Logic meets **graph theory** and **algorithm design**

Michał Pilipczuk
University of Warsaw

Logic Mentoring Workshop @ CSL 2022

February 14th, 2022

Graph algorithms

Graph algorithms

Area of **graph algorithms**:

Graph algorithms

Area of **graph algorithms**:

- **I:** A graph G and some parameters $\bar{k}$.

Graph algorithms

Area of **graph algorithms**:

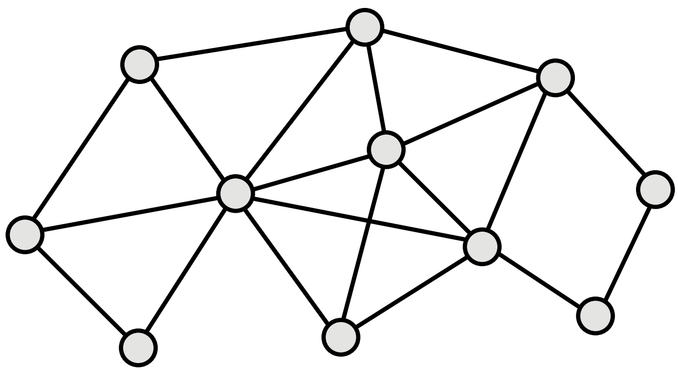
- **I:** A graph G and some parameters $\bar{k}$.
- **Q:** Decide the existence of some object based on G and $\bar{k}$.

Graph algorithms

Area of **graph algorithms**:

- **I:** A graph G and some parameters $\bar{k}$.
- **Q:** Decide the existence of some object based on G and $\bar{k}$.

Examples:



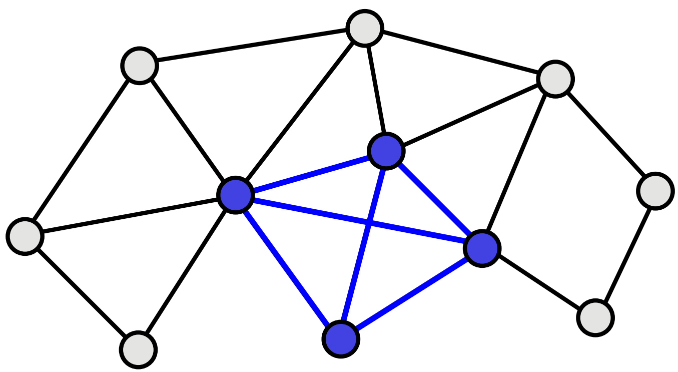
Graph algorithms

Area of **graph algorithms**:

- **I:** A graph G and some parameters $\bar{k}$.
- **Q:** Decide the existence of some object based on G and $\bar{k}$.

Examples:

- CLIQUE: Does G have k pairwise adjacent vertices?



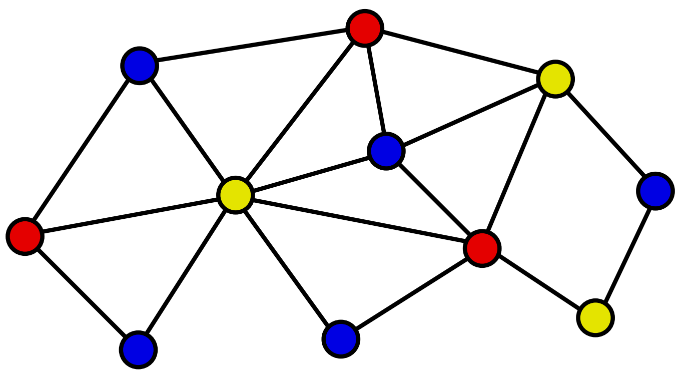
Graph algorithms

Area of **graph algorithms**:

- **I:** A graph G and some parameters $\bar{k}$.
- **Q:** Decide the existence of some object based on G and $\bar{k}$.

Examples:

- Clique: Does G have k pairwise adjacent vertices?
- 3-Coloring: Does G have a proper coloring using 3 colors?



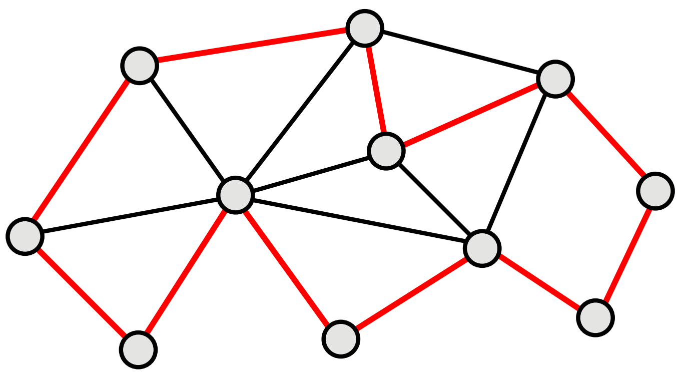
Graph algorithms

Area of **graph algorithms**:

- **I:** A graph G and some parameters $\bar{k}$.
- **Q:** Decide the existence of some object based on G and $\bar{k}$.

