Approximate First-Order Counting Queries on Sparse and Dense Graphs

Jan Dreier
Vienna University of Technology

joint work with Peter Rossmanith

March 16, 2021

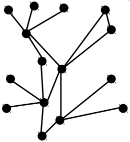
Algorithmic Meta-Theorems

*"All problems expressible in a certain logic
can be solved efficiently on certain graphs"*

Algorithmic Meta-Theorems

"All problems expressible in a certain logic can be solved efficiently on certain graphs"

MSO on treewidth

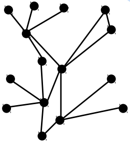


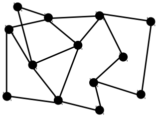
Algorithmic Meta-Theorems

"All problems expressible in a certain logic can be solved efficiently on certain graphs"

MSO on treewidth

FO on sparse graphs



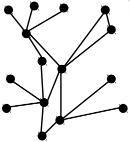


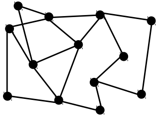
Algorithmic Meta-Theorems

"All problems expressible in a certain logic
can be solved efficiently on certain graphs"

MSO on treewidth

FO on sparse graphs





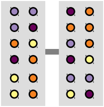
FO({>0}) for ***approximation*** on sparse graphs

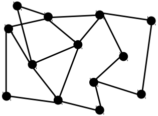
Algorithmic Meta-Theorems

"All problems expressible in a certain logic can be solved efficiently on certain graphs"

MSO on cliquewidth

FO on sparse graphs





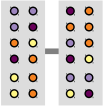
FO({>0}) for ***approximation*** on sparse graphs

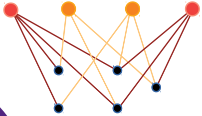
Algorithmic Meta-Theorems

"All problems expressible in a certain logic can be solved efficiently on certain graphs"

MSO on cliquewidth

?





?

Logic

Many problems can be expressed in first-order (FO) logic.

Logic

Many problems can be expressed in first-order (FO) logic.

- independent set of size k:

$$\exists x_1 \ldots \exists x_k \bigwedge_{i,j} x_i \not\sim x_j \wedge x_i \neq x_j$$

Logic

Many problems can be expressed in first-order (FO) logic.

- independent set of size k:

$$\exists x_1 \ldots \exists x_k \bigwedge_{i,j} x_i \not\sim x_j \wedge x_i \neq x_j$$

- dominating set of size k:

$$\exists x_1 \ldots \exists x_k \forall y \bigvee_{i} y \sim x_i \vee y = x_i$$

Logic

Many problems can be expressed in first-order (FO) logic.

- independent set of size k:

$$\exists x_1 \ldots \exists x_k \bigwedge_{i,j} x_i \not\sim x_j \wedge x_i \neq x_j$$

- dominating set of size k:

$$\exists x_1 \ldots \exists x_k \forall y \bigvee_{i} y \sim x_i \vee y = x_i$$

- basic database queries

Logic

Many problems can be expressed in first-order (FO) logic.

- independent set of size k:

$$\exists x_1 \ldots \exists x_k \bigwedge_{i,j} x_i \not\sim x_j \wedge x_i \neq x_j$$

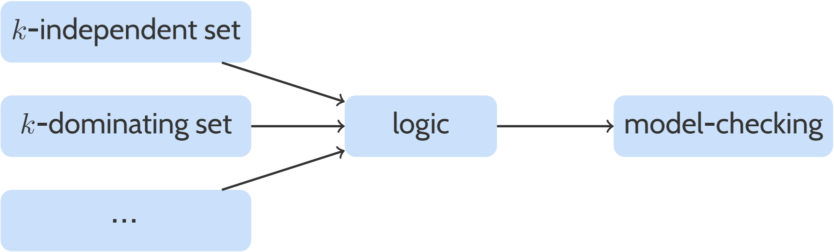
- dominating set of size k:

$$\exists x_1 \ldots \exists x_k \forall y \bigvee_{i} y \sim x_i \vee y = x_i$$

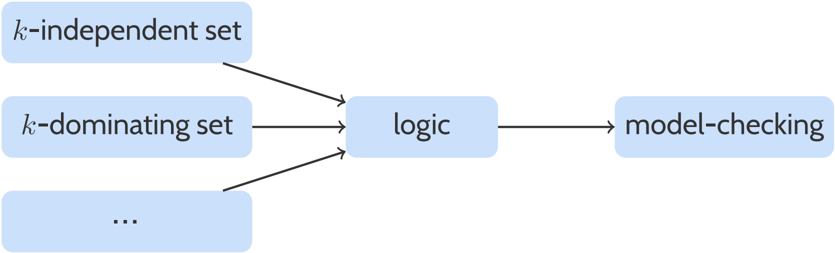
- basic database queries

Best algorithms on general graphs: $n^{O(k)}$

Model-Checking



Model-Checking



MC($\mathcal{G}$, L)

Input: A graph $G \in \mathcal{G}$ and a sentence $\varphi \in \mathrm{L}$

Parameter: $|\varphi|$

Problem: Is φ true in G?

Goal: linear FPT run time $f(|\varphi|)n$

Model-Checking

k-independent set

k-dominating set

...

$\to$ logic $\to$ model-checking

$\varphi = \exists x_1 \ldots \exists x_k [\ldots]$

length depends on k

$f(|\varphi|)n$

algorithm

MC($\mathcal{G}$, L)

Input:	A graph $G \in \mathcal{G}$ and a sentence $\varphi \in \mathrm{L}$
Parameter:	$\lvert\varphi\rvert$
Problem:	Is φ true in G?

Goal: linear FPT run time $f(|\varphi|)n$

Some Sparse Graph Classes

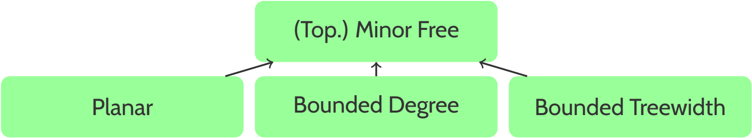
Planar

Bounded Degree

Bounded Treewidth

If $\mathcal{G}$ has bounded treewidth then MC($\mathcal{G}$, MSO) $\in$ FPT.

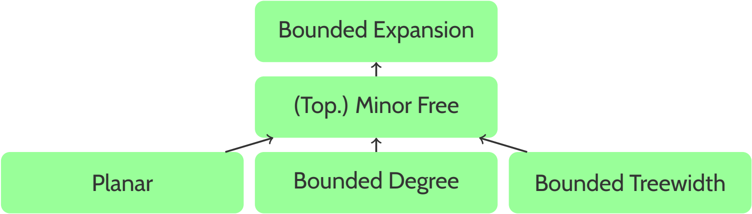
[Courcelle 1990]



If $\mathcal{G}$ has bounded treewidth then MC($\mathcal{G}$, MSO) $\in$ FPT.

[Courcelle 1990]

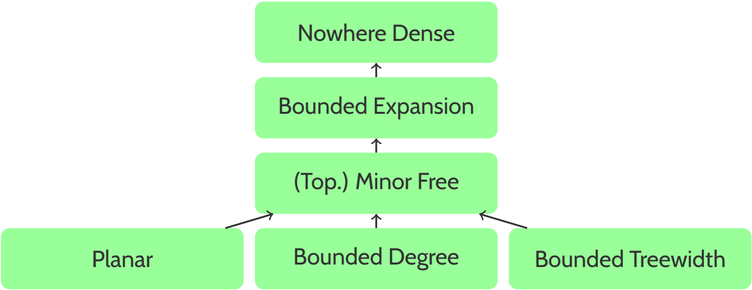
Some Sparse Graph Classes



If $\mathcal{G}$ has bounded treewidth then MC($\mathcal{G}$, MSO) $\in$ FPT.

[Courcelle 1990]

Some Sparse Graph Classes



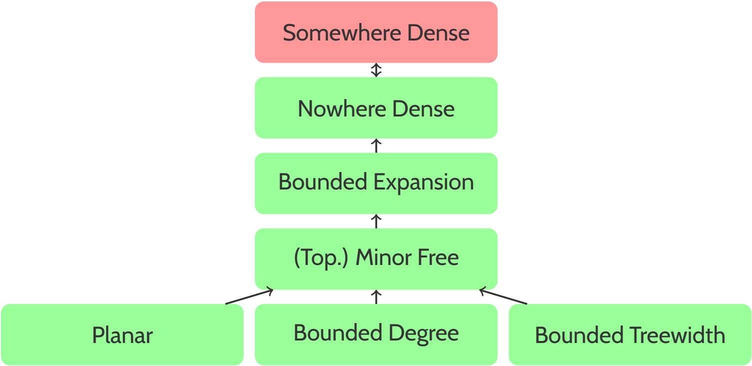
If $\mathcal{G}$ has bounded treewidth then MC($\mathcal{G}$, MSO) $\in$ FPT.

[Courcelle 1990]

If $\mathcal{G}$ is nowhere dense then MC($\mathcal{G}$, FO) $\in$ FPT.

[Grohe, Kreutzer, Sieberz 2011]

Some Sparse Graph Classes



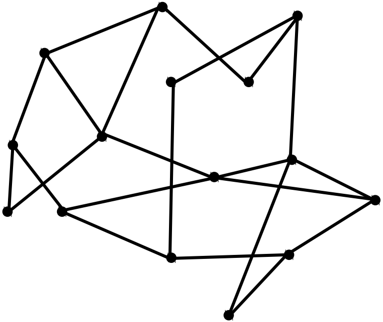
If $\mathcal{G}$ has bounded treewidth then MC($\mathcal{G}$, MSO) $\in$ FPT.

[Courcelle 1990]

If $\mathcal{G}$ is nowhere dense then MC($\mathcal{G}$, FO) $\in$ FPT.

[Grohe, Kreutzer, Sieberz 2011]

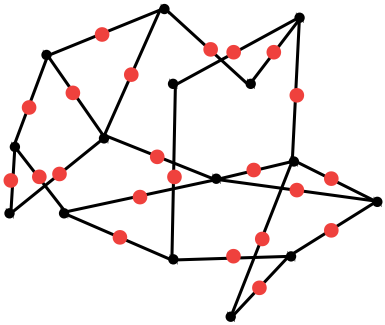
Bounded Expansion — Minors



$$\frac{|E|}{|V|} \leq c$$

for every graph
in the graph class.

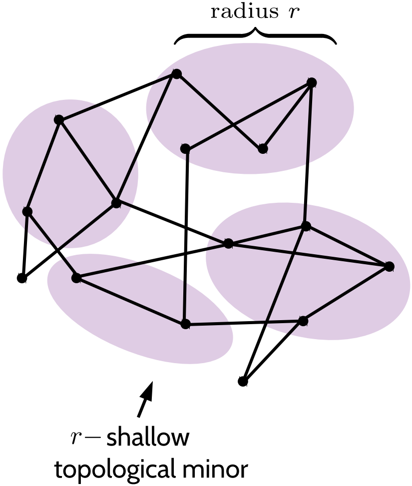
Bounded Expansion — Minors



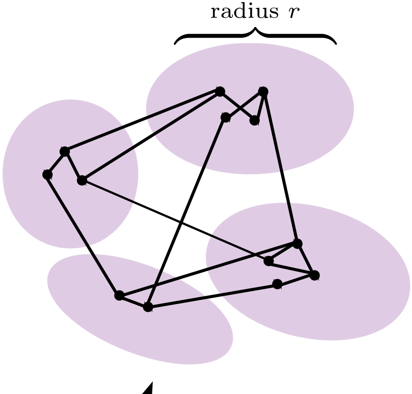
$$\frac{|E|}{|V|} \leq 2$$

for every graph
in the graph class.

Bounded Expansion — Minors

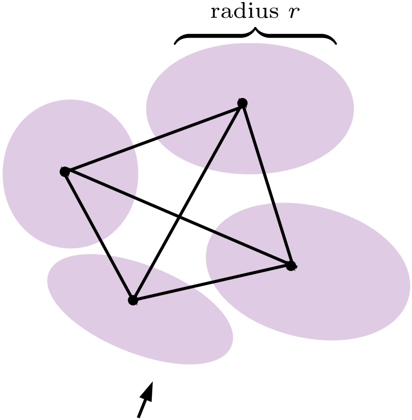


Bounded Expansion — Minors



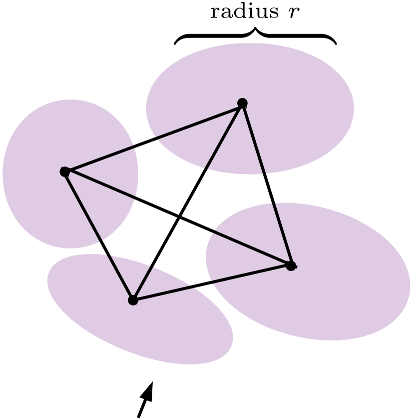
$r-$shallow
topological minor

Bounded Expansion — Minors



$r-$shallow
topological minor

Bounded Expansion — Minors



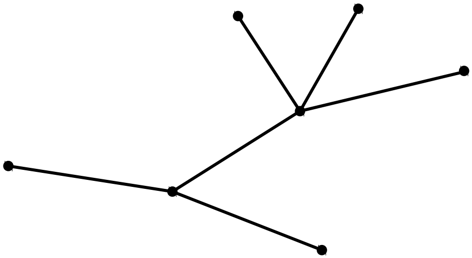
$r-$ shallow
topological minor

$$\frac{|E|}{|V|} \leq f(r)$$

for every r-shallow minor of every graph in the graph class.

