q(G_t, v)$ *determines* $\mathrm{ltp}_q(G_{t+1}, v)$.

Algorithm.

Input: construction sequence $G_1, \ldots, G_m$ and sentence φ.

Key Lemma. *Fix* $t \in \{1, \ldots, m\}$. *For all* $v \in V$, $\mathrm{ltp}_q(G_t, v)$ *determines* $\mathrm{ltp}_q(G_{t+1}, v)$.

Algorithm.

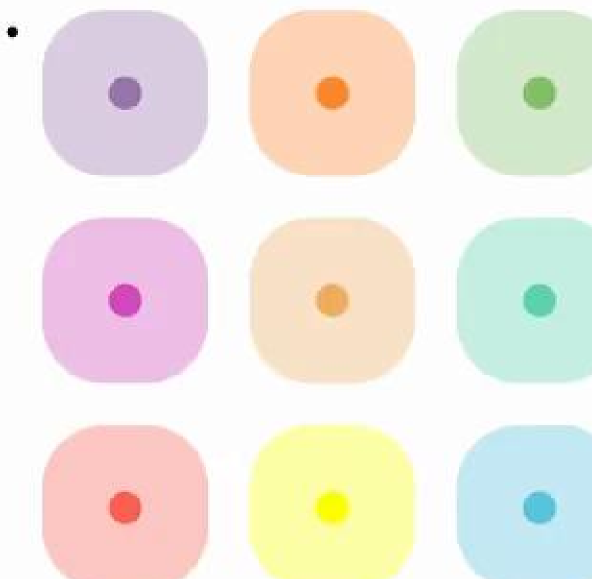
Input: construction sequence $G_1, \ldots, G_m$ and sentence φ.

$q :=$ quantifier depth of φ
$r := r(q)$
for $t = 1, \ldots, m$
    for $v \in V$ compute $\mathrm{ltp}_q(G_t, v)$.

Key Lemma. *Fix* $t \in \{1, \ldots, m\}$. *For all* $v \in V$, $\mathrm{ltp}_q(G_t, v)$ *determines* $\mathrm{ltp}_q(G_{t+1}, v)$.

Algorithm.

Input: construction sequence $G_1, \ldots, G_m$ and sentence φ.

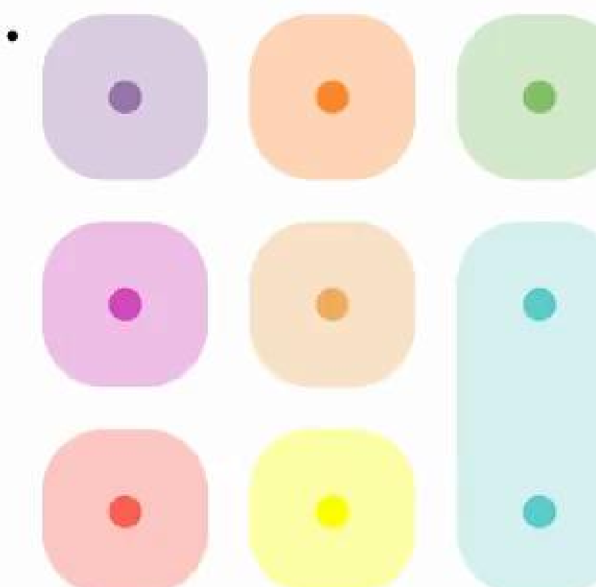


$q :=$ quantifier depth of φ
$r := r(q)$
for $t = 1, \ldots, m$
    for $v \in V$ compute $\mathrm{ltp}_q(G_t, v)$.

Key Lemma. *Fix* $t \in \{1, \ldots, m\}$. *For all* $v \in V$, $\mathrm{ltp}_q(G_t, v)$ *determines* $\mathrm{ltp}_q(G_{t+1}, v)$.

Algorithm.

Input: construction sequence $G_1, \ldots, G_m$ and sentence φ.

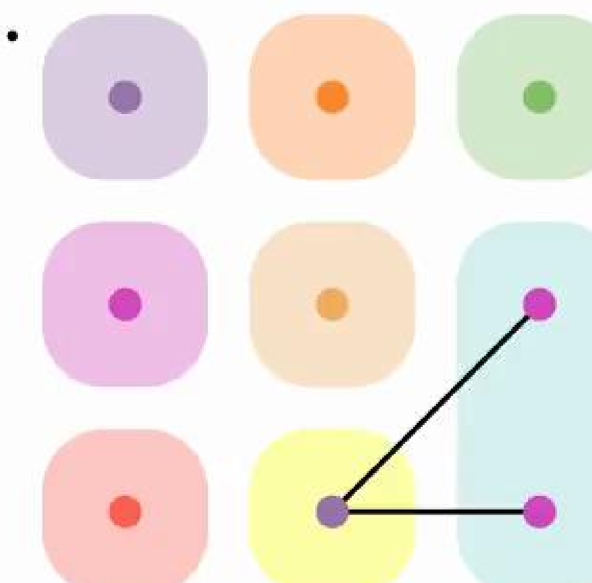


$q :=$ quantifier depth of φ
$r := r(q)$
for $t = 1, \ldots, m$
    for $v \in V$ compute $\mathrm{ltp}_q(G_t, v)$.

Key Lemma. *Fix* $t \in \{1, \ldots, m\}$. *For all* $v \in V$, $\mathrm{ltp}_q(G_t, v)$ *determines* $\mathrm{ltp}_q(G_{t+1}, v)$.

Algorithm.

Input: construction sequence $G_1, \ldots, G_m$ and sentence φ.