Examples:

- Clique: Does G have k pairwise adjacent vertices?
- 3-Coloring: Does G have a proper coloring using 3 colors?
- Hamiltonicity: Does G have a cycle visiting every vertex once?



Meta-theorems

Meta-theorems

Typical: Take a problem P, solve it efficiently on some class $\mathscr{C}$ of graphs.

Meta-theorems

Typical: Take a problem P, solve it efficiently on some class $\mathscr{C}$ of graphs.

– Often **highly non-trivial** and **problem-dependent**.

Meta-theorems

Typical: Take a problem P, solve it efficiently on some class $\mathscr{C}$ of graphs.

– Often **highly non-trivial** and **problem-dependent**.

– **Common denominators:** techniques that work for certain problems.

Meta-theorems

Typical: Take a problem P, solve it efficiently on some class $\mathscr{C}$ of graphs.

- Often **highly non-trivial** and **problem-dependent**.
- **Common denominators:** techniques that work for certain problems.

Idea: Look at **classes of problems**.

Meta-theorems

Typical: Take a problem P, solve it efficiently on some class $\mathscr{C}$ of graphs.

- Often **highly non-trivial** and **problem-dependent**.
- **Common denominators:** techniques that work for certain problems.

Idea: Look at **classes of problems**.

- For instance, all problems expressible in a **logic** $\mathcal{L}$.

Meta-theorems

Typical: Take a problem P, solve it efficiently on some class $\mathscr{C}$ of graphs.

- Often **highly non-trivial** and **problem-dependent**.
- **Common denominators:** techniques that work for certain problems.

Idea: Look at **classes of problems**.

- For instance, all problems expressible in a **logic** $\mathcal{L}$.

Theorem (Meta-theorem template)

Every problem expressible in $\mathcal{L}$ can be solved in time *Blah*

on every graph from $\mathscr{C}$.

Meta-theorems

Typical: Take a problem P, solve it efficiently on some class $\mathscr{C}$ of graphs.

– Often **highly non-trivial** and **problem-dependent**.
– **Common denominators:** techniques that work for certain problems.

Idea: Look at **classes of problems**.

– For instance, all problems expressible in a **logic** $\mathcal{L}$.

Theorem (Meta-theorem template)

Every problem expressible in $\mathcal{L}$ can be solved in time *Blah*
on every graph from $\mathscr{C}$.

Explains the range of applicability of certain techniques.

Meta-theorems

Typical: Take a problem P, solve it efficiently on some class $\mathscr{C}$ of graphs.

- Often **highly non-trivial** and **problem-dependent**.
- **Common denominators:** techniques that work for certain problems.

Idea: Look at **classes of problems**.

- For instance, all problems expressible in a **logic** $\mathcal{L}$.

Theorem (Meta-theorem template)

Every problem expressible in $\mathcal{L}$ can be solved in time *Blah* on every graph from $\mathscr{C}$.

Explains the range of applicability of certain techniques.

Model-checking $\mathcal{L}$ on $\mathscr{C}$: Given $\varphi \in \mathcal{L}$ and $G \in \mathscr{C}$, decide $G \models \varphi$.

Logic on graphs

Logic on graphs

$$\textsc{Clique}_k \;=\; \exists x_1 \exists x_2 \ldots \exists x_k \; \bigwedge_{i \neq j} [(x_i \neq x_j) \wedge \mathrm{adj}(x_i, x_j)]$$

Logic on graphs

$$\textsc{Clique}_k = \exists x_1 \exists x_2 \ldots \exists x_k \bigwedge_{i \neq j} [(x_i \neq x_j) \wedge \mathrm{adj}(x_i, x_j)]$$

First-Order logic (FO):

– variables for single vertices, can check adjacency

Logic on graphs

$$\textsc{Clique}_k \;=\; \exists x_1\, \exists\, x_2 \ldots \exists\, x_k\; \bigwedge_{i\neq j} [(x_i \neq x_j) \wedge \mathrm{adj}(x_i, x_j)]$$

First-Order logic (FO):