Bounded Expansion — Augmentations

Possible to do these augmentations such that after r augmentations, outdegree at most $f(r)$.

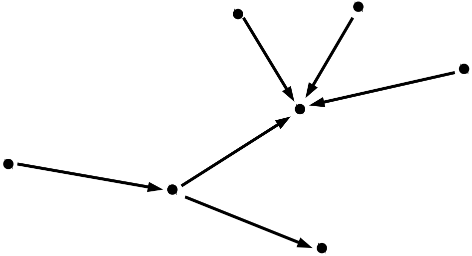


orient edges

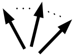
Bounded Expansion — Augmentations

Possible to do these augmentations such that after r augmentations, outdegree at most $f(r)$.

0$^{\text{th}}$ Augmentation



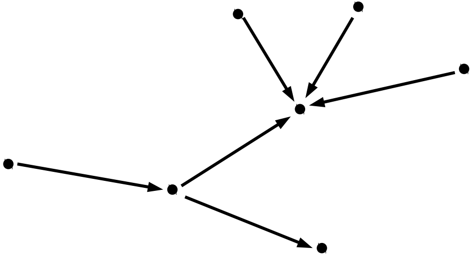
orient edges

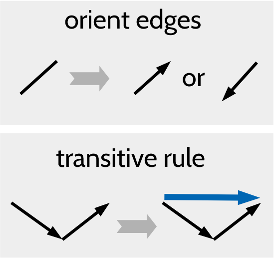
max outdegree  = 2

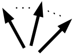
Bounded Expansion — Augmentations

Possible to do these augmentations such that after r augmentations, outdegree at most $f(r)$.

1st Augmentation



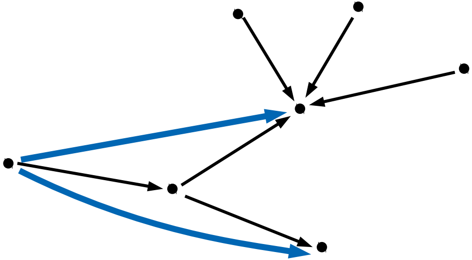


max outdegree  = 2

Bounded Expansion — Augmentations

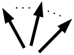
Possible to do these augmentations such that after r augmentations, outdegree at most $f(r)$.

1st Augmentation



orient edges

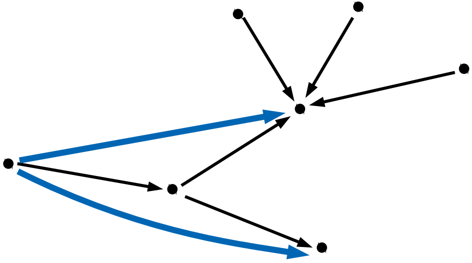
transitive rule

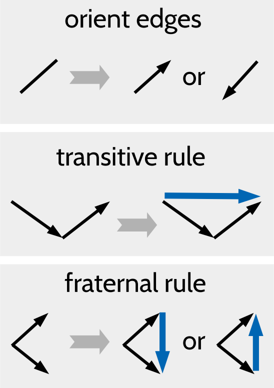
max outdegree  = 3

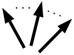
Bounded Expansion — Augmentations

Possible to do these augmentations such that after r augmentations, outdegree at most $f(r)$.

1st Augmentation



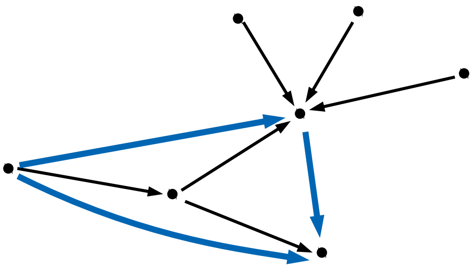


max outdegree  = 3

Bounded Expansion — Augmentations

Possible to do these augmentations such that after r augmentations, outdegree at most $f(r)$.

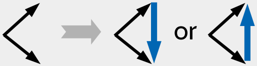
1st Augmentation

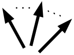


orient edges

transitive rule

fraternal rule

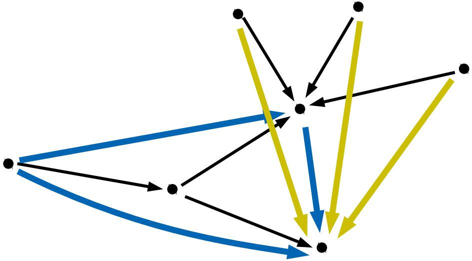


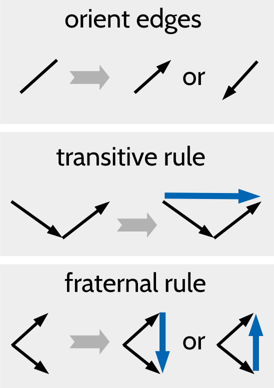
max outdegree  = 3

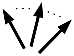
Bounded Expansion — Augmentations

Possible to do these augmentations such that after r augmentations, outdegree at most $f(r)$.

2nd Augmentation





max outdegree  = 3

Approximate Counting Queries

Counting Problems

Partial Dominating Set

Input: A graph G and $k, m \in \mathbf{N}$

Parameter: k

Problem: Are there k vertices dominating m vertices?

Partial Dominating Set

Input:	A graph G and $k, m \in \mathbf{N}$
Parameter:	k
Problem:	Are there k vertices dominating m vertices?

Cannot be expressed in first-order logic (requires $\exists y_1 \ldots \exists y_m$).

Counting Problems

PARTIAL DOMINATING SET	
Input:	A graph G and $k, m \in \mathbf{N}$
Parameter:	k
Problem:	Are there k vertices dominating m vertices?

Cannot be expressed in first-order logic (requires $\exists y_1 \ldots \exists y_m$).

Can be solved on H-minor free graphs in time $(g(H)k)^k n^{O(1)}$.

[Amini, Fomin, Saurabh, 2008]

Can be solved on apex-minor-free graphs in time $2^{\sqrt{k}} n^{O(1)}$.

[Fomin, Lokshtanov, Raman, Saurabh, 2011]

Is W[1]-hard for 2-degenerate graphs.

[Golovach, Villanger 2008]

PARTIAL DOMINATING SET

Input:	A graph G and $k, m \in \mathbf{N}$
Parameter:	k
Problem:	Are there k vertices dominating m vertices?

FO($\{>0\}$) = FO + “there are at least/most $m \in \mathbf{N}$ elements”

Counting Problems

Partial Dominating Set	
Input:	A graph G and $k, m \in \mathbf{N}$
Parameter:	k
Problem:	Are there k vertices dominating m vertices?

FO($\{>0\}$) = FO + "there are at least/most $m \in \mathbf{N}$ elements"

$$\exists x_1 \ldots \exists x_k \, \#y \left(\bigvee_i y \sim x_i \vee y = x_i \right) \geq m$$

Counting Problems

Partial Dominating Set	
Input:	A graph G and $k, m \in \mathbf{N}$
Parameter:	k
Problem:	Are there k vertices dominating m vertices?

FO($\{>0\}$) = FO + "there are at least/most $m \in \mathbf{N}$ elements"

$$\exists x_1 \ldots \exists x_k \, \#y \left(\bigvee_i y \sim x_i \vee y = x_i \right) \geq m$$

Length of formula depends only on k (and not on m)

Counting Logic

Definition of FO$(\{>0\})$

built recursively using

- the rules of FO
- $\#y\ \varphi \geq m$ for every $m \in \mathbf{N}$ and FO$(\{>0\})$ formula φ

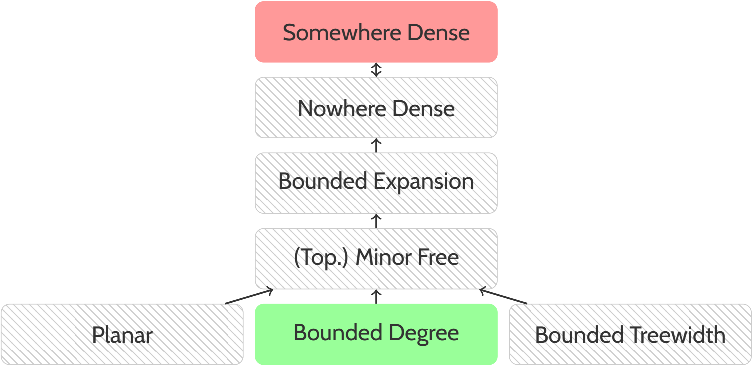
Example 1: Partial Dominating Set

$$\exists x_1 \ldots \exists x_k\, \#y\ \Big(\bigvee_i y \sim x_i \wedge y = x_i\Big) \geq m$$

Example 2: h-Index

$$\#\text{mypaper}\ \Big(\#\text{otherpaper cite}(\text{otherpaper}, \text{mypaper}) \geq h\Big) \geq h$$

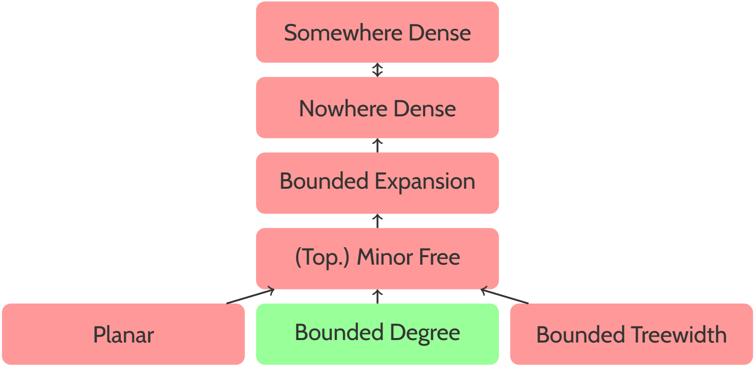
Good News



If $\mathcal{G}$ has bounded degree then MC($\mathcal{G}$, FOC) $\in$ FPT.

[Kuske, Schweikardt 2017]

Bad News



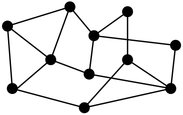
If $\mathcal{G}$ has bounded degree then MC($\mathcal{G}$, FOC) $\in$ FPT.

[Kuske, Schweikardt 2017]

MC($\mathcal{G}$, FO($\{>0\}$)) is AW[$*$]-hard on trees.

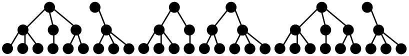
similar to [Grohe, Schweikardt 2018]

Bad News



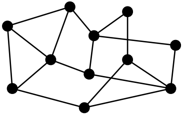
contains k-clique

$\Leftrightarrow$



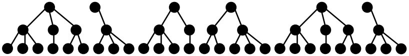
satisfies FO({>0}) formula

Bad News

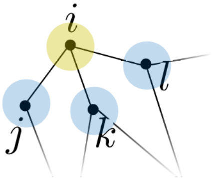


contains k-clique

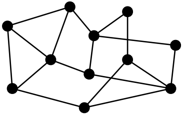
$\Longleftrightarrow$



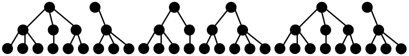
satisfies FO({>0}) formula



Bad News

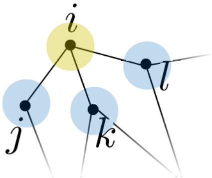


$\Leftrightarrow$

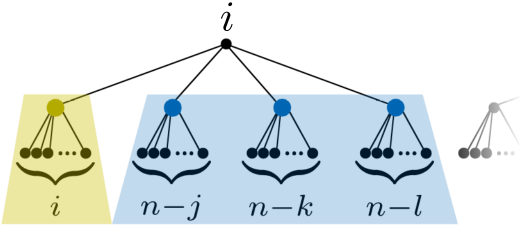


contains k-clique

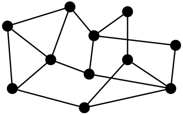
satisfies FO({>0}) formula



$\Leftrightarrow$

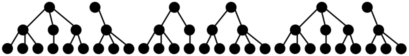


Bad News

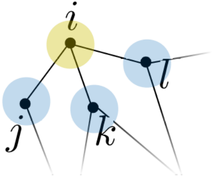


contains k-clique

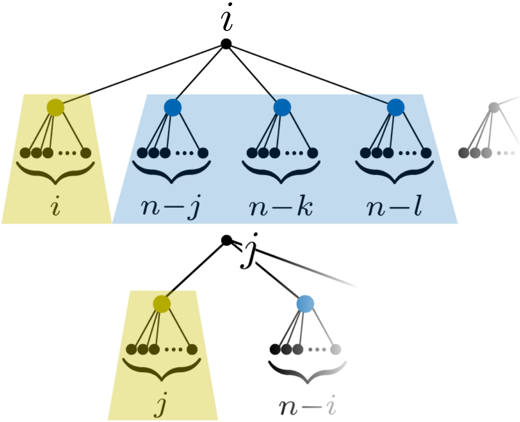
$\Leftrightarrow$



satisfies FO({>0}) formula

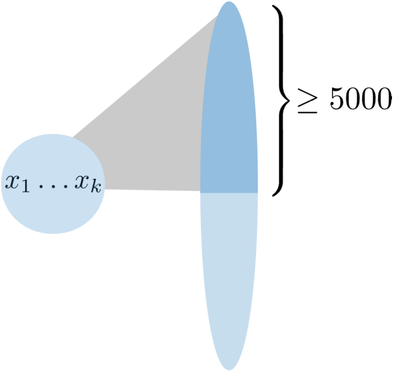


$\Leftrightarrow$



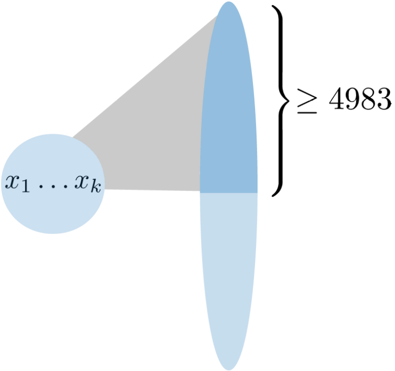
Stability

Are there k vertices dominating at least $m = 5000$ vertices?