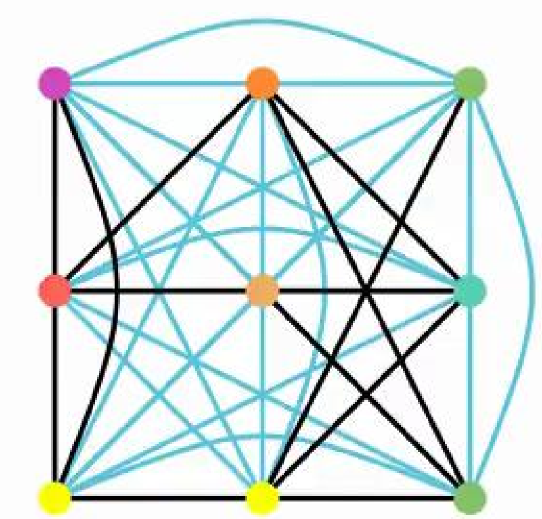


q := quantifier depth of φ
$r := r(q)$
for $t = 1, \ldots, m$
    for $v \in V$ compute $\mathrm{ltp}_q(G_t, v)$.

Key Lemma. *Fix $t \in \{1, \ldots, m\}$. For all $v \in V$, $\mathrm{ltp}_q(G_t, v)$ determines $\mathrm{ltp}_q(G_{t+1}, v)$.*

Algorithm.

Input: construction sequence $G_1, \ldots, G_m$ and sentence φ.



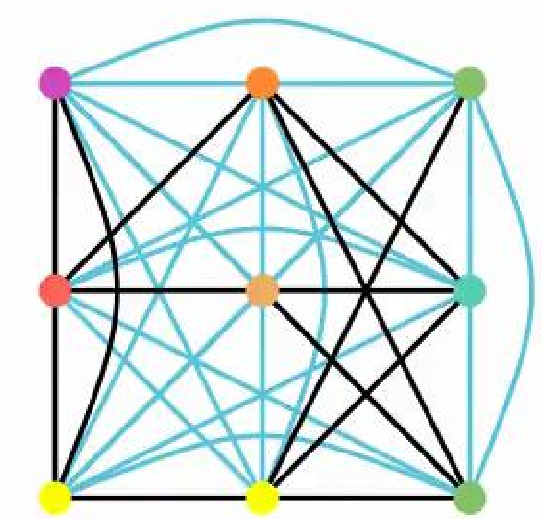
q := quantifier depth of φ
$r := r(q)$
for $t = 1, \ldots, m$
    for $v \in V$ compute $\mathrm{ltp}_q(G_t, v)$.

$\mathrm{Gaifman}(G_m)$ is a clique, so $\mathrm{ltp}_q(G_m, v) = \mathrm{tp}_q(G_m, v)$.

Key Lemma. *Fix* $t \in \{1, \ldots, m\}$. *For all* $v \in V$, $\mathrm{ltp}_q(G_t, v)$ *determines* $\mathrm{ltp}_q(G_{t+1}, v)$.

Algorithm.

Input: construction sequence $G_1, \ldots, G_m$ and sentence φ.

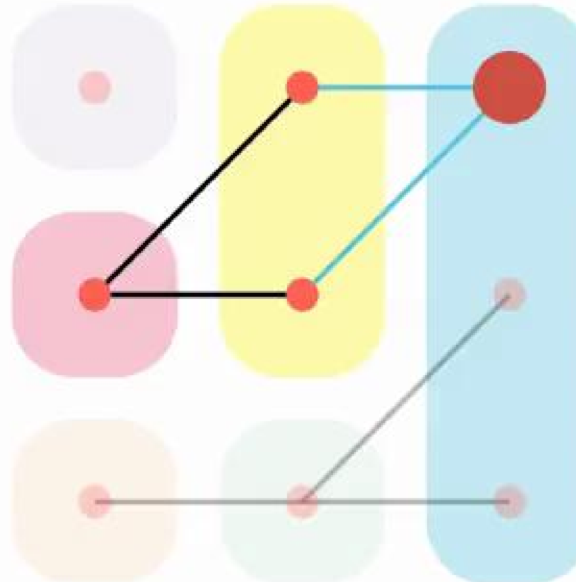


q := quantifier depth of φ
$r := r(q)$
for $t = 1, \ldots, m$
  for $v \in V$ compute $\mathrm{ltp}_q(G_t, v)$.

$\mathrm{Gaifman}(G_m)$ is a clique, so $\mathrm{ltp}_q(G_m, v) = \mathrm{tp}_q(G_m, v)$.

Output: 'yes' iff $\varphi(x) \in \mathrm{ltp}_q(G_m, v_1)$.

Locality Theorem



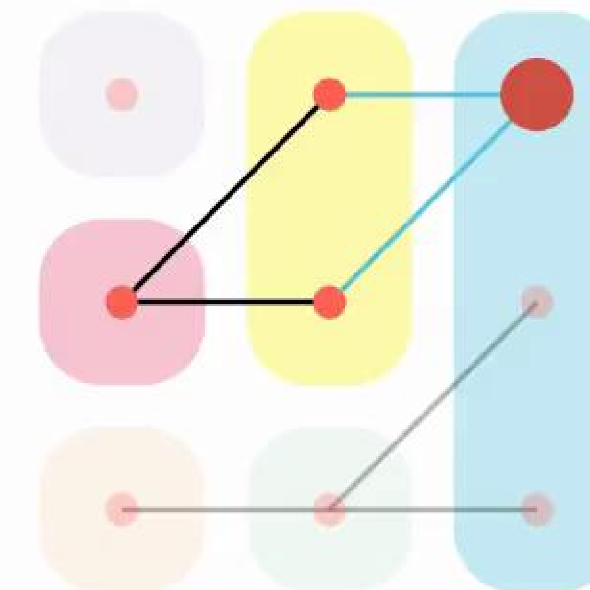
Fix a structure A, vertex $v \in V(A)$, $q \in \mathbb{N}$.