– variables for single vertices, can check adjacency

3-$\textsc{Coloring} \;=\; \exists A\, \exists B\, \exists C \quad (A, B, C)$ is a partition of V and

every two adjacent vertices are colored differently

Logic on graphs

$$\textsc{Clique}_k \;=\; \exists x_1\, \exists\, x_2 \ldots \exists\, x_k\; \bigwedge_{i \neq j} [(x_i \neq x_j) \wedge \mathrm{adj}(x_i, x_j)]$$

First-Order logic (FO):

– variables for single vertices, can check adjacency

$$\text{3-}\textsc{Coloring} \;=\; \exists A\, \exists B\, \exists C \quad (A, B, C) \text{ is a partition of } V \text{ and}$$
$$\text{every two adjacent vertices are colored differently}$$

Monadic Second-Order logic, first variant (MSO_1):

– variables for single vertices and sets of vertices, can check membership

Logic on graphs

$$\textsc{Clique}_k \;=\; \exists x_1 \,\exists\, x_2 \ldots \exists\, x_k \; \bigwedge_{i \neq j} [(x_i \neq x_j) \wedge \mathrm{adj}(x_i, x_j)]$$

First-Order logic (FO):

– variables for single vertices, can check adjacency

$$\textsc{3-Coloring} \;=\; \exists A \,\exists B \,\exists C \quad (A, B, C) \text{ is a partition of } V \text{ and}$$
$$\text{every two adjacent vertices are colored differently}$$

Monadic Second-Order logic, first variant (MSO_1):

– variables for single vertices and sets of vertices, can check membership

$$\textsc{Hamiltonicity} \;=\; \exists S \subseteq E \quad S \text{ is connected and}$$
$$\text{every vertex is incident to exactly two edges of } S$$

Logic on graphs

$$\textsc{Clique}_k \;=\; \exists x_1 \exists x_2 \ldots \exists x_k \; \bigwedge_{i \neq j} [(x_i \neq x_j) \wedge \mathrm{adj}(x_i, x_j)]$$

First-Order logic (FO):

– variables for single vertices, can check adjacency

$$3\text{-}\textsc{Coloring} \;=\; \exists A \exists B \exists C \quad (A, B, C) \text{ is a partition of } V \text{ and}$$
$$\text{every two adjacent vertices are colored differently}$$

Monadic Second-Order logic, first variant (MSO_1):

– variables for single vertices and sets of vertices, can check membership

$$\textsc{Hamiltonicity} \;=\; \exists S \subseteq E \quad S \text{ is connected and}$$
$$\text{every vertex is incident to exactly two edges of } S$$

Monadic Second-Order logic, second variant (MSO_2):

– vars for (sets of) vertices & edges, can check membership & incidence

MSO_2 on graphs: complexity

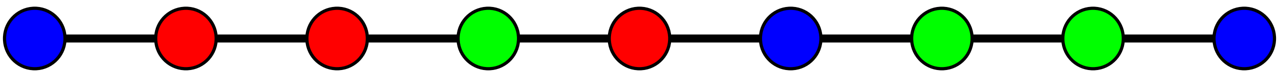
MSO_2 on graphs: complexity

HAMILTONICITY is NP-complete $\rightsquigarrow$ MC MSO_2 intractable on general graphs

MSO_2 on graphs: complexity

HAMILTONICITY is NP-complete $\rightsquigarrow$ MC MSO_2 intractable on general graphs

Obs: Every MSO_2-definable problem can be decided in time $\mathcal{O}(n)$ on **colored paths**.

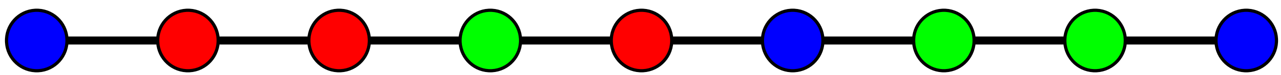


MSO_2 on graphs: complexity

HAMILTONICITY is NP-complete $\rightsquigarrow$ MC MSO_2 intractable on general graphs

Obs: Every MSO_2-definable problem can be decided in time $\mathcal{O}(n)$ on **colored paths**.

That is, model-checking MSO_2 in time $f(\varphi) \cdot n$.



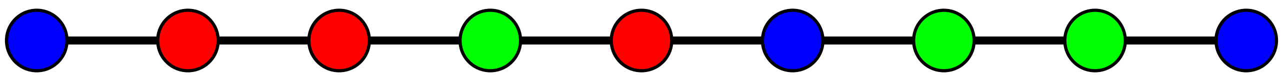
MSO_2 on graphs: complexity

HAMILTONICITY is NP-complete $\rightsquigarrow$ MC MSO_2 intractable on general graphs

Obs: Every MSO_2-definable problem can be decided in time $\mathcal{O}(n)$ on **colored paths**.