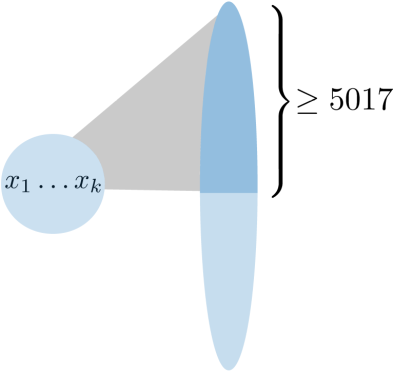
Stability

Are there k vertices dominating at least $m = 4983$ vertices?



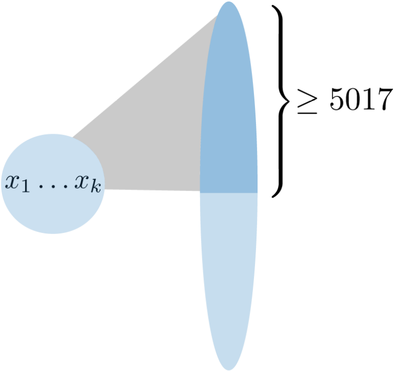
Stability

Are there k vertices dominating at least $m = 5017$ vertices?



Stability

Are there k vertices dominating at least $m = 5017$ vertices?



Let $\varepsilon > 0$. A formula φ is ε-*unstable* on a graph G if scaling the counting literals by $(1 \pm \varepsilon)$ changes whether φ is true in G.

Approximate Model-Checking

Theorem

Let $\mathcal{G}$ be a graph class with bounded expansion and $\varepsilon > 0$.

Theorem

Let $\mathcal{G}$ be a graph class with bounded expansion and $\varepsilon > 0$.
There exists an algorithm which takes $G \in \mathcal{G}$, $\varphi \in$ FO($\{>0\}$),
runs in time $f(|\varphi|)n$ and returns , , *or* .

Approximate Model-Checking

Theorem

Let $\mathcal{G}$ be a graph class with bounded expansion and $\varepsilon > 0$.
There exists an algorithm which takes $G \in \mathcal{G}$, $\varphi \in$ FO($\{>0\}$)*,*
runs in time $f(|\varphi|)n$ and returns , , *or* .

- *If*  *then φ is true on G.*

Approximate Model-Checking

Theorem

Let $\mathcal{G}$ be a graph class with bounded expansion and $\varepsilon > 0$.
There exists an algorithm which takes $G \in \mathcal{G}$, $\varphi \in$ FO($\{>0\}$),
runs in time $f(|\varphi|)n$ and returns , , *or* .

- *If*  *then φ is true on G.*
- *If*  *then φ is false on G.*

Approximate Model-Checking

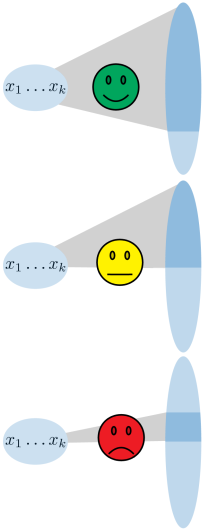
Theorem

Let $\mathcal{G}$ be a graph class with bounded expansion and $\varepsilon > 0$.
There exists an algorithm which takes $G \in \mathcal{G}$, $\varphi \in$ FO($\{>0\}$),
runs in time $f(|\varphi|)n$ and returns , , *or* .

- *If*  *then φ is true on G.*
- *If*  *then φ is false on G.*
- *If*  *then φ is ε-unstable on G.*

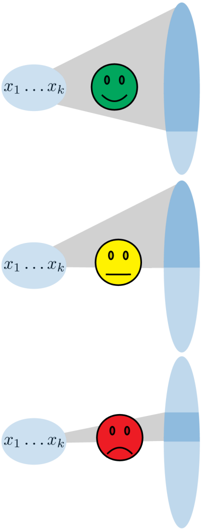
Approximate Model-Checking

PARTIAL DOMINATING SET: $\exists x_1 \ldots \exists x_k \, \#y \left(\bigvee_i y \sim x_i \wedge y = x_i \right) \geq m$



Approximate Model-Checking

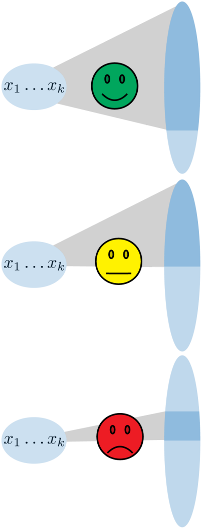
PARTIAL DOMINATING SET: $\exists x_1 \ldots \exists x_k \, \#y \left(\bigvee_i y \sim x_i \wedge y = x_i \right) \geq m$



There exists a set dominating $\geq (1+\varepsilon)m$ vertices.

Approximate Model-Checking

PARTIAL DOMINATING SET: $\exists x_1 \ldots \exists x_k \, \#y \left(\bigvee_i y \sim x_i \wedge y = x_i \right) \geq m$

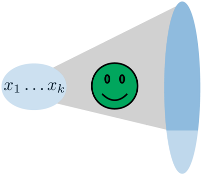


There exists a set dominating $\geq (1+\varepsilon)m$ vertices.

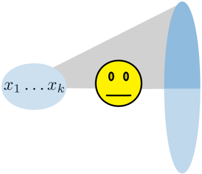
All sets dominate $< (1-\varepsilon)m$ vertices.

Approximate Model-Checking

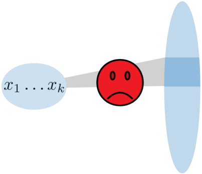
PARTIAL DOMINATING SET: $\exists x_1 \ldots \exists x_k \,\#y \left(\bigvee_i y \sim x_i \wedge y = x_i\right) \geq m$



There exists a set dominating $\geq (1+\varepsilon)m$ vertices.



All sets dominate $< (1+\varepsilon)m$ vertices and there exists a set dominating $\geq (1-\varepsilon)m$ vertices.



All sets dominate $< (1-\varepsilon)m$ vertices.

Exact Counting

Theorem

PartialDominatingSet can be solved in time $f(k)n$ on graph classes with bounded expansion.

Exact Counting

Theorem

PARTIALDOMINATINGSET can be solved in time $f(k)n$ on graph classes with bounded expansion.

This holds for all problems of the form

$$\exists x_1 \dots \exists x_k \# y \ \underbrace{\varphi(y x_1 \dots x_k)}_{\in \mathrm{FO}} \geq m.$$

How about extensions of FO($\{>0\}$)?

FO($\{>0\}$) allows comparing $\#y$ and $m \in \mathbf{N}$.

Theorem

Approximate model-checking becomes hard on trees if also allow one of the following:

How about extensions of FO($\{>0\}$)?

FO($\{>0\}$) allows comparing $\#y$ and $m \in \mathbf{N}$.

Theorem

Approximate model-checking becomes hard on trees if also allow one of the following:

- comparing $\#y$ and $\#z$ (e.g., $\#y\ \varphi > \#z\ \psi$)

How about extensions of FO($\{>0\}$)?

FO($\{>0\}$) allows comparing $\#y$ and $m \in \mathbf{N}$.

Theorem

Approximate model-checking becomes hard on trees if also allow one of the following:

- comparing $\#y$ and $\#z$ (e.g., $\#y\ \varphi > \#z\ \psi$)
- counting tuples $\#yz$ (e.g., $\#yz\ \varphi > m$)

How about extensions of FO($\{>0\}$)?

FO($\{>0\}$) allows comparing $\#y$ and $m \in \mathbf{N}$.

Theorem

Approximate model-checking becomes hard on trees if also allow one of the following:

- comparing $\#y$ and $\#z$ (e.g., $\#y\ \varphi > \#z\ \psi$)
- counting tuples $\#yz$ (e.g., $\#yz\ \varphi > m$)
- multiplying of counting terms (e.g., $\#y\ \varphi \cdot \#z\ \psi > m$)

How about extensions of FO($\{> 0\}$)?

FO($\{>0\}$) allows comparing $\#y$ and $m \in \mathbf{N}$.

Theorem

Approximate model-checking becomes hard on trees if also allow one of the following:

- comparing $\#y$ and $\#z$ (e.g., $\#y\ \varphi > \#z\ \psi$)
- counting tuples $\#yz$ (e.g., $\#yz\ \varphi > m$)
- multiplying of counting terms (e.g., $\#y\ \varphi \cdot \#z\ \psi > m$)
- subtraction of counting terms (e.g., $\#y\ \varphi - \#z\ \psi > m$)

Summary

FO($\{>0\}$) is

- hard to solve exactly on trees,

Summary

FO($\{>0\}$) is

- hard to solve exactly on trees,
- possible to approximate on bounded expansion.

Summary

FO($\{>0\}$) is

- hard to solve exactly on trees,
- possible to approximate on bounded expansion.

Slight extensions of FO($\{>0\}$) are

- hard to approximate on trees.

Summary

FO($\{>0\}$) is

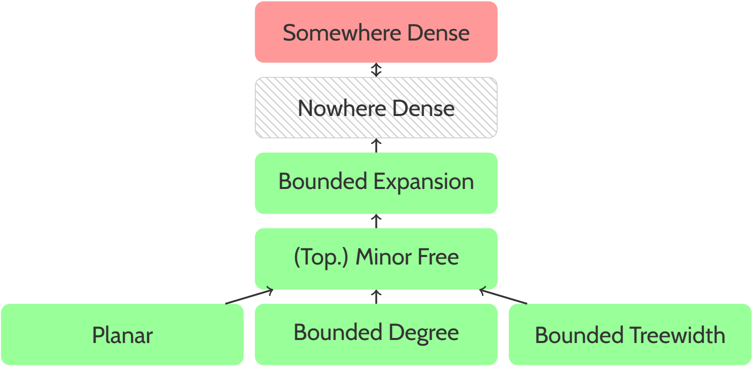
- hard to solve exactly on trees,
- possible to approximate on bounded expansion.

Slight extensions of FO($\{>0\}$) are

- hard to approximate on trees.

$\Rightarrow$ FO($\{>0\}$) seems like “the right logic” for approximation on sparse graphs

Big Question



Can we generalize our results to nowhere dense graph classes?