$\mathrm{tp}_q(A, v) := \{$formulas $\varphi(x)$ of depth q with $A \models \varphi(v)\}$

$\mathrm{ltp}_q(A, v) := \{$"local" formulas $\varphi(x)$ of depth q with $G_t \models \varphi(v)\}$
– quantifiers range over r-ball around v in $\mathrm{Gaifman}(A)$, for some $r := r(q)$

Locality Theorem. $\mathrm{tp}_q(A, v)$ *is determined by* $\mathrm{ltp}_q(A, v)$ *(and global sentences).*

Locality Theorem



Fix a structure A, vertex $v \in V(A)$, $q \in \mathbb{N}$.

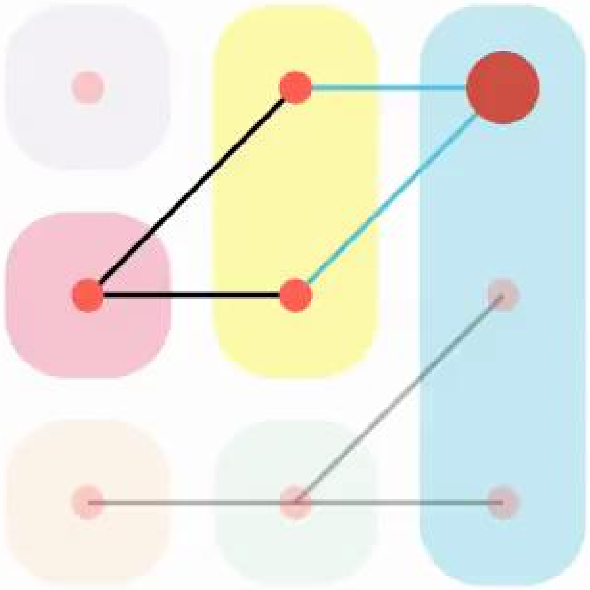
$\mathrm{tp}_q(A, v) :=$ {formulas $\varphi(x)$ of depth q with $A \models \varphi(v)$}

$\mathrm{ltp}_q(A, v) :=$ {"local" formulas $\varphi(x)$ of depth q with $G_t \models \varphi(v)$}
– quantifiers range over r-ball around v in $\mathrm{Gaifman}(A)$, for some $r := r(q)$

Locality Theorem. $\mathrm{tp}_q(A, v)$ *is determined by* $\mathrm{ltp}_q(A, v)$ *(and global sentences).*

Gaifman's Theorem. (1982) $\mathrm{tp}_q(A, v)$ *is determined by* $\mathrm{ltp}_p(A, v)$*, for some* $p > q$*.*

Locality Theorem



Fix a structure A, vertex $v \in V(A)$, $q \in \mathbb{N}$.

$\mathrm{tp}_q(A, v) := \{$formulas $\varphi(x)$ of depth q with $A \models \varphi(v)\}$

$\mathrm{ltp}_q(A, v) := \{$"local" formulas $\varphi(x)$ of depth q with $G_t \models \varphi(v)\}$
– quantifiers range over r-ball around v in $\mathrm{Gaifman}(A)$, for some $r := r(q)$

Locality Theorem. $\mathrm{tp}_q(A, v)$ *is determined by* $\mathrm{ltp}_q(A, v)$ *(and global sentences).*

Gaifman's Theorem. (1982) $\mathrm{tp}_q(A, v)$ *is determined by* $\mathrm{ltp}_p(A, v)$*, for some* $p > q$.

Rank-Preserving Locality Theorem. (Grohe, Kreutzer, Siebertz 2013)
$\mathrm{tp}_q(A, v)$ *is determined by* $\mathrm{ltp}_q(\widehat{A}, v)$*, for a suitable coloring* $\widehat{A}$ *of* A.

Theorem (Dreier and me, 2025). *Every dist-FO formula* $\varphi(\bar{x})$ *is equivalent to a boolean combination of local dist-FO formulas* $\psi(\bar{x})$ *and scatter sentences* σ *of the same distance rank.*

Theorem (Dreier and me, 2025). *Every dist-FO formula $\varphi(\bar{x})$ is equivalent to a boolean combination of local dist-FO formulas $\psi(\bar{x})$ and scatter sentences σ of the same distance rank.*

dist-FO: extension of FO by distance atoms $\mathrm{dist}(x, y) \leq r$, for each radius $r \in \mathbb{N}$.

Theorem (Dreier and me, 2025). *Every dist-FO formula $\varphi(\bar{x})$ is equivalent to a boolean combination of local dist-FO formulas $\psi(\bar{x})$ and scatter sentences σ of the same distance rank.*

dist-FO: extension of FO by distance atoms $\mathrm{dist}(x, y) \leq r$, for each radius $r \in \mathbb{N}$.

local dist-FO: uses only local quantification $\exists y \; (\mathrm{dist}(y, \bar{x}) \leq r) \;\wedge\; \psi(y, \bar{x})$.

Theorem (Dreier and me, 2025). *Every dist-FO formula* $\varphi(\bar{x})$ *is equivalent to a boolean combination of local dist-FO formulas* $\psi(\bar{x})$ *and scatter sentences* σ *of the same distance rank.*

dist-FO: extension of FO by distance atoms $\mathrm{dist}(x, y) \leq r$, for each radius $r \in \mathbb{N}$.

local dist-FO: uses only local quantification $\exists y\ (\mathrm{dist}(y, \bar{x}) \leq r)\ \wedge\ \psi(y, \bar{x})$.

distance rank: imposes restrictions on r in terms of quantifier depth (deeper $\rightarrow$ shorter).

Theorem (Dreier and me, 2025). *Every dist-FO formula $\varphi(\bar{x})$ is equivalent to a boolean combination of local dist-FO formulas $\psi(\bar{x})$ and scatter sentences σ of the same distance rank.*

dist-FO: extension of FO by distance atoms $\mathrm{dist}(x, y) \leq r$, for each radius $r \in \mathbb{N}$.

local dist-FO: uses only local quantification $\exists y \; (\mathrm{dist}(y, \bar{x}) \leq r) \; \wedge \; \psi(y, \bar{x})$.

distance rank: imposes restrictions on r in terms of quantifier depth (deeper $\rightarrow$ shorter).

scatter sentences: ask about size of any inclusion-wise maximal r-scattered set of elements satisfying $\alpha(x)$.