That is, model-checking MSO_2 in time $f(\varphi) \cdot n$.

Sketch:



MSO_2 on graphs: complexity

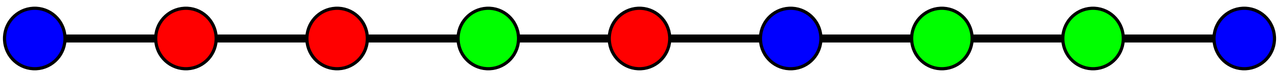
HAMILTONICITY is NP-complete $\rightsquigarrow$ MC MSO_2 intractable on general graphs

Obs: Every MSO_2-definable problem can be decided in time $\mathcal{O}(n)$ on **colored paths**.

That is, model-checking MSO_2 in time $f(\varphi) \cdot n$.

Sketch:

– Colored path $\rightsquigarrow$ Word $w \in \Sigma^\star$.



a *b* *b* *c* *b* *a* *c* *c* *a*

MSO_2 on graphs: complexity

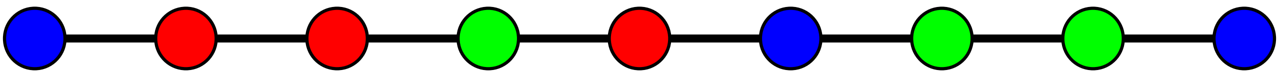
HAMILTONICITY is NP-complete $\rightsquigarrow$ MC MSO_2 intractable on general graphs

Obs: Every MSO_2-definable problem can be decided in time $\mathcal{O}(n)$ on **colored paths**.

That is, model-checking MSO_2 in time $f(\varphi) \cdot n$.

Sketch:

- Colored path $\rightsquigarrow$ Word $w \in \Sigma^\star$.
- Sentence φ defining the problem $\rightsquigarrow$ Finite automaton $\mathcal{A}$



$a \quad b \quad b \quad c \quad b \quad a \quad c \quad c \quad a$

MSO_2 on graphs: complexity

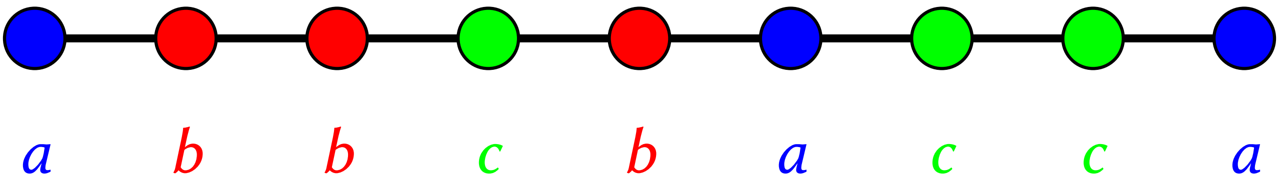
HAMILTONICITY is NP-complete $\rightsquigarrow$ MC MSO_2 intractable on general graphs

Obs: Every MSO_2-definable problem can be decided in time $\mathcal{O}(n)$ on **colored paths**.

That is, model-checking MSO_2 in time $f(\varphi) \cdot n$.

Sketch:

- Colored path $\rightsquigarrow$ Word $w \in \Sigma^\star$.
- Sentence φ defining the problem $\rightsquigarrow$ Finite automaton $\mathcal{A}$
- Just run $\mathcal{A}$ on w in linear time.



Treewidth

Treewidth

The same idea will work on **colored trees**.

Treewidth

The same idea will work on **colored trees**. (MSO on trees = tree automata)

Treewidth

The same idea will work on **colored trees**. (MSO on trees = tree automata)

Q: How far can we go?

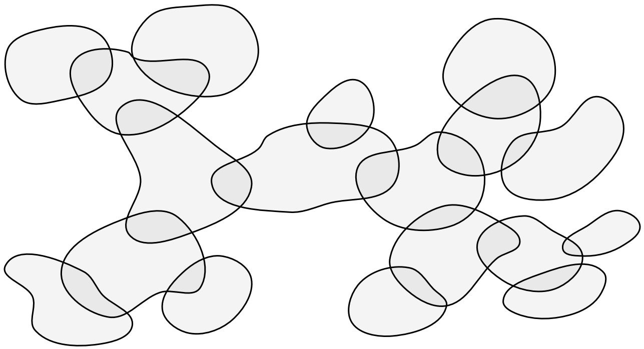
Treewidth

The same idea will work on **colored trees**. (MSO on trees = tree automata)

Q: How far can we go?