We want to gradually simplify this formula.

$$m_1 \leq \#x_1 \Big(\qquad\qquad\qquad\qquad\qquad\qquad \Big)$$

Proof — Quantifier Elimination

We want to gradually simplify this formula.

$$m_1 \leq \#x_1 \left(m_2 \leq \#x_2 \left(\qquad\qquad\qquad\qquad \right)\right)$$

Proof — Quantifier Elimination

We want to gradually simplify this formula.

$$m_1 \leq \#x_1 \left(m_2 \leq \#x_2 \left(m_3 \leq \#x_3 \qquad\qquad\qquad \right)\right)$$

Proof — Quantifier Elimination

We want to gradually simplify this formula.

$$m_1 \leq \#x_1 \left(m_2 \leq \#x_2 \left(m_3 \leq \#x_3 \quad \overbrace{\varphi(x_1 x_2 x_3)}^{\text{quantifer-free FO}} \quad \right)\right)$$

Proof — Quantifier Elimination

We want to gradually simplify this formula.

$$m_1 \leq \#x_1 \left(m_2 \leq \#x_2 \left(\underbrace{m_3 \leq \#x_3 \quad \overbrace{\varphi(x_1 x_2 x_3)}^{\text{quantifer-free FO}}}_{\text{replace with quantifier-free FO}} \right) \right)$$

Proof — Quantifier Elimination

We want to gradually simplify this formula.

$$m_1 \leq \#x_1 \left(m_2 \leq \#x_2 \quad \overbrace{\varphi'(x_1 x_2)}^{\text{quantifier-free FO}} \quad \right)$$

Proof — Quantifier Elimination

We want to gradually simplify this formula.

$$m_1 \leq \#x_1 \left(\underbrace{m_2 \leq \#x_2 \quad \overbrace{\varphi'(x_1 x_2)}^{\text{quantifier-free FO}}}_{\text{replace with quantifier-free FO}} \right)$$

Proof — Quantifier Elimination

We want to gradually simplify this formula.

$$m_1 \leq \#x_1 \quad \overbrace{\varphi''(x_1)}^{\text{quantifier-free FO}}$$

Proof — Quantifier Elimination

We want to gradually simplify this formula.

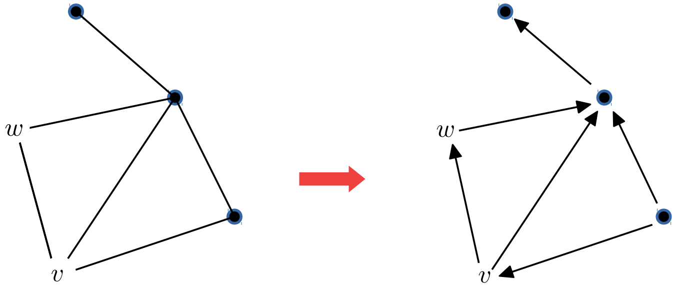
$$\underbrace{m_1 \leq \#x_1 \quad \overbrace{\varphi''(x_1)}^{\text{quantifier-free FO}}}_{\text{replace with quantifier-free FO}}$$

Proof — Quantifier Elimination

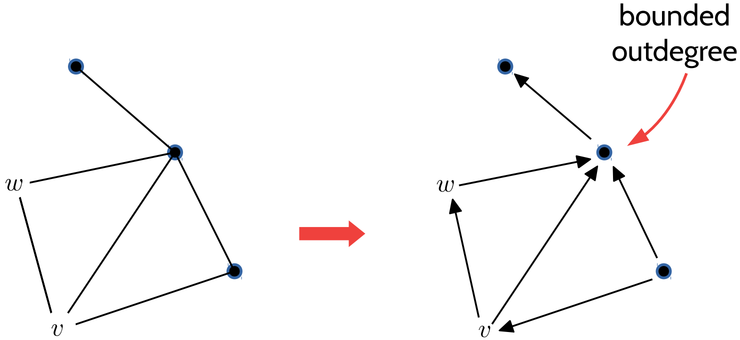
We want to gradually simplify this formula.

$$\overbrace{\varphi'''}^{\text{quantifier-free FO}}$$

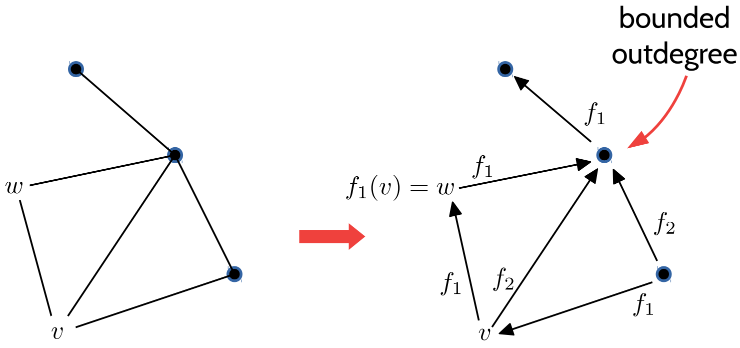
Proof — Functional Representation



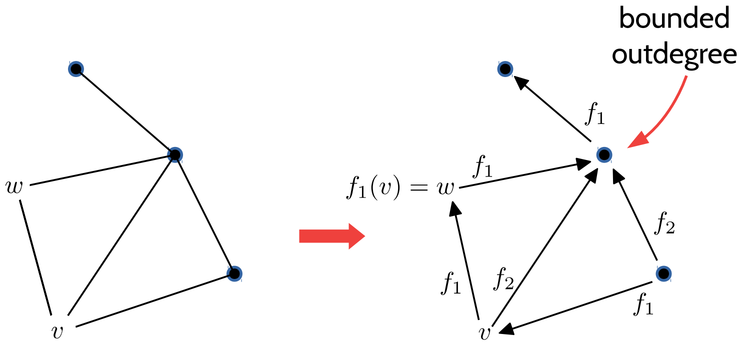
Proof — Functional Representation



Proof — Functional Representation



Proof — Functional Representation

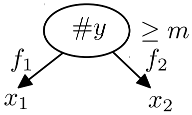


$x_1 \sim x_2$

$$\bigvee_i f_i(x_1) = x_2 \vee f_i(x_2) = x_1$$

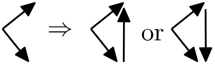
Proof — Equalities Only

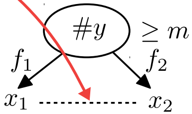
$$\#y\ \Big(\ f_1(y) = x_1 \wedge f_2(y) = x_2\ \Big) \geq m$$



Proof — Equalities Only

fraternal rule:

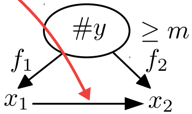




$$\#y\ \Big(\ f_1(y) = x_1 \wedge\ f_2(y) = x_2\ \Big) \geq m$$

fraternal rule:

$$\#y\ \Big(f_1(y) = x_1 \wedge f_2(y) = x_2 \Big) \geq m$$



Proof — Equalities Only

$$\bigvee_h \Big(h(x_1) = x_2 \wedge \\ \#y \; \big(f_1(y) = x_1 \wedge f_2(y) = x_2 \big) \geq m \Big)$$

$$x_1 \xrightarrow[h]{} x_2 \quad \wedge \quad \begin{tikzcd} & \#y \arrow[dl, "f_1"'] \arrow[dr, "f_2"] & \geq m \\ x_1 & & x_2 \end{tikzcd}$$

Proof — Equalities Only

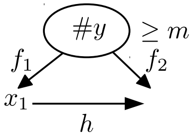
$$
\bigvee_h \Big(h(x_1) = x_2 \wedge \\
\quad \#y \; (\\
\qquad f_1(y) = x_1 \wedge \\
\qquad \cancel{f_2(y) = x_2} \\
\qquad f_2(y) = h(x_1) \\
\quad) \geq m \\
\Big)
$$

$$
x_1 \xrightarrow[h]{} x_2 \qquad \wedge \qquad
\begin{tikzcd}
& \#y \arrow[dl, "f_1"'] \arrow[dr, "f_2"] & \geq m \\
x_1 \arrow[rr, "h"'] & & {}
\end{tikzcd}
$$

Proof — Equalities Only

$$\bigvee_h \Big(h(x_1) = x_2 \wedge \#y \big(f_1(y) = x_1 \wedge \cancel{f_2(y) = x_2} \; f_2(y) = h(x_1) \big) \geq m \Big)$$

$$x_1 \xrightarrow[h]{} x_2 \qquad \wedge$$



$\#y \; \varphi(yx_1) \geq m$

Proof — Equalities Only

$$\bigvee_h \Big(h(x_1) = x_2 \wedge \\ \#y \; \big(f_1(y) = x_1 \wedge \cancel{f_2(y) = x_2} \; f_2(y) = h(x_1) \big) \geq m \Big)$$

$$\begin{tikzcd} x_1 \arrow[r, "h"'] & x_2 \end{tikzcd} \quad \wedge \quad \begin{tikzcd} & \#y \arrow[dl, "f_1"'] \arrow[dr, "f_2"] & \\ x_1 \arrow[rr, "h"'] & & {} \end{tikzcd} \geq m$$

evaluate for
every vertex

$$\#y \; \varphi(yx_1) \geq m$$

$$\iff \; R_h(x_1)$$

Proof — Equalities Only

$$\bigvee_h \Big(h(x_1) = x_2 \wedge R_h(x_1) \quad \cancel{\#y\ \big(f_1(y) = x_1 \wedge f_2(y) = x_2 \wedge f_2(y) = h(x_1) \big) \geq m} \quad \Big)$$

$$x_1 \xrightarrow{h} x_2 \qquad \wedge \qquad \begin{tikzcd} & \#y \arrow[dl, "f_1"'] \arrow[dr, "f_2"] & \geq m \\ x_1 \arrow[rr, "h"'] & & {} \end{tikzcd}$$

evaluate for
every vertex

$$\#y\ \varphi(yx_1) \geq m$$

$$\iff R_h(x_1)$$

Proof — Equalities Only

$$\bigvee_h \Big(h(x_1) = x_2 \wedge R_h(x_1) \cancel{\#y \big(f_1(y) = x_1 \wedge f_2(y) = x_2 \wedge f_2(y) = h(x_1) \big) \geq m} \Big)$$

$$x_1 \xrightarrow[h]{} x_2 \qquad \wedge \qquad \begin{tikzcd} & \#y \arrow[dl, "f_1"'] \arrow[dr, "f_2"] & \geq m \\ x_1 \arrow[rr, "h"'] & & {} \end{tikzcd}$$

evaluate for
every vertex

$$\#y\ \varphi(yx_1) \geq m$$

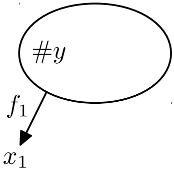
$$\iff R_h(x_1)$$

Proof — Inequalities

$$
\begin{aligned}
\#y\ \ & f_1(y) = x_1 \wedge \\
& f_2(y) \neq x_2
\end{aligned}
$$

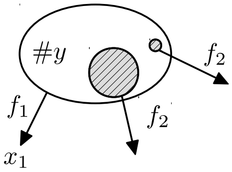
Proof — Inequalities

$$\#y \; f_1(y) = x_1 \wedge$$
$$f_2(y) \neq x_2$$



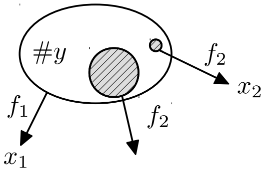
Proof — Inequalities

$$\#y\ f_1(y) = x_1 \land$$
$$f_2(y) \neq x_2$$

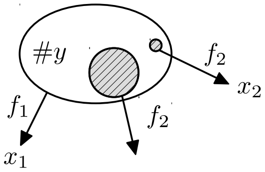


Proof — Inequalities

$$\#y\ f_1(y) = x_1 \wedge \\ f_2(y) \neq x_2$$

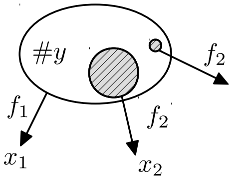


$$\#y\ f_1(y) = x_1 \wedge$$
$$\cancel{f_2(y) \neq x_2}$$



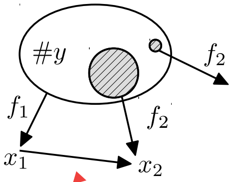
Proof — Inequalities

$$\#y\ f_1(y) = x_1 \land$$
$$f_2(y) \neq x_2$$



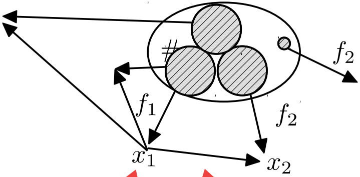
Proof — Inequalities

$$\#y\ f_1(y) = x_1 \wedge$$
$$f_2(y) \neq x_2$$



add extra edge and
proceed as before

$$\#y \; f_1(y) = x_1 \land$$
$$f_2(y) \neq x_2$$



still bounded outdegree!

add extra edge and proceed as before

Proof — Quantifier Elimination

Gradually simplify formula.

$$m_1 \leq \#x_1 \left(m_2 \leq \#x_2 \left(\underbrace{m_3 \leq \#x_3 \quad \overbrace{\varphi(x_1x_2x_3)}^{\text{quantifer-free FO}}}_{\text{replace with quantifier-free FO}}\right)\right)$$

Proof — Quantifier Elimination

Gradually simplify formula.

$$m_1 \leq \#x_1 \left(m_2 \leq \#x_2 \quad \overbrace{\varphi'(x_1 x_2)}^{\text{quantifier-free FO}} \quad \right)$$

Proof — Quantifier Elimination

Gradually simplify formula.

$$m_1 \leq \#x_1 \left(\underbrace{m_2 \leq \#x_2 \quad \overbrace{\varphi'(x_1 x_2)}^{\text{quantifier-free FO}}}_{\text{replace with quantifier-free FO}} \right)$$

Proof — Quantifier Elimination

Gradually simplify formula.

$$m_1 \leq \#x_1 \quad \overbrace{\varphi''(x_1)}^{\text{quantifier-free FO}}$$

Proof — Quantifier Elimination

Gradually simplify formula.

$$\underbrace{m_1 \leq \#x_1 \quad \overbrace{\varphi''(x_1)}^{\text{quantifier-free FO}}}_{\text{replace with quantifier-free FO}}$$

Proof — Quantifier Elimination

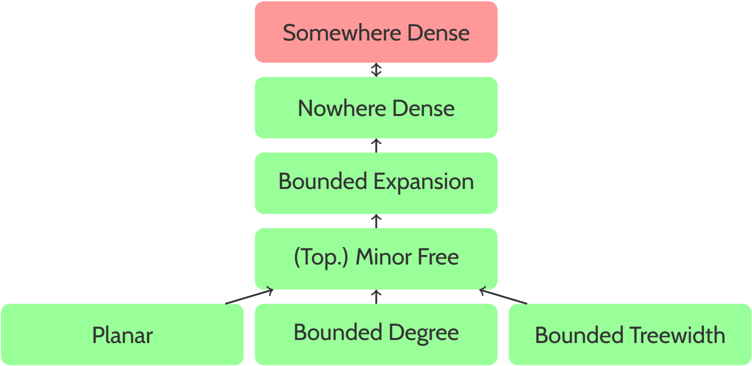
Gradually simplify formula.

$$\overbrace{\varphi'''}^{\text{quantifier-free FO}}$$

Gradually simplify formula.