Closure properties

$\varphi(x, y)$ – first-order formula, G – graph. Define a graph $\varphi(G)$ with:

Closure properties

$\varphi(x, y)$ – first-order formula, G – graph. Define a graph $\varphi(G)$ with:

- vertices $V(G)$

Closure properties

$\varphi(x, y)$ – first-order formula, G – graph. Define a graph $\varphi(G)$ with:

- vertices $V(G)$
- edges $\{uv \mid G \models \varphi(u, v)\}$.

Closure properties

$\varphi(x, y)$ – first-order formula, G – graph. Define a graph $\varphi(G)$ with:

- vertices $V(G)$
- edges $\{uv \mid G \models \varphi(u, v)\}$.

Example. If $\varphi(x, y) = \exists z.E(x, z) \wedge E(y, z)$ then $\varphi(G) = G^2$.

Closure properties

$\varphi(x, y)$ – first-order formula, G – graph. Define a graph $\varphi(G)$ with:

- vertices $V(G)$
- edges $\{uv \mid G \models \varphi(u, v)\}$.

Example. If $\varphi(x, y) = \exists z.E(x, z) \wedge E(y, z)$ then $\varphi(G) = G^2$.

Theorem. C has bounded merge-width $\Rightarrow \varphi(C) := \{\varphi(G) \mid G \in C\}$ has bounded merge-width.

Example. If $\varphi(x, y) = \exists z. E(x, z) \wedge E(y, z)$ then $\varphi(G) = G^2$.

Theorem. C has bounded merge-width $\Rightarrow \varphi(C) := \{\varphi(G) \mid G \in C\}$ has bounded merge-width.

Proof sketch. Fix $\varphi(x, y)$ of depth q.

Example. If $\varphi(x, y) = \exists z. E(x, z) \wedge E(y, z)$ then $\varphi(G) = G^2$.

Theorem. C has bounded merge-width $\Rightarrow \varphi(C) := \{\varphi(G) \mid G \in C\}$ has bounded merge-width.

Proof sketch. Fix $\varphi(x, y)$ of depth q.

- partition vertices $v \in V$ by $\mathrm{ltp}_q(G_t, v)$.

Example. If $\varphi(x, y) = \exists z. E(x, z) \wedge E(y, z)$ then $\varphi(G) = G^2$.

Theorem. C has bounded merge-width $\Rightarrow \varphi(C) := \{\varphi(G) \mid G \in C\}$ has bounded merge-width.

Proof sketch. Fix $\varphi(x, y)$ of depth q.

- partition vertices $v \in V$ by $\mathrm{ltp}_q(G_t, v)$.
- $\mathrm{ltp}_q(G_t, vw)$ determines $\mathrm{ltp}_q(G_{t+1}, vw)$, for all $t \in \{1, \ldots, m\}$, $v, w \in V$.

Example. If $\varphi(x, y) = \exists z. E(x, z) \wedge E(y, z)$ then $\varphi(G) = G^2$.

Theorem. C has bounded merge-width $\Rightarrow \varphi(C) := \{\varphi(G) \mid G \in C\}$ has bounded merge-width.

Proof sketch. Fix $\varphi(x, y)$ of depth q.

- partition vertices $v \in V$ by $\mathrm{ltp}_q(G_t, v)$.
- $\mathrm{ltp}_q(G_t, vw)$ determines $\mathrm{ltp}_q(G_{t+1}, vw)$, for all $t \in \{1, \ldots, m\}$, $v, w \in V$.
- $\mathrm{ltp}_q(G_t, vw)$ determines if $G \models \varphi(v, w)$.

Example. If $\varphi(x,y) = \exists z. E(x,z) \wedge E(y,z)$ then $\varphi(G) = G^2$.

Theorem. C has bounded merge-width $\Rightarrow \varphi(C) := \{\varphi(G) \mid G \in C\}$ has bounded merge-width.

Proof sketch. Fix $\varphi(x,y)$ of depth q.

- partition vertices $v \in V$ by $\mathrm{ltp}_q(G_t, v)$.
- $\mathrm{ltp}_q(G_t, vw)$ determines $\mathrm{ltp}_q(G_{t+1}, vw)$, for all $t \in \{1, \ldots, m\}$, $v, w \in V$.
- $\mathrm{ltp}_q(G_t, vw)$ determines if $G \models \varphi(v,w)$.
- $\mathrm{ltp}_q(G_t, vw)$ depends only on $\mathrm{ltp}_q(G_t, v)$ and $\mathrm{ltp}_q(G_t, w)$ if $\mathrm{dist}_{G_t}(v,w) > r$

Example. If $\varphi(x, y) = \exists z. E(x, z) \wedge E(y, z)$ then $\varphi(G) = G^2$.

Theorem. C has bounded merge-width $\Rightarrow \varphi(C) := \{\varphi(G) \mid G \in C\}$ has bounded merge-width.

Proof sketch. Fix $\varphi(x, y)$ of depth q.

- partition vertices $v \in V$ by $\mathrm{ltp}_q(G_t, v)$.
- $\mathrm{ltp}_q(G_t, vw)$ determines $\mathrm{ltp}_q(G_{t+1}, vw)$, for all $t \in \{1, \ldots, m\}, v, w \in V$.
- $\mathrm{ltp}_q(G_t, vw)$ determines if $G \models \varphi(v, w)$.
- $\mathrm{ltp}_q(G_t, vw)$ depends only on $\mathrm{ltp}_q(G_t, v)$ and $\mathrm{ltp}_q(G_t, w)$ if $\mathrm{dist}_{G_t}(v, w) > r$
- at time t resolve pairs v, w with $\mathrm{dist}_{G_t}(v, w) \leq r$. ■

Example. If $\varphi(x, y) = \exists z. E(x, z) \wedge E(y, z)$ then $\varphi(G) = G^2$.