Definition (Treewidth)

A graph has **treewidth** k if it can be confined to a tree of **bags**, each of size $\leqslant k$.



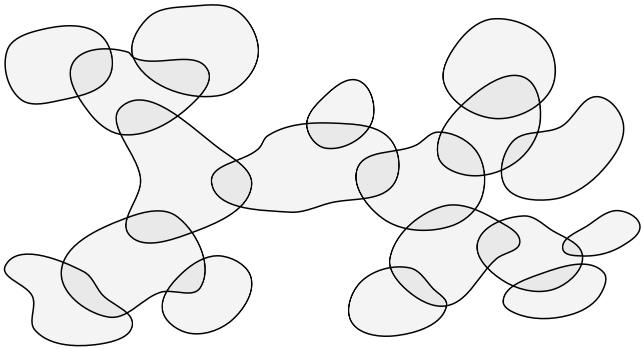
Treewidth

The same idea will work on **colored trees**. (MSO on trees = tree automata)

Q: How far can we go?

Definition (Treewidth)

A graph has **treewidth** k if it can be confined to a tree of **bags**, each of size $\leqslant k$.



Theorem (Courcelle)

For every fixed k, every MSO_2-definable problem can be decided in linear time on graphs of treewidth $\leqslant k$.

Grid minors

Grid minors

Is this it? Can we go beyond bounded treewidth?

Grid minors

Is this it? Can we go beyond bounded treewidth?

Q: How do graphs of large treewidth look like?

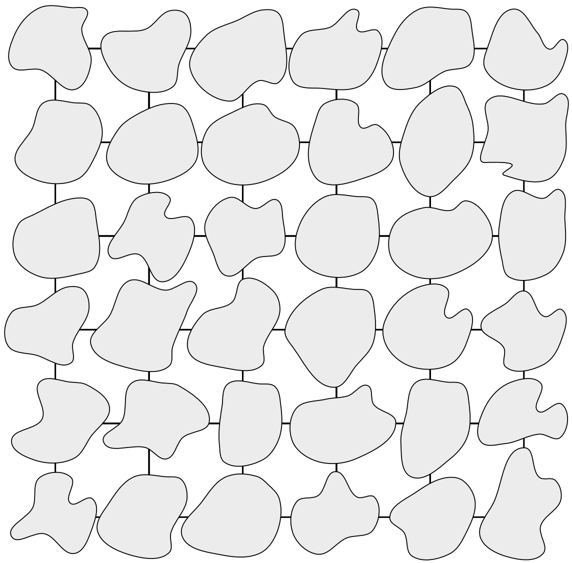
Grid minors

Is this it? Can we go beyond bounded treewidth?

Q: How do graphs of large treewidth look like?

Theorem (Excluded Grid Minor)

There is a function $f\colon \mathbb{N} \to \mathbb{N}$ such that if the treewidth of G is larger than $f(k)$, then G contains a $k \times k$ **grid minor**.



Complexity of MSO_2 model-checking

Complexity of MSO_2 model-checking

Consider a class of graphs $\mathscr{C}$.

Complexity of MSO_2 model-checking

Consider a class of graphs $\mathscr{C}$.

$\mathscr{C}$ has **bnd treewidth** $\Rightarrow$

Every MSO_2-definable problem solvable in linear time

Complexity of MSO_2 model-checking

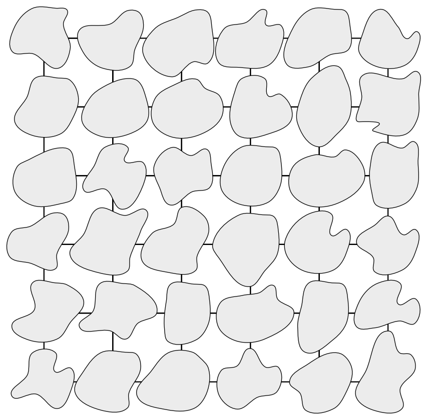
Consider a class of graphs $\mathscr{C}$.

$\mathscr{C}$ has **bnd treewidth** $\Rightarrow$

Every MSO_2-definable problem solvable in linear time

$\mathscr{C}$ has **unbnd treewidth** $\Rightarrow$

$\mathscr{C}$ contains arbitrarily large grid minors



Complexity of MSO_2 model-checking

Consider a class of graphs $\mathscr{C}$.