Dense Graphs

Some Sparse Graph Classes

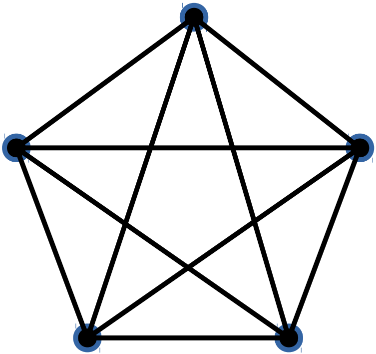


For graph classes $\mathcal{G}$ *closed under subgraphs*,
FO model-checking is tractable iff $\mathcal{G}$ is nowhere dense.

[Grohe, Kreutzer, Sieberz 2011]

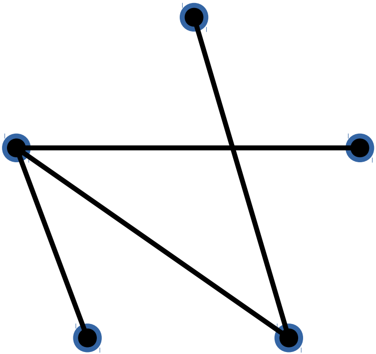
Dense Graphs

- What dense graph classes are tractable?



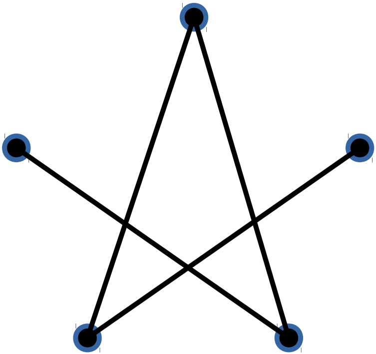
Dense Graphs

- What dense graph classes are tractable?
- Closure under subgraphs is not a good requirement.



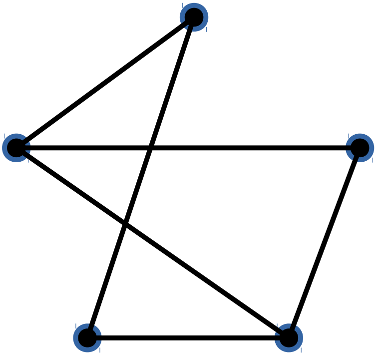
Dense Graphs

- What dense graph classes are tractable?
- Closure under subgraphs is not a good requirement.



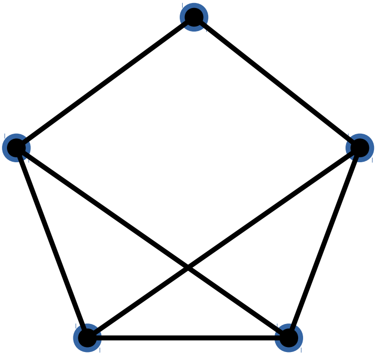
Dense Graphs

- What dense graph classes are tractable?
- Closure under subgraphs is not a good requirement.



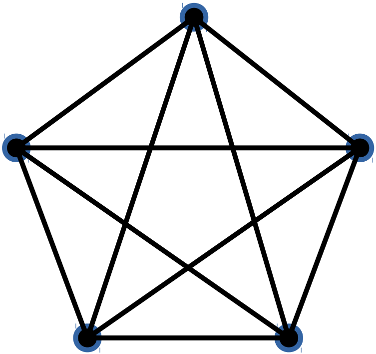
Dense Graphs

- What dense graph classes are tractable?
- Closure under subgraphs is not a good requirement.



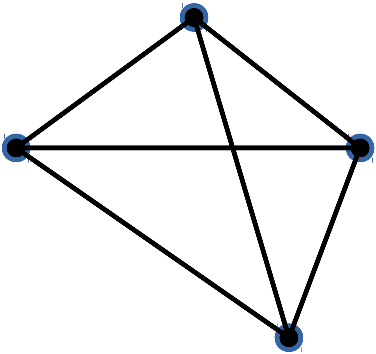
Dense Graphs

- What dense graph classes are tractable?
- Closure under subgraphs is not a good requirement.



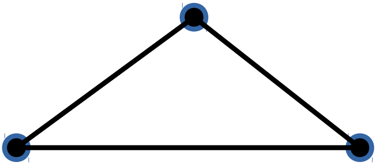
Dense Graphs

- What dense graph classes are tractable?
- Closure under subgraphs is not a good requirement.



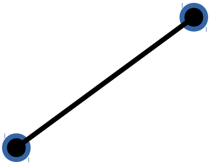
Dense Graphs

- What dense graph classes are tractable?
- Closure under subgraphs is not a good requirement.



Dense Graphs

- What dense graph classes are tractable?
- Closure under subgraphs is not a good requirement.



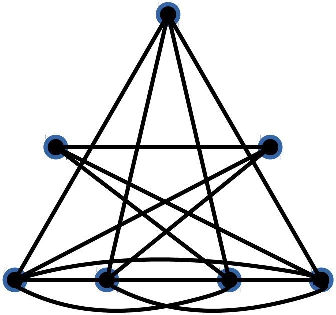
Dense Graphs

- What dense graph classes are tractable?
- Closure under subgraphs is not a good requirement.

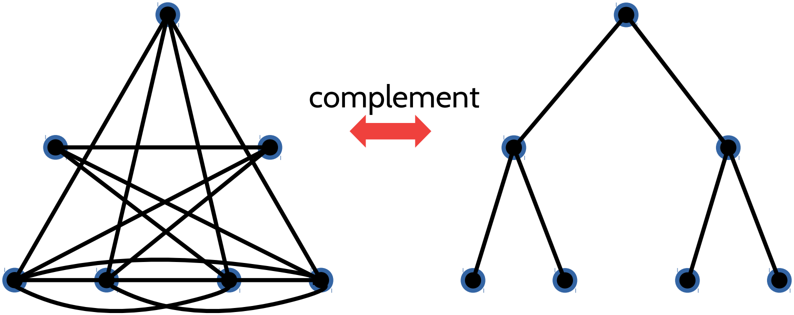
Dense Graphs

- What dense graph classes are tractable?
- Closure under subgraphs is not a good requirement.
- **Goal:** For graph classes $\mathcal{G}$ *closed under induced subgraphs*, FO model-checking is tractable iff [...].

Example: Complements

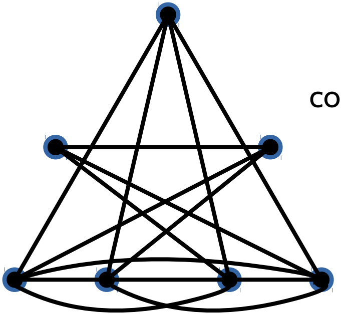


Example: Complements

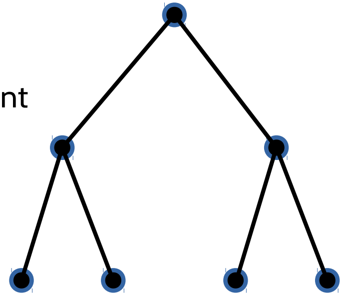


$x_1 \sim x_2$ $x_1 \not\sim x_2$

Example: Complements



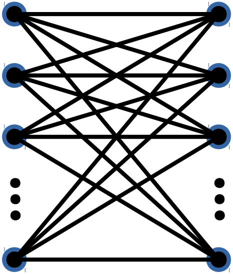
complement



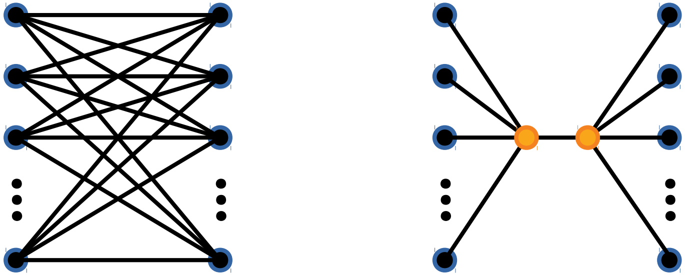
$$\exists x_1 \exists x_2 \exists x_3$$
$$x_1 \sim x_2 \wedge x_2 \sim x_3$$
$$\wedge x_1 \sim x_3$$

$$\exists x_1 \exists x_2 \exists x_3$$
$$x_1 \not\sim x_2 \wedge x_2 \not\sim x_3$$
$$\wedge x_1 \not\sim x_3$$

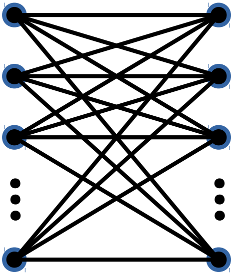
Example: Fully Bipartite



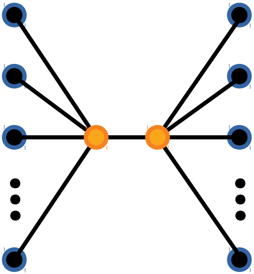
Example: Fully Bipartite



Example: Fully Bipartite

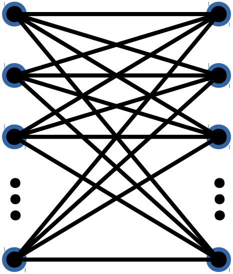


$$x \sim y$$



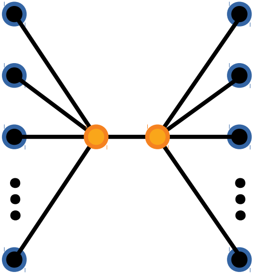
$$\mathrm{dist}(x, y) = 3$$

Example: Fully Bipartite



$$x \sim y$$

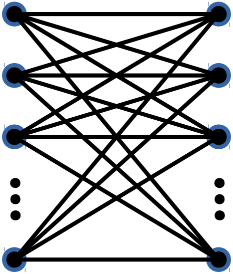
$$\exists x \ \varphi$$



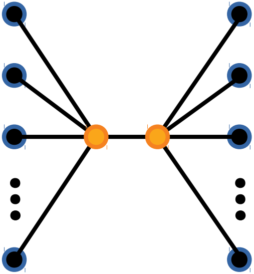
$$\mathrm{dist}(x, y) = 3$$

$$\exists x \ \mathrm{blue}(x) \wedge \varphi$$

Example: Fully Bipartite



$$\exists x_1 \exists x_2 \exists x_3$$
$$x_1 \sim x_2 \wedge x_2 \sim x_3$$
$$\wedge x_1 \sim x_3$$



$$\exists x_1 \exists x_2 \exists x_3$$
$$\mathrm{blue}(x_1) \wedge \mathrm{blue}(x_2) \wedge \mathrm{blue}(x_3)$$
$$\mathrm{dist}(x_1, x_2) = 3 \wedge \mathrm{dist}(x_2, x_3) = 3$$
$$\wedge \mathrm{dist}(x_1, x_3) = 3$$

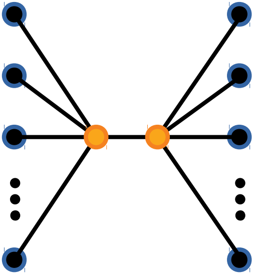
Interpretations

$$I = (\nu(x), \mu(x, y))$$

Interpretations

$$I = (\nu(x), \mu(x, y))$$

G



$I(G)$

vertices: $\{v \mid G \models \nu(v)\}$

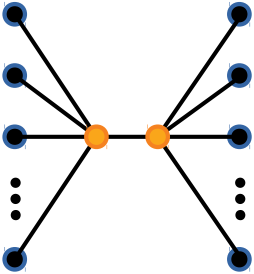
edges: $\{uv \mid G \models \mu(u, v)\}$