Theorem. C has bounded merge-width $\Rightarrow \varphi(C) := \{\varphi(G) \mid G \in C\}$ has bounded merge-width.

Corollary. C has bounded merge-width $\Rightarrow C$ is *monadically dependent*: $\varphi(C) \neq \{\textit{Graphs}\}$, for every formula $\varphi(x, y)$.

Theorem. C has bounded merge-width $\Rightarrow$ C has bounded flip-width.

Theorem. C has bounded merge-width $\Rightarrow C$ has bounded flip-width.

Proof sketch. A construction sequence of radius $2r$ yields a strategy for the pursuer in the Flipper game of radius r. ■

Theorem. C has bounded merge-width $\Rightarrow$ C has bounded flip-width.

Proof sketch. A construction sequence of radius $2r$ yields a strategy for the pursuer in the Flipper game of radius r. ■

Corollary. C has bounded expansion $\Leftrightarrow$ C has bounded merge-width and is $K_{t,t}$-free.

Theorem. C has bounded merge-width $\Rightarrow$ C has bounded flip-width.

Proof sketch. A construction sequence of radius $2r$ yields a strategy for the pursuer in the Flipper game of radius r. ■

Corollary. C has bounded expansion $\Leftrightarrow$ C has bounded merge-width and is $K_{t,t}$-free.

Proof. This holds for classes of bounded flip-width. ■

Theorem. C has bounded merge-width $\Rightarrow$ C has bounded flip-width.

Proof sketch. A construction sequence of radius $2r$ yields a strategy for the pursuer in the Flipper game of radius r. ■

Corollary. C has bounded expansion $\Leftrightarrow$ C has bounded merge-width and is $K_{t,t}$-free.

Proof. This holds for classes of bounded flip-width. ■

Conjecture. The following are equivalent:

- bounded merge-width,

Theorem. C has bounded merge-width $\Rightarrow$ C has bounded flip-width.

Proof sketch. A construction sequence of radius $2r$ yields a strategy for the pursuer in the Flipper game of radius r. ■

Corollary. C has bounded expansion $\Leftrightarrow$ C has bounded merge-width and is $K_{t,t}$-free.

Proof. This holds for classes of bounded flip-width. ■

Conjecture. The following are equivalent:

- bounded merge-width,
- bounded flip-width,

Theorem. C has bounded merge-width $\Rightarrow C$ has bounded flip-width.

Proof sketch. A construction sequence of radius $2r$ yields a strategy for the pursuer in the Flipper game of radius r. ■

Corollary. C has bounded expansion $\Leftrightarrow C$ has bounded merge-width and is $K_{t,t}$-free.

Proof. This holds for classes of bounded flip-width. ■

Conjecture. The following are equivalent:

- bounded merge-width,
- bounded flip-width,
- every $K_{t,t}$-free transduction has bounded expansion.

Almost bounded merge-width

Almost bounded merge-width

C has *almost bounded merge-width* if for all $r \in \mathbb{N}$,

$$\mathrm{mw}_r(G) = |V(G)|^{o(1)} \quad \text{for all } G \in C.$$

Almost bounded merge-width

C has *almost bounded merge-width* if for all $r \in \mathbb{N}$,

$$\mathrm{mw}_r(G) = |V(G)|^{o(1)} \quad \text{for all } G \in C.$$

Theorem. Hereditary, almost bounded merge-width $\Rightarrow$ monadically dependent.

Almost bounded merge-width

C has *almost bounded merge-width* if for all $r \in \mathbb{N}$,

$$\mathrm{mw}_r(G) = |V(G)|^{o(1)} \quad \text{for all } G \in C.$$

Theorem. Hereditary, almost bounded merge-width $\Rightarrow$ monadically dependent.

Proof.
Almost bounded merge-width $\Rightarrow$ almost bounded flip-width:
$\forall r$ pursuer wins the Flipper game of radius r using flips of size $n^{o(1)}$.

$\mathrm{mw}_r(G) = |V(G)|^{o(1)}$ for all $G \in C$.

Theorem. Hereditary, almost bounded merge-width ⇒ monadically dependent.

Proof.
Almost bounded merge-width ⇒ almost bounded flip-width:
$\forall r$ pursuer wins the Flipper game of radius r using flips of size $n^{o(1)}$.

not monadically dependent ⇒
$\exists r$ pursuer needs flips of size $\Omega(n^{\frac{1}{2}})$ [Dreier, Mählmann, me, 2024] ■

$$\mathrm{mw}_r(G) = |V(G)|^{o(1)} \quad \text{for all } G \in C.$$

Theorem. Hereditary, almost bounded merge-width $\Rightarrow$ monadically dependent.

Proof.
Almost bounded merge-width $\Rightarrow$ almost bounded flip-width:
$\forall r$ pursuer wins the Flipper game of radius r using flips of size $n^{o(1)}$.

not monadically dependent $\Rightarrow$
$\exists r$ pursuer needs flips of size $\Omega(n^{\frac{1}{2}})$ [Dreier, Mählmann, me, 2024] ∎

Conjecture. The following are equivalent:

- almost bounded merge-width,

$$\mathrm{mw}_r(G) = |V(G)|^{o(1)} \quad \text{for all } G \in C.$$

Theorem. Hereditary, almost bounded merge-width $\Rightarrow$ monadically dependent.

Proof.