$\mathscr{C}$ has **bnd treewidth** $\Rightarrow$

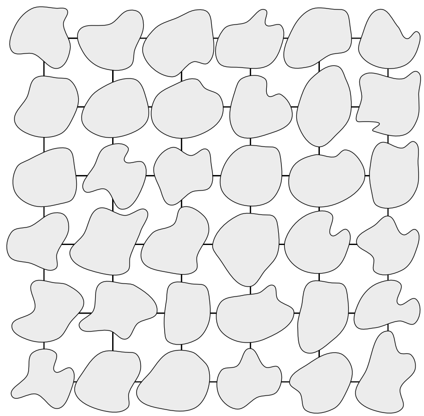
Every MSO_2-definable problem solvable in linear time

$\mathscr{C}$ has **unbnd treewidth** $\Rightarrow$

$\mathscr{C}$ contains arbitrarily large grid minors $\Rightarrow^{\star}$

Model-checking MSO_2 on colored graphs from $\mathscr{C}$

is as **hard** as on general graphs.



Complexity of MSO_2 model-checking

Consider a class of graphs $\mathscr{C}$.

$\mathscr{C}$ has **bnd treewidth** $\Rightarrow$

Every MSO_2-definable problem solvable in linear time

$\mathscr{C}$ has **unbnd treewidth** $\Rightarrow$

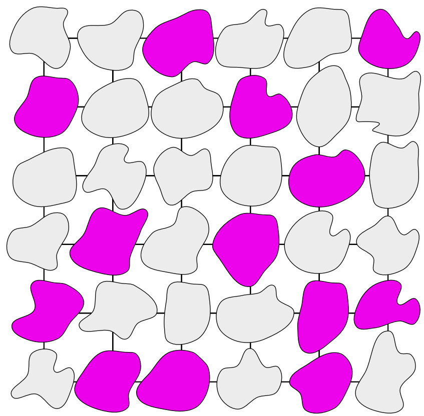
$\mathscr{C}$ contains arbitrarily large grid minors $\Rightarrow^{\star}$

Model-checking MSO_2 on colored graphs from $\mathscr{C}$

is as **hard** as on general graphs.

Reason: Encode an arbitrary

adjacency matrix in a grid minor.



Dichotomy

Dichotomy

Consider a graph class $\mathscr{C}$. Then either:

Dichotomy

Consider a graph class $\mathscr{C}$. Then either:

$\mathscr{C}$ has **bounded treewidth**.

MSO$_2$ model-checking on colored $\mathscr{C}$ in **time** $f(\varphi) \cdot n$.

In colored $\mathscr{C}$ one can MSO$_2$-interpret **only tree-like graphs**.

$\vdots$

$\mathscr{C}$ has arbitrarily **large grid minors.**

MSO$_2$ model-checking on colored $\mathscr{C}$ as **hard** as on general graphs.

In colored $\mathscr{C}$ one can MSO$_2$-interpret **all graphs**.

$\vdots$

Dichotomy

Consider a graph class $\mathscr{C}$. Then either:

graph theory

$\mathscr{C}$ has **bounded treewidth**.

MSO$_2$ model-checking on colored $\mathscr{C}$ in **time** $f(\varphi) \cdot n$.

In colored $\mathscr{C}$ one can MSO$_2$-interpret **only tree-like graphs**.

$\vdots$

$\mathscr{C}$ has arbitrarily **large grid minors.**

MSO$_2$ model-checking on colored $\mathscr{C}$ as **hard** as on general graphs.

In colored $\mathscr{C}$ one can MSO$_2$-interpret **all graphs**.

$\vdots$

Dichotomy

Consider a graph class $\mathscr{C}$. Then either:

graph theory

$\mathscr{C}$ has **bounded treewidth**.

$\mathscr{C}$ has arbitrarily **large grid minors.**

algorithms

MSO_2 model-checking on colored $\mathscr{C}$ in **time** $f(\varphi) \cdot n$.

MSO_2 model-checking on colored $\mathscr{C}$ as **hard** as on general graphs.

In colored $\mathscr{C}$ one can MSO_2-interpret **only tree-like graphs**.

In colored $\mathscr{C}$ one can MSO_2-interpret **all graphs**.

$\vdots$

$\vdots$

Dichotomy

Consider a graph class $\mathscr{C}$. Then either:

graph theory

$\mathscr{C}$ has **bounded treewidth**. | $\mathscr{C}$ has arbitrarily **large grid minors.**

algorithms

MSO_2 model-checking on colored $\mathscr{C}$ in **time** $f(\varphi) \cdot n$. | MSO_2 model-checking on colored $\mathscr{C}$ as **hard** as on general graphs.

finite model theory

In colored $\mathscr{C}$ one can MSO_2-interpret **only tree-like graphs**. | In colored $\mathscr{C}$ one can MSO_2-interpret **all graphs**.