Interpretations

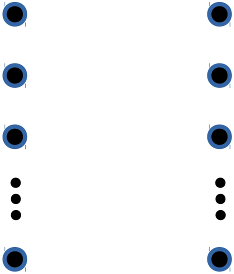
$$I = (\nu(x), \mu(x, y))$$

all blue vertices

G



$I(G)$



vertices: $\{v \mid G \models \nu(v)\}$

edges: $\{uv \mid G \models \mu(u, v)\}$

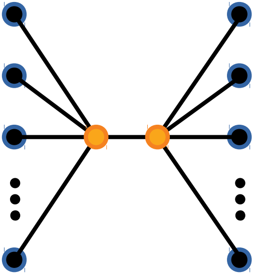
Interpretations

$$I = (\nu(x), \mu(x, y))$$

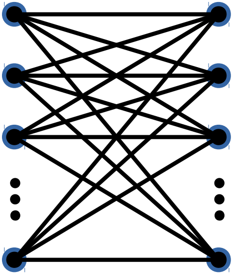
all pairs with distance 3

all blue vertices

G



$I(G)$



vertices: $\{v \mid G \models \nu(v)\}$

edges: $\{uv \mid G \models \mu(u, v)\}$

Structurally Property X

A graph class $\mathcal{G}$ has *structurally property X* if there exists

Structurally Property X

A graph class $\mathcal{G}$ has *structurally property X* if there exists

- a class $\mathcal{G}'$ with property X,

Structurally Property X

A graph class $\mathcal{G}$ has *structurally property X* if there exists

- a class $\mathcal{G}'$ with property X,
- an interpretation $I = (\nu(x), \mu(x, y))$,

Structurally Property X

A graph class $\mathcal{G}$ has *structurally property X* if there exists

- a class $\mathcal{G}'$ with property X,
- an interpretation $I = (\nu(x), \mu(x, y))$,

such that for every $G \in \mathcal{G}$ there is $G' \in \mathcal{G}'$ with $G = I(G')$.

Structurally Property X

A graph class $\mathcal{G}$ has *structurally property X* if there exists

- a class $\mathcal{G}'$ with property X,
- an interpretation $I = (\nu(x), \mu(x, y))$,

such that for every $G \in \mathcal{G}$ there is $G' \in \mathcal{G}'$ with $G = I(G')$.

The class of all fully bipartite graphs has *structurally treewidth 1*:

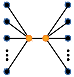
Structurally Property X

A graph class $\mathcal{G}$ has *structurally property X* if there exists

- a class $\mathcal{G}'$ with property X,
- an interpretation $I = (\nu(x), \mu(x, y))$,

such that for every $G \in \mathcal{G}$ there is $G' \in \mathcal{G}'$ with $G = I(G')$.

The class of all fully bipartite graphs has *structurally treewidth 1*:

- The class of all  has treewidth 1

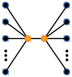
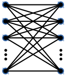
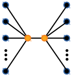
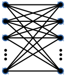
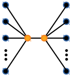
Structurally Property X

A graph class $\mathcal{G}$ has *structurally property X* if there exists

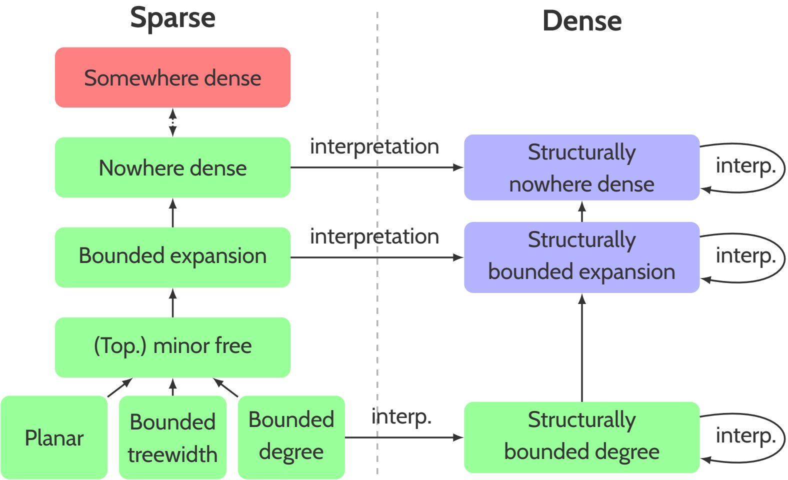
- a class $\mathcal{G}'$ with property X,
- an interpretation $I = (\nu(x), \mu(x, y))$,

such that for every $G \in \mathcal{G}$ there is $G' \in \mathcal{G}'$ with $G = I(G')$.

The class of all fully bipartite graphs has *structurally treewidth 1*:

- The class of all  has treewidth 1
- For every  there is  with  $= I\Big($  $\Big)$.

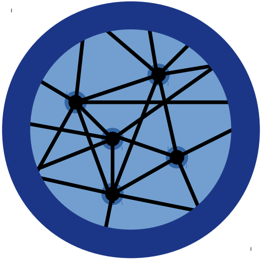
Model-Checking in Sparse and Dense Classes



Nowhere Dense: Grohe, Kreutzer, Sieberz 2011
Structurally Bounded Degree: Gajarský, Hlinenỳ, Obdržálek, Lokshtanov, Ramanujan 2016

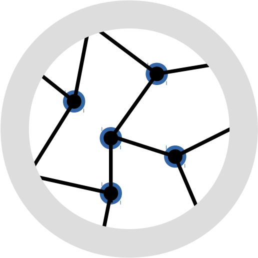
Structurally Bounded Expansion

Structurally
Bounded Expansion
$I(G)$

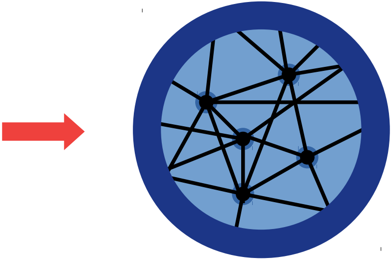


Structurally Bounded Expansion

Bounded Expansion
G

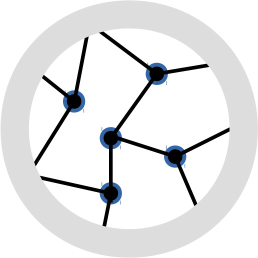


Structurally Bounded Expansion
$I(G)$



Bounded Expansion

G



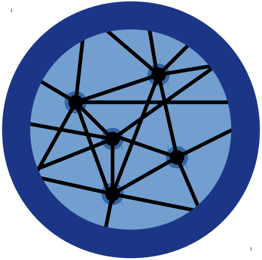
$I = (\nu(x), \mu(x, y))$

$x \sim y \to \mu(x, y)$

$\exists x \to \exists x \nu(x) \wedge$

Structurally Bounded Expansion

$I(G)$



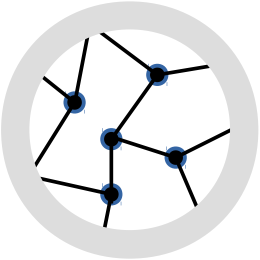
φ'

φ

Structurally Bounded Expansion

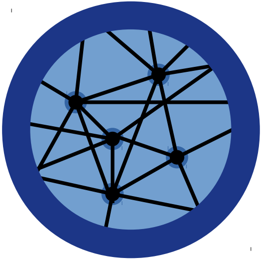
Bounded Expansion

G



Structurally Bounded Expansion

$I(G)$



$I = (\nu(x), \mu(x,y))$

$x \sim y \to \mu(x,y)$

$\exists x \to \exists x \nu(x) \wedge$

φ'

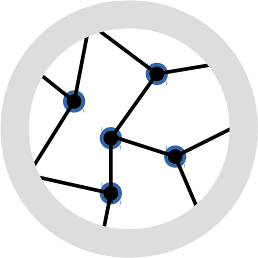
φ

MC-algorithm

Structurally Bounded Expansion

Bounded Expansion

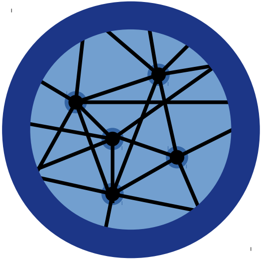
G



$I = (\nu(x), \mu(x,y))$

Structurally Bounded Expansion

$I(G)$



$x \sim y \to \mu(x,y)$

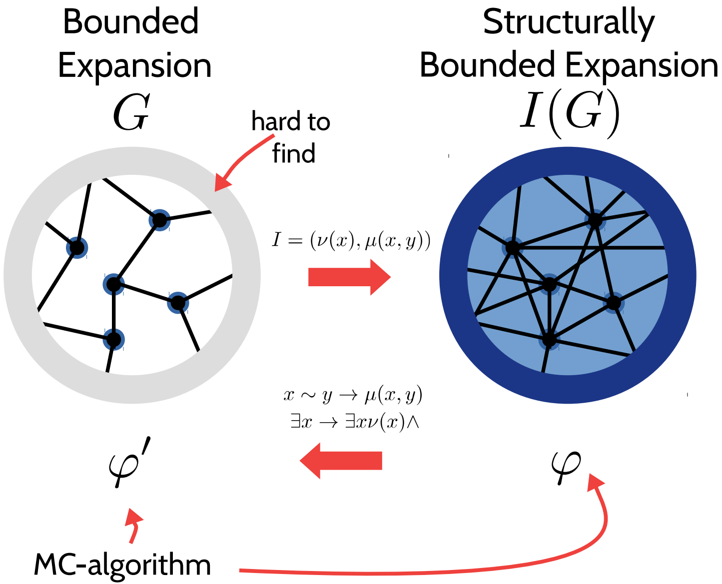
$\exists x \to \exists x \nu(x) \wedge$

φ'