Almost bounded merge-width $\Rightarrow$ almost bounded flip-width:
$\forall r$ pursuer wins the Flipper game of radius r using flips of size $n^{o(1)}$.

not monadically dependent $\Rightarrow$
$\exists r$ pursuer needs flips of size $\Omega(n^{\frac{1}{2}})$ [Dreier, Mählmann, me, 2024] ■

Conjecture. The following are equivalent:

- almost bounded merge-width,
- almost bounded flip-width,

$$\mathrm{mw}_r(G) = |V(G)|^{o(1)} \quad \text{for all } G \in C.$$

Theorem. Hereditary, almost bounded merge-width $\Rightarrow$ monadically dependent.

Proof.
Almost bounded merge-width $\Rightarrow$ almost bounded flip-width:
$\forall r$ pursuer wins the Flipper game of radius r using flips of size $n^{o(1)}$.

not monadically dependent $\Rightarrow$
$\exists r$ pursuer needs flips of size $\Omega(n^{\frac{1}{2}})$ [Dreier, Mählmann, me, 2024] ∎

Conjecture. The following are equivalent:

- almost bounded merge-width,
- almost bounded flip-width,
- monadically dependent.

$$\mathrm{mw}_r(G) = |V(G)|^{o(1)} \quad \text{for all } G \in C.$$

Theorem. Hereditary, almost bounded merge-width ⇒ monadically dependent.

Proof.
Almost bounded merge-width ⇒ almost bounded flip-width:
$\forall r$ pursuer wins the Flipper game of radius r using flips of size $n^{o(1)}$.

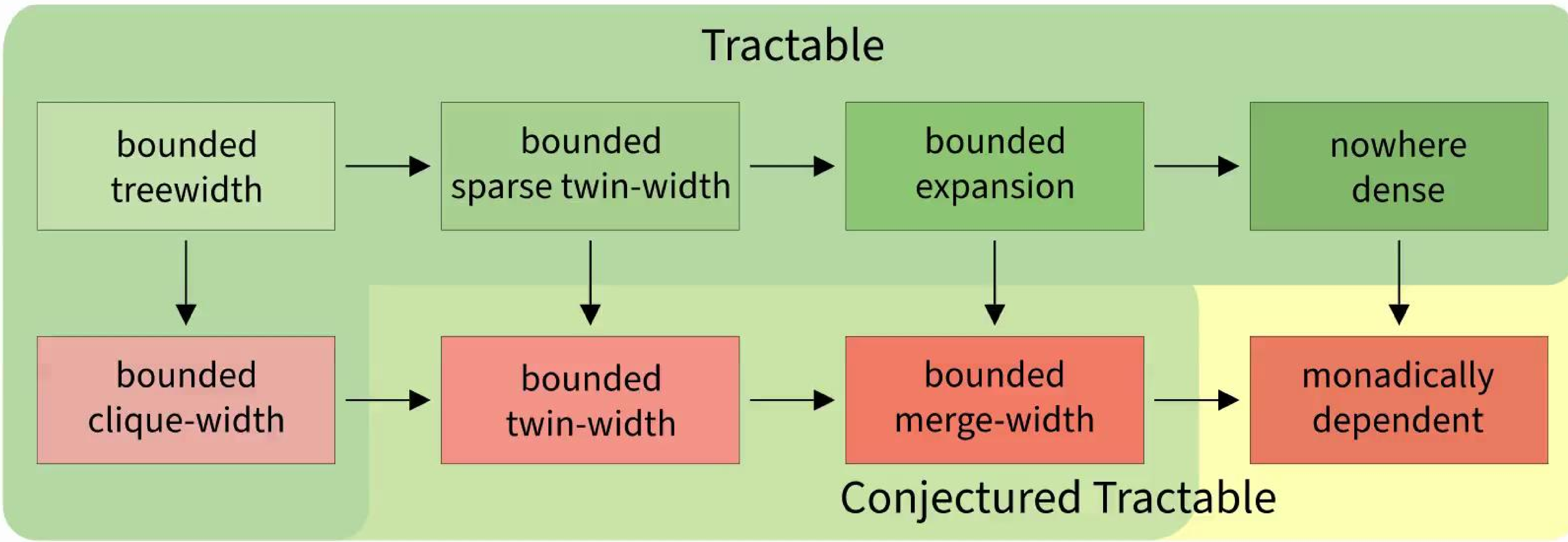
not monadically dependent ⇒
$\exists r$ pursuer needs flips of size $\Omega(n^{\frac{1}{2}})$ [Dreier, Mählmann, me, 2024] ■

Conjecture. The following are equivalent:

- almost bounded merge-width,
- almost bounded flip-width,
- monadically dependent.