$\vdots$ | $\vdots$

Other logic

Other logic

$\text{MSO}_2 \longleftrightarrow$ bnd treewidth

Other logic

$$\begin{tikzcd} \mathrm{MSO}_2 \arrow[r, leftrightarrow, dashed] \arrow[d, phantom, "\cup" description] & \text{bnd treewidth} \arrow[d, phantom, "\cap" description] \\ \mathrm{MSO}_1 \arrow[r, leftrightarrow, dashed] & \text{bnd cliquewidth} \end{tikzcd}$$

Other logic

$$\begin{array}{ccc} \mathrm{MSO}_2 & \longleftrightarrow & \text{bnd treewidth} \\ \cup & & \cap \\ \mathrm{MSO}_1 & \longleftrightarrow & \text{bnd cliquewidth} \\ \cup & & \\ \mathrm{FO} & & ?? \end{array}$$

Other logic

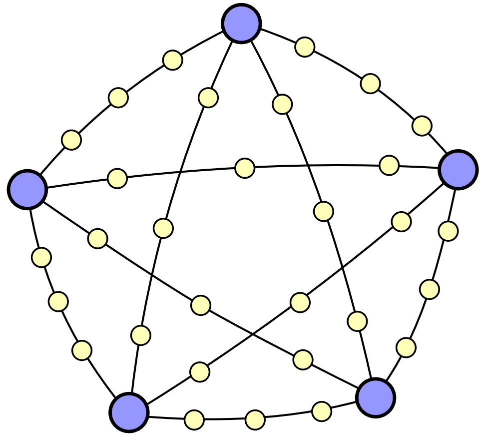
$$\begin{array}{cccc} \mathrm{MSO}_2 & \longleftrightarrow & & \text{bnd treewidth} \\ \cup & & & \cap \\ \mathrm{MSO}_1 & \longleftrightarrow & & \text{bnd cliquewidth} \\ \cup & & & \\ \mathrm{FO} & \longleftrightarrow & \text{nowhere dense} & ?? \end{array}$$

Nowhere denseness

Nowhere denseness

Definition (Nowhere denseness)

A class of graphs $\mathscr{C}$ is **nowhere dense** if for every $d \in \mathbb{N}$ there is $t(d)$ such that graphs from $\mathscr{C}$ **exclude** the d-**subdivision** of the **clique** $K_{t(d)}$.



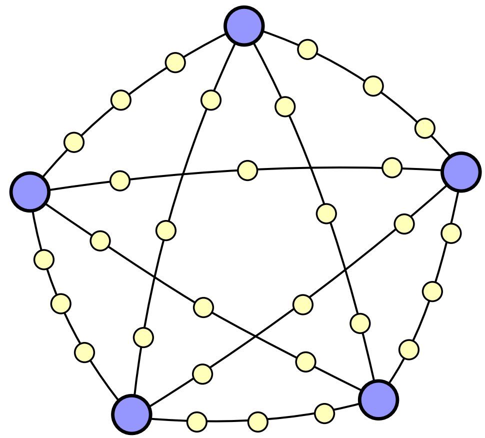
Nowhere denseness

Definition (Nowhere denseness)

A class of graphs $\mathscr{C}$ is **nowhere dense** if for every $d \in \mathbb{N}$ there is $t(d)$ such that graphs from $\mathscr{C}$ **exclude** the d-**subdivision** of the **clique** $K_{t(d)}$.

Examples:

- planar graphs;
- graphs with a fixed excluded minor;
- graphs with maximum degree $\leqslant 15$;
- graphs of treewidth $\leqslant 15$.



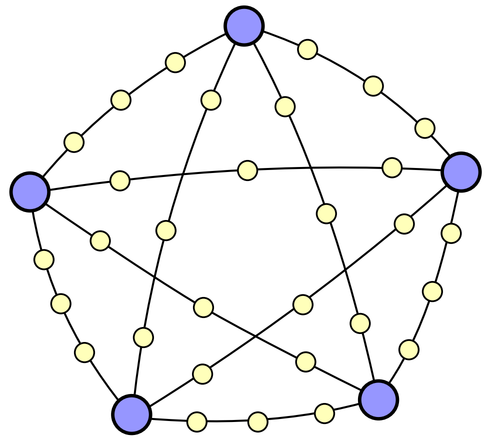
Nowhere denseness

Definition (Nowhere denseness)

A class of graphs $\mathscr{C}$ is **nowhere dense** if for every $d \in \mathbb{N}$ there is $t(d)$ such that graphs from $\mathscr{C}$ **exclude** the ***d*-subdivision** of the **clique** $K_{t(d)}$.