φ

MC-algorithm

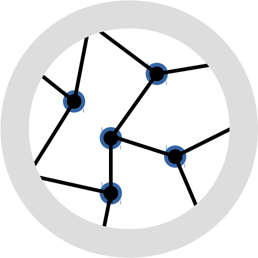
Structurally Bounded Expansion



Structurally Bounded Degree

Degree 3

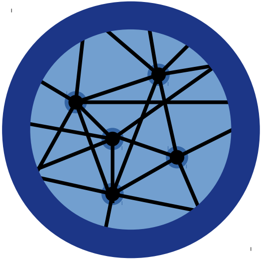
G



I

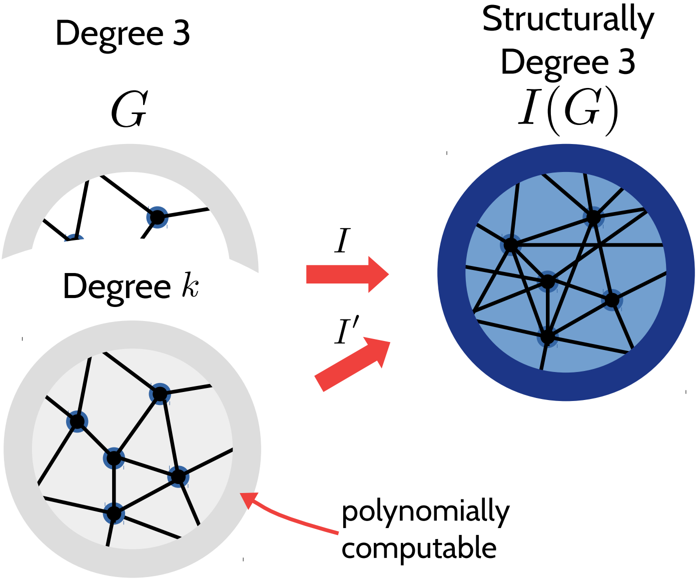
Structurally
Degree 3

$I(G)$

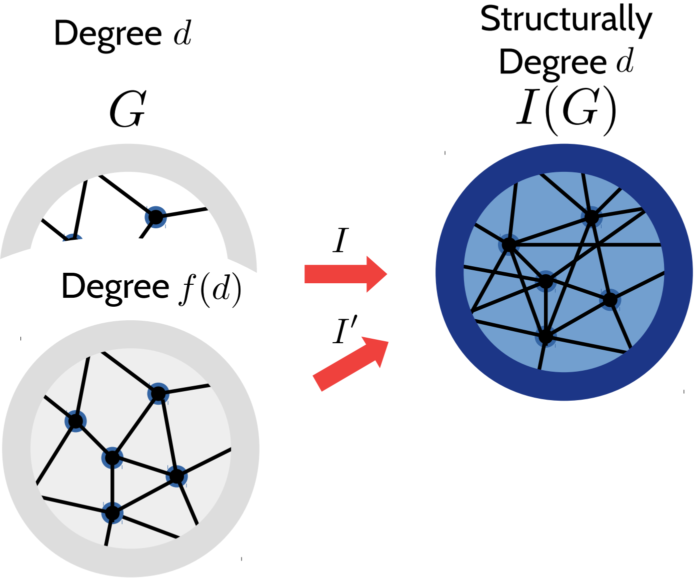


NP-complete to find preimage

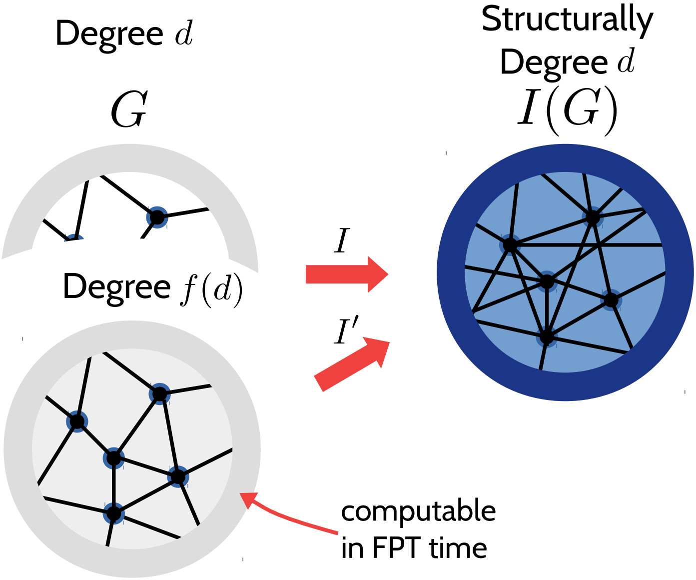
Structurally Bounded Degree



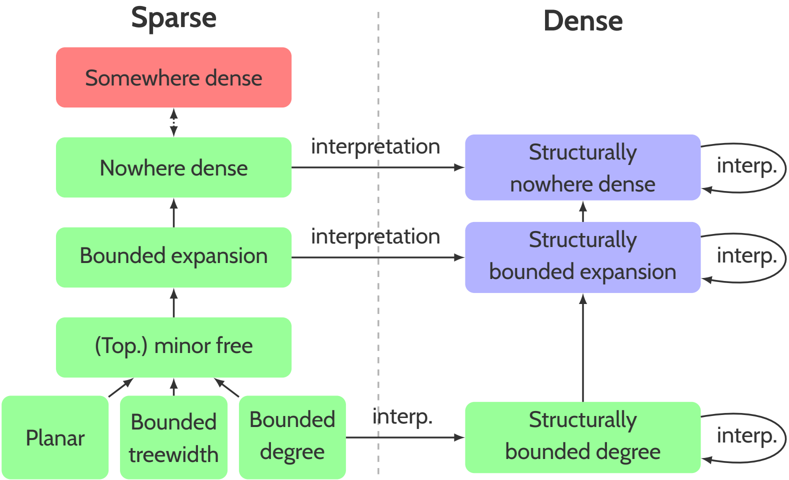
Structurally Bounded Degree



Structurally Bounded Degree



Model-Checking in Sparse and Dense Classes

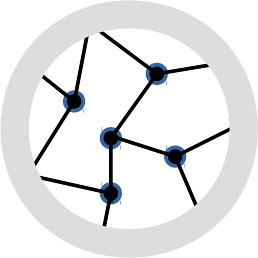


Nowhere Dense: Grohe, Kreutzer, Sieberz 2011
Structurally Bounded Degree: Gajarský, Hlineny̌, Obdržálek, Lokshtanov, Ramanujan 2016