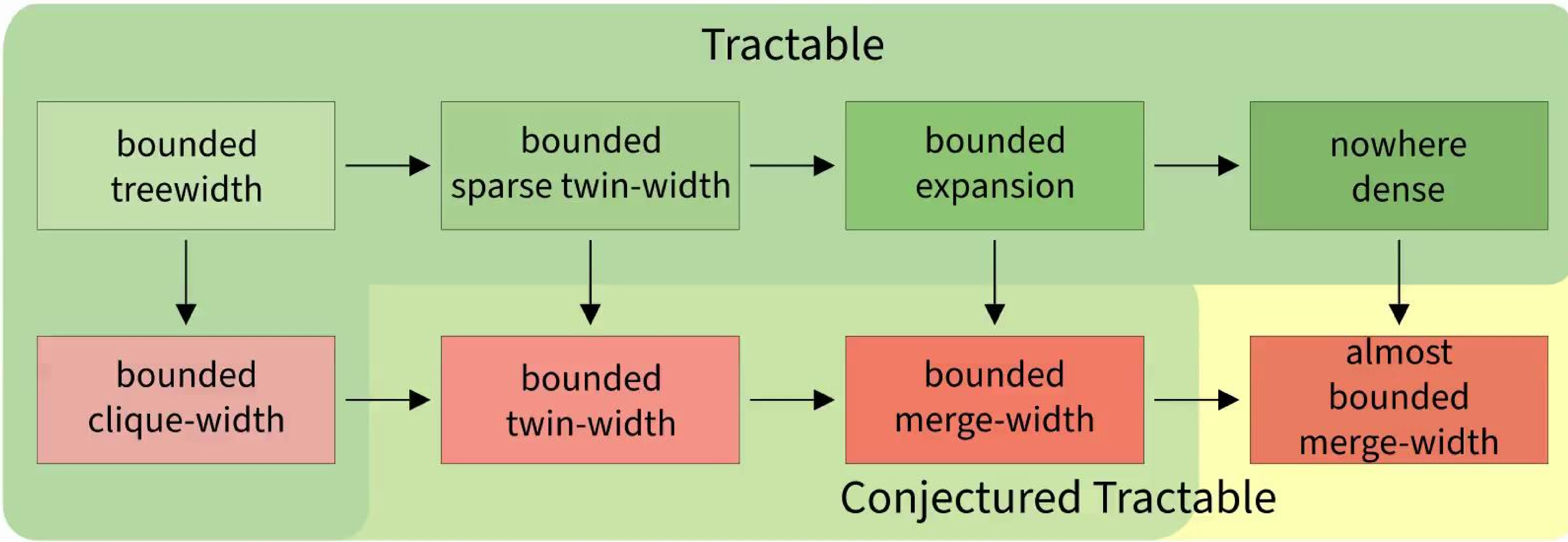
Conjecture. The following are equivalent:

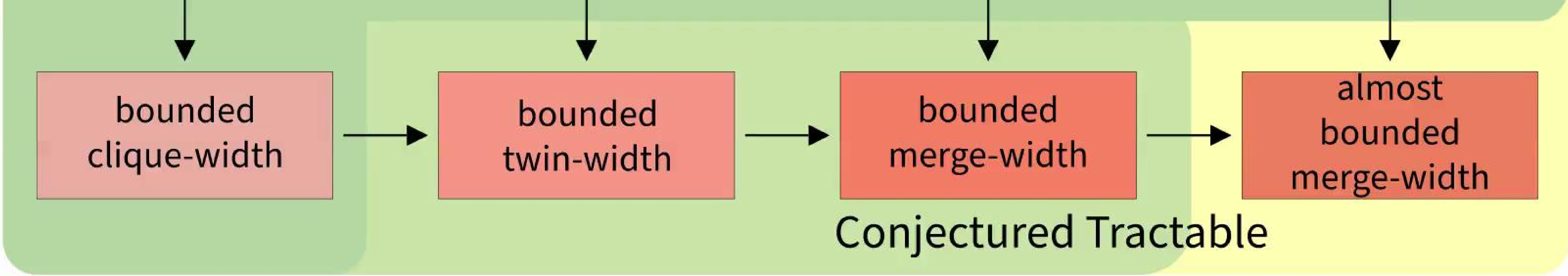
- almost bounded merge-width,
- almost bounded flip-width,
- monadically dependent.



Conjecture. The following are equivalent:

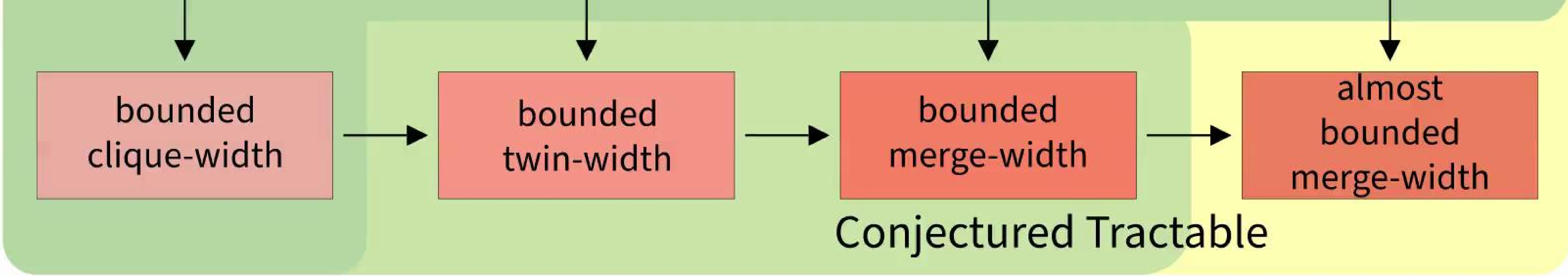
- almost bounded merge-width,
- almost bounded flip-width,
- monadically dependent.





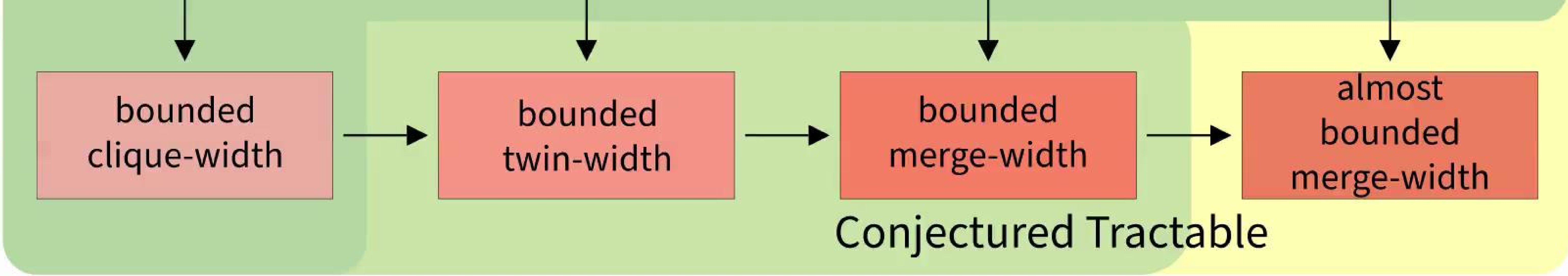
Questions:

- approximating merge-width (and finding a witnessing construction sequence),



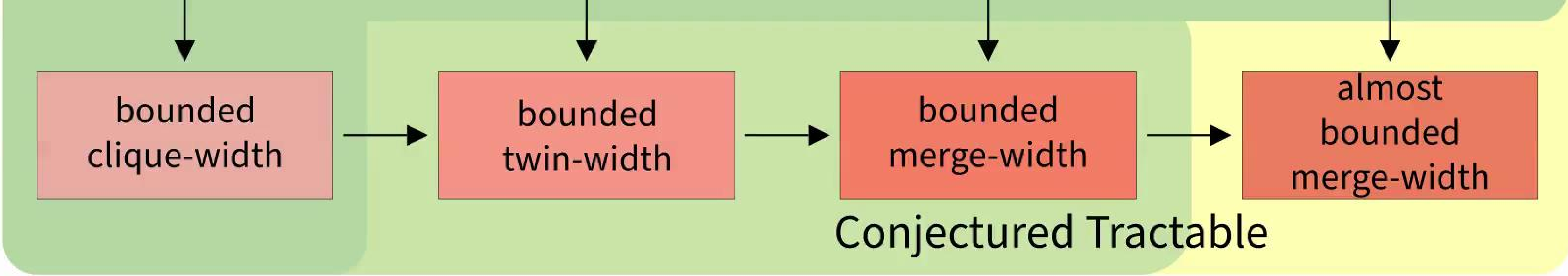
Questions:

- approximating merge-width (and finding a witnessing construction sequence),
- almost bounded merge-width $\Leftrightarrow$ monadically dependent?



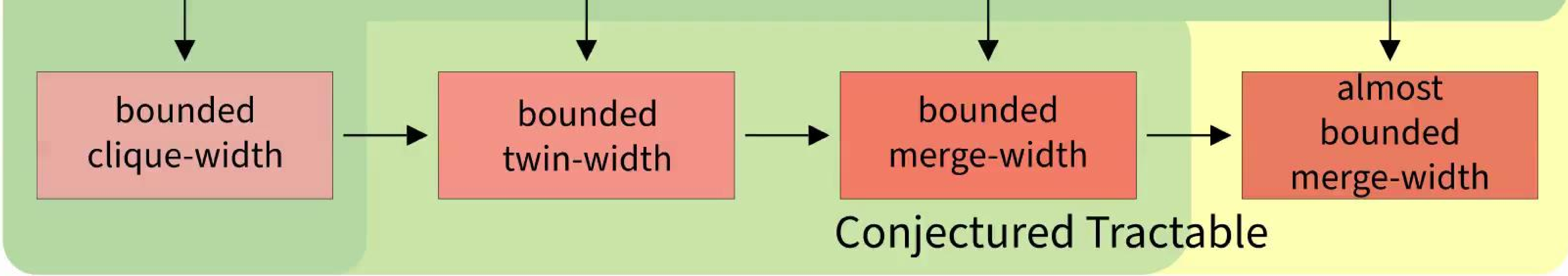
Questions:

- approximating merge-width (and finding a witnessing construction sequence),
- almost bounded merge-width ⇔ monadically dependent?
- bounded merge-width ⇔ bounded flip-width?



Questions:

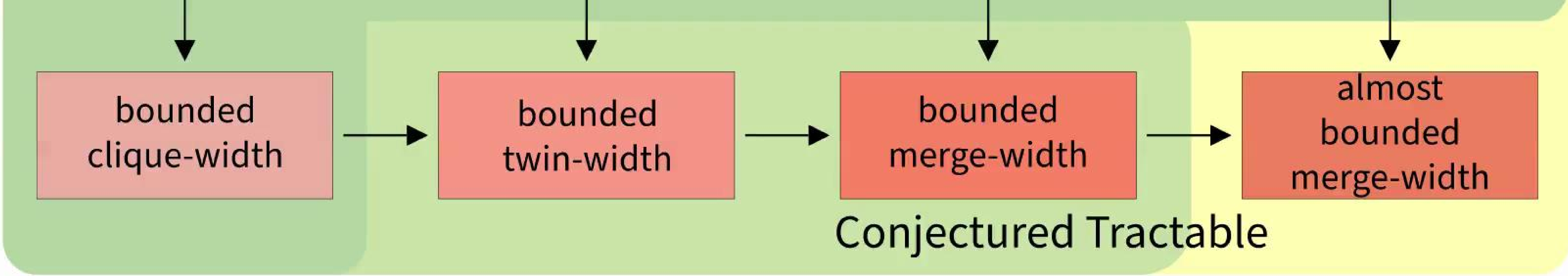
- approximating merge-width (and finding a witnessing construction sequence),
- almost bounded merge-width ⇔ monadically dependent?
- bounded merge-width ⇔ bounded flip-width?
- unbounded merge-width ⇔ transduces a $K_{t,t}$-free class of unbounded min. deg?



Questions:

- approximating merge-width (and finding a witnessing construction sequence),
- almost bounded merge-width ⇔ monadically dependent?
- bounded merge-width ⇔ bounded flip-width?
- unbounded merge-width ⇔ transduces a $K_{t,t}$-free class of unbounded min. deg?
- bounded merge-width ⇒ χ-bounded – **done:** [Bonamy, Geniet '25].

Polynomial-χ-boundedness?



Questions:

- approximating merge-width (and finding a witnessing construction sequence),
- almost bounded merge-width ⇔ monadically dependent?
- bounded merge-width ⇔ bounded flip-width?
- unbounded merge-width ⇔ transduces a $K_{t,t}$-free class of unbounded min. deg?
- bounded merge-width ⇒ χ-bounded – **done:** [Bonamy, Geniet '25].

Polynomial-χ-boundedness?

- other algorithmic problems on classes of bounded merge-width?