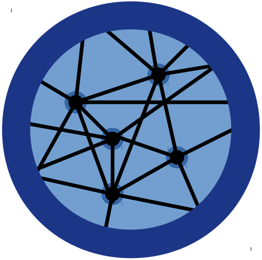
Big Question

Bounded Expansion

G



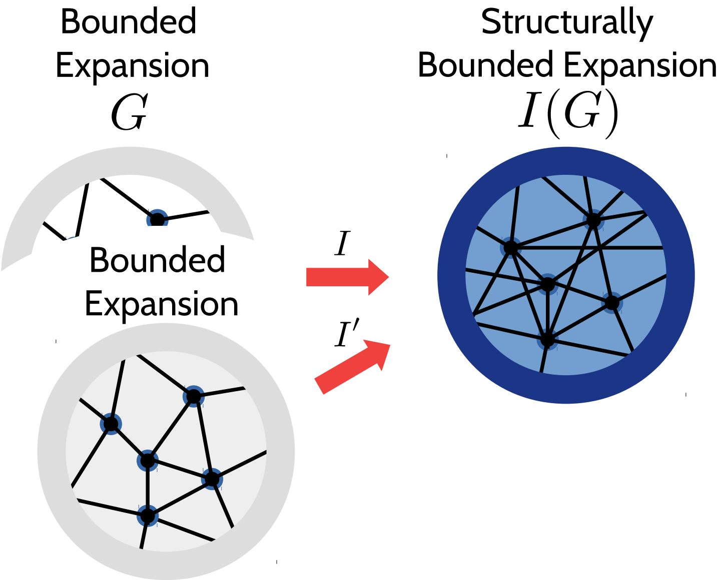
$\xrightarrow{I}$



Structurally Bounded Expansion

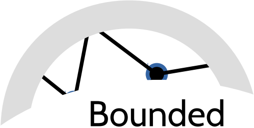
$I(G)$

Big Question

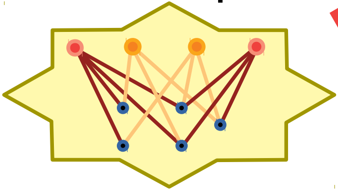


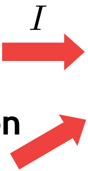
Big Question

Bounded
Expansion
G

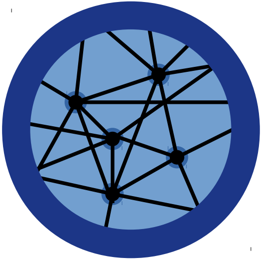


Bounded
Expansion
Lacon Decomposition

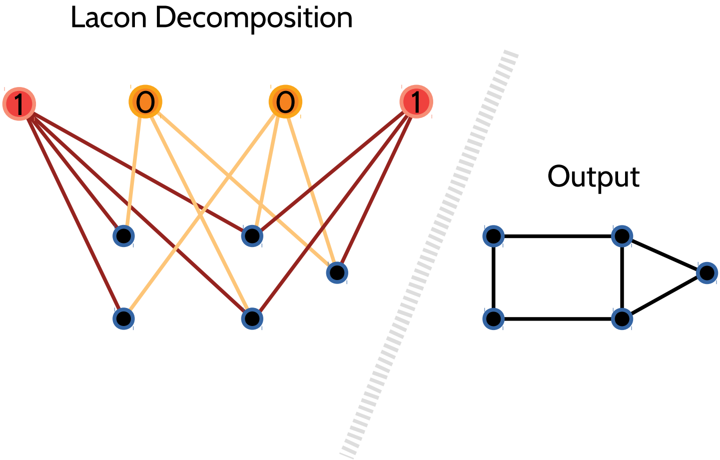




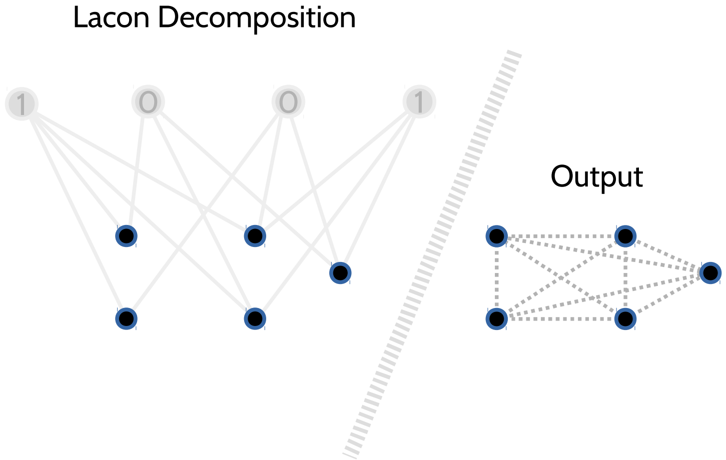
Structurally
Bounded Expansion
$I(G)$



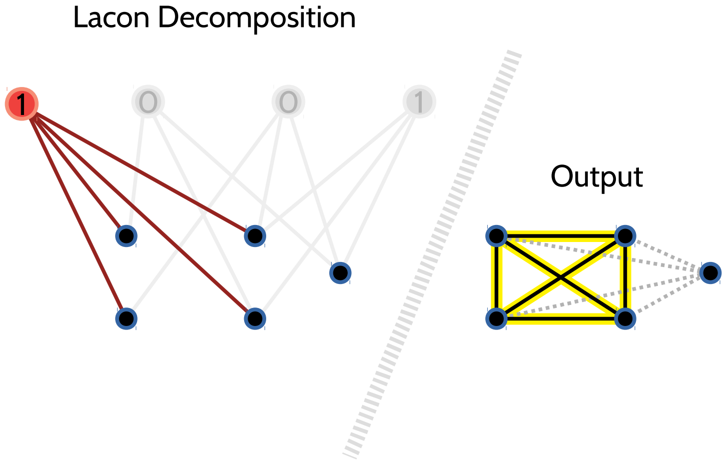
Lacon Decompositions

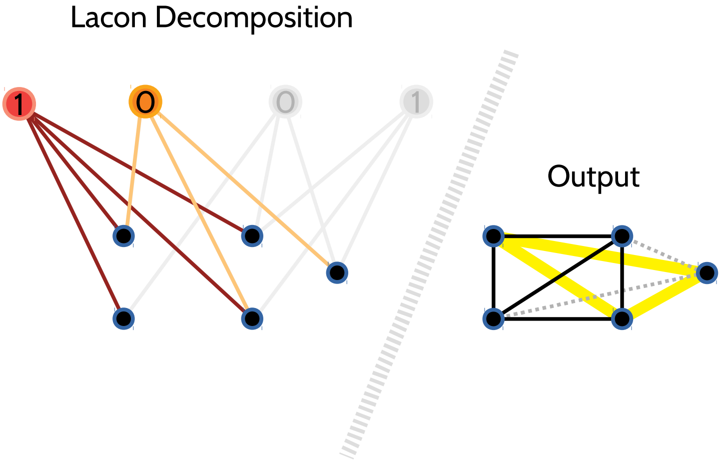


Lacon Decompositions

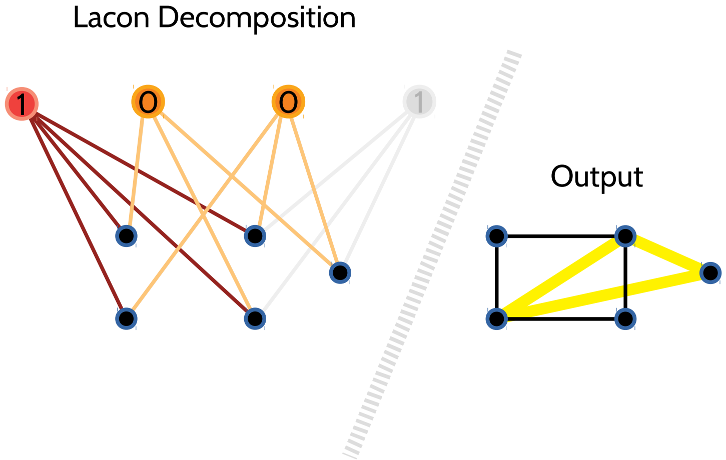


Lacon Decompositions

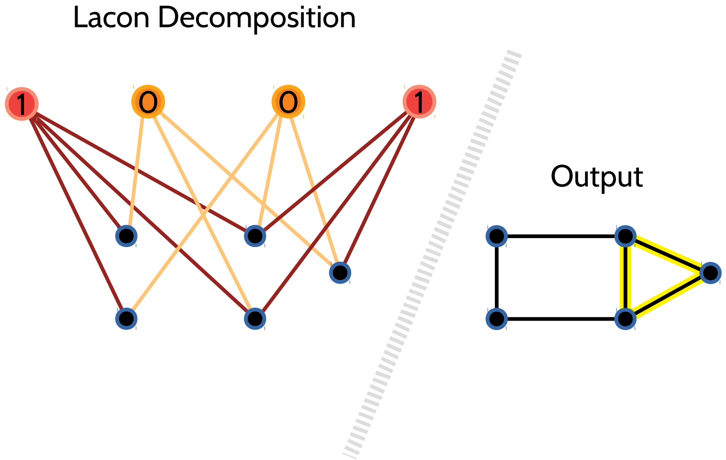


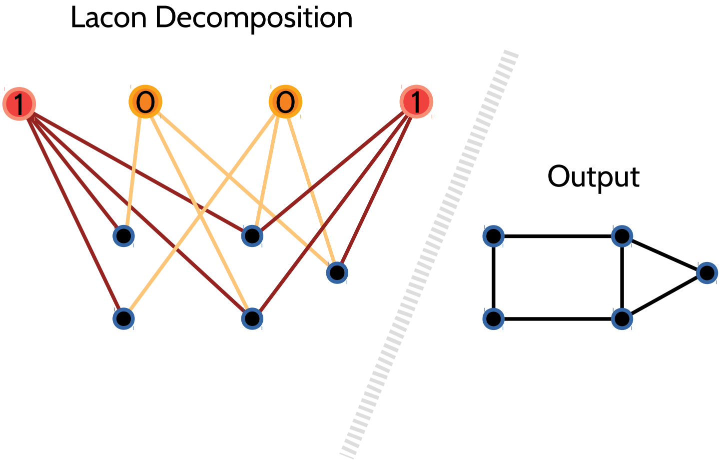

Lacon Decomposition
1
0
0
1
Output

Lacon Decompositions



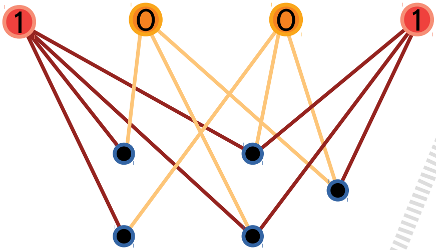
Lacon Decompositions



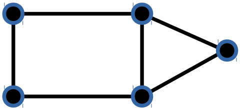

Lacon Decomposition
1
0
0
1
Output

Lacon Decompositions

Lacon Decomposition

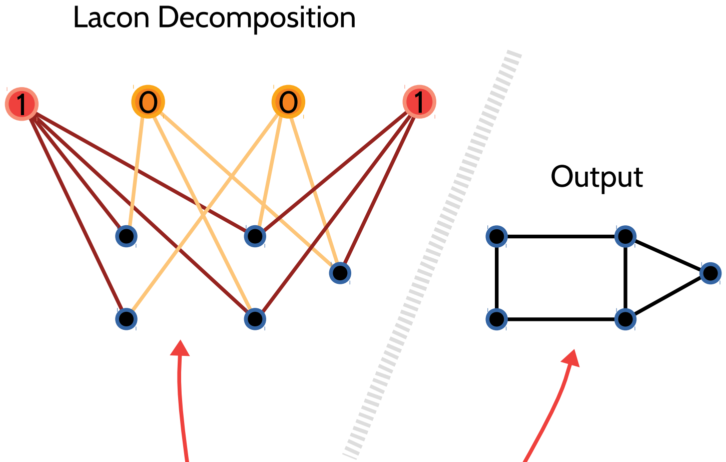


Output



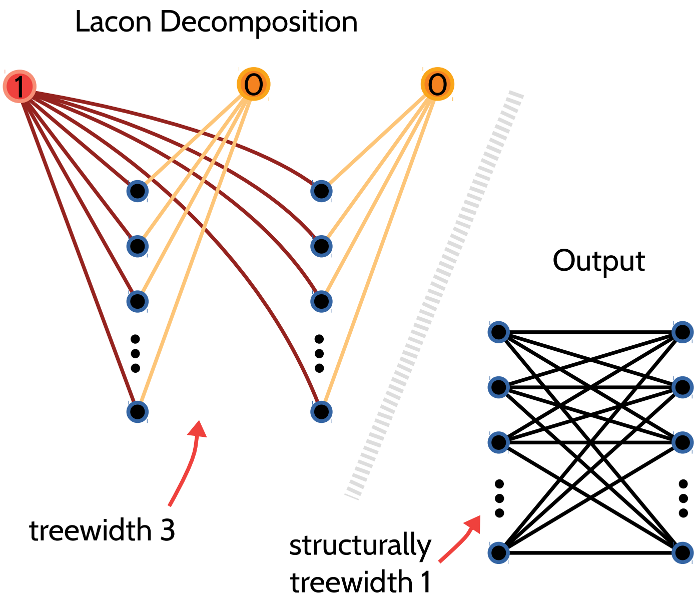
has lacon decomposition with property X $\implies$ structurally property X

Lacon Decompositions

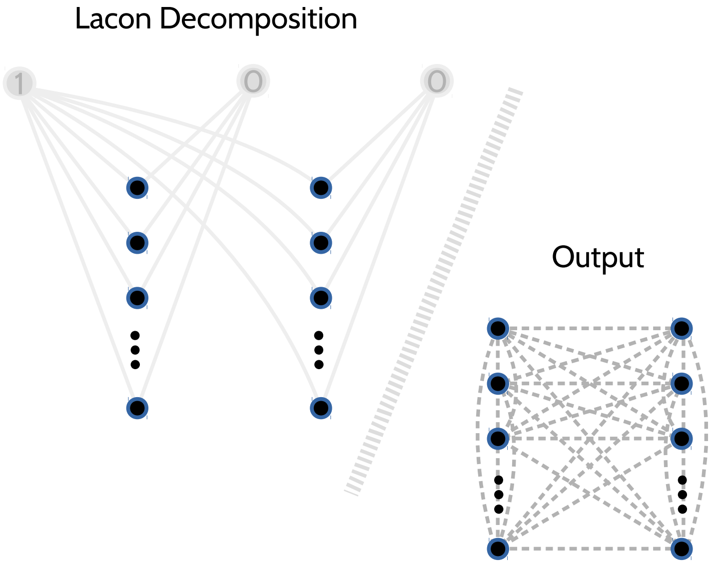


has lacon decomposition with property X $\iff$ structurally property X

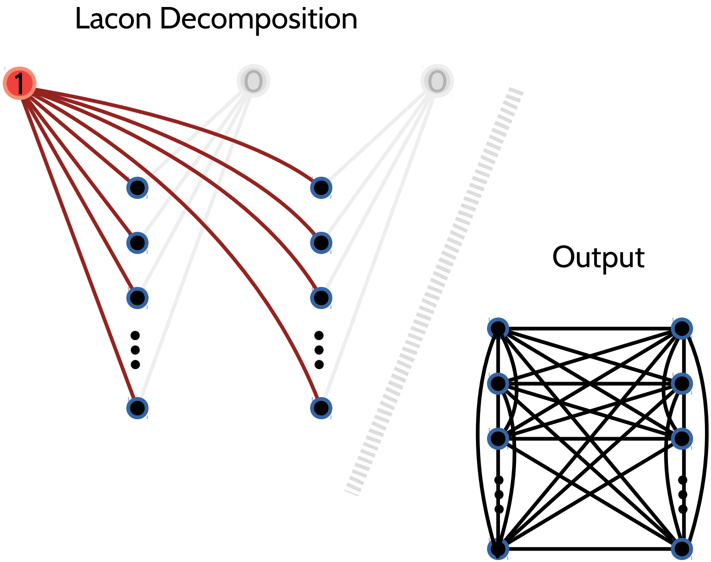
Lacon Decompositions

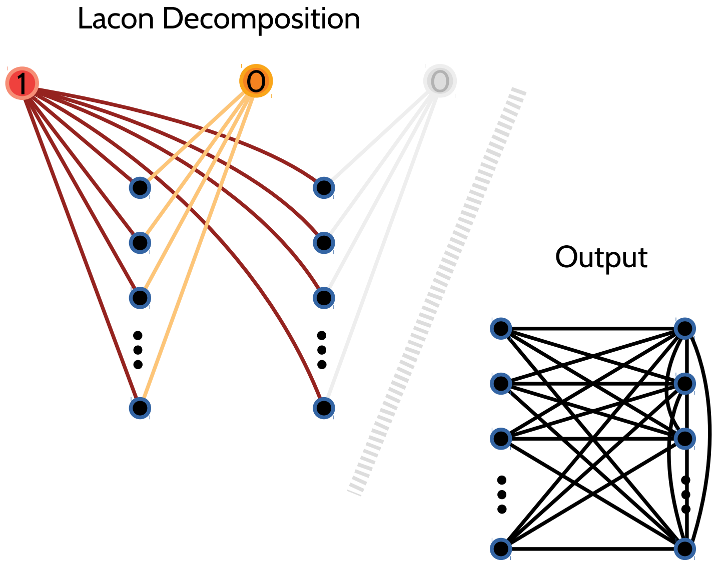


Lacon Decompositions

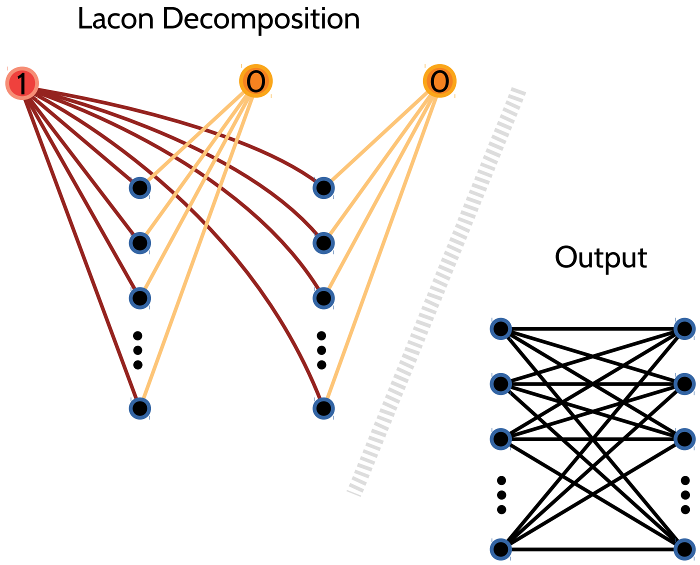


Lacon Decompositions



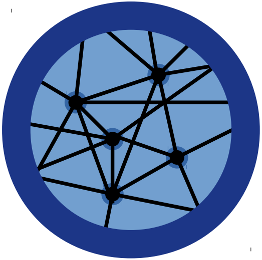

Lacon Decomposition
1
0
0
Output

Lacon Decompositions



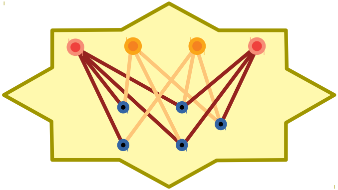
Big Question

Structurally
Bounded Expansion

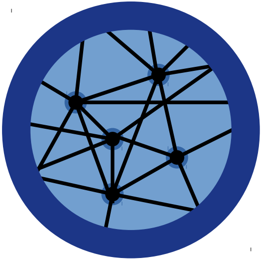


Big Question

has Lacon Decomposition
with Bounded Expansion

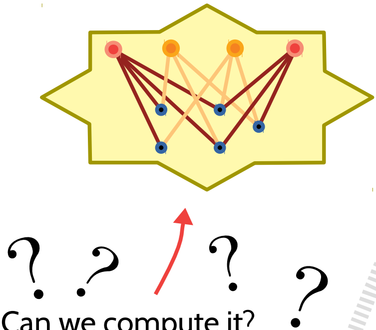


Structurally
Bounded Expansion



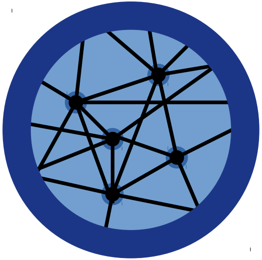
Big Question

has Lacon Decomposition
with Bounded Expansion

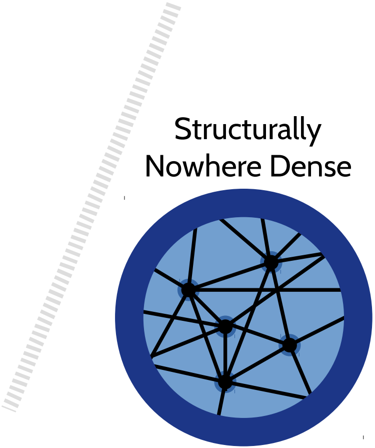


Can we compute it?
That would solve the
model-checking problem.

Structurally
Bounded Expansion

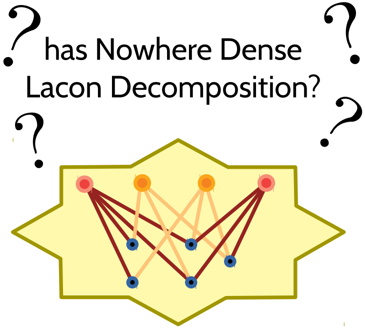


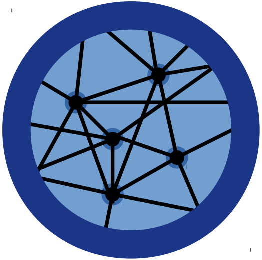
Big Question



Structurally Nowhere Dense

Big Question





End

Thanks!