Examples:

- planar graphs;
- graphs with a fixed excluded minor;
- graphs with maximum degree $\leqslant 15$;
- graphs of treewidth $\leqslant 15$.



Theorem (Grohe, Kreutzer, Siebertz)

Suppose $\mathscr{C}$ is a class of graphs closed under taking subgraphs. Then:

- $\mathscr{C}$ **nowhere dense** $\Rightarrow$ MC FO in time $f(\varphi) \cdot n^{1+\varepsilon}$ for any $\varepsilon > 0$.
- $\mathscr{C}$ **somewhere dense** $\Rightarrow$ MC FO as hard as on general graphs.

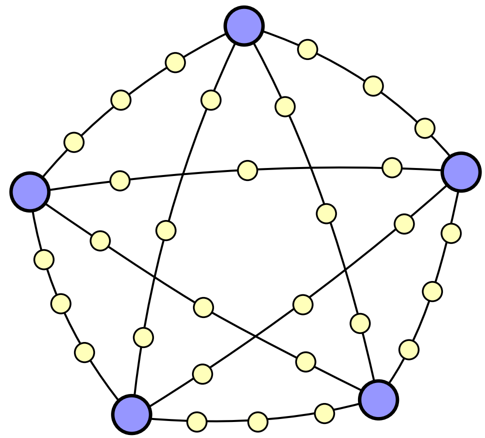
Nowhere denseness

Definition (Nowhere denseness)

A class of graphs $\mathscr{C}$ is **nowhere dense** if for every $d \in \mathbb{N}$ there is $t(d)$ such that graphs from $\mathscr{C}$ **exclude** the d-**subdivision** of the **clique** $K_{t(d)}$.

Examples:

- planar graphs;
- graphs with a fixed excluded minor;
- graphs with maximum degree $\leqslant 15$;
- graphs of treewidth $\leqslant 15$.



Theorem (Grohe, Kreutzer, Siebertz)

Suppose $\mathscr{C}$ is a class of graphs closed under taking subgraphs. Then:

- $\mathscr{C}$ **nowhere dense** $\Rightarrow$ MC FO in time $f(\varphi) \cdot n^{1+\varepsilon}$ for any $\varepsilon > 0$.
- $\mathscr{C}$ **somewhere dense** $\Rightarrow$ MC FO as hard as on general graphs.

Towards a characterization for FO

Towards a characterization for FO

The theorem of GKS originates from the theory of **Sparsity**.

Towards a characterization for FO

The theorem of GKS originates from the theory of **Sparsity**.

– A wealth of **graph-theoretic** tools for nowhere dense classes.

Towards a characterization for FO

The theorem of GKS originates from the theory of **Sparsity**.

– A wealth of **graph-theoretic** tools for nowhere dense classes.

There are classes of **dense** graphs where FO model-checking is tractable.

Towards a characterization for FO

The theorem of GKS originates from the theory of **Sparsity**.

- A wealth of **graph-theoretic** tools for nowhere dense classes.

There are classes of **dense** graphs where FO model-checking is tractable.

- cliques;

Towards a characterization for FO

The theorem of GKS originates from the theory of **Sparsity**.

– A wealth of **graph-theoretic** tools for nowhere dense classes.

There are classes of **dense** graphs where FO model-checking is tractable.

– cliques;

– for any nowhere dense $\mathscr{C}$, edge-complements of $\mathscr{C}$;

Towards a characterization for FO

The theorem of GKS originates from the theory of **Sparsity**.

– A wealth of **graph-theoretic** tools for nowhere dense classes.

There are classes of **dense** graphs where FO model-checking is tractable.

– cliques;
– for any nowhere dense $\mathscr{C}$, edge-complements of $\mathscr{C}$;
– any class of bounded cliquewidth.

Towards a characterization for FO

The theorem of GKS originates from the theory of **Sparsity**.

- A wealth of **graph-theoretic** tools for nowhere dense classes.

There are classes of **dense** graphs where FO model-checking is tractable.

- cliques;
- for any nowhere dense $\mathscr{C}$, edge-complements of $\mathscr{C}$;
- any class of bounded cliquewidth.

Q: Proposition for ?? in the following:

$$\mathrm{FO} \longleftrightarrow \;??$$

Towards a characterization for FO

The theorem of GKS originates from the theory of **Sparsity**.

- A wealth of **graph-theoretic** tools for nowhere dense classes.

There are classes of **dense** graphs where FO model-checking is tractable.

- cliques;
- for any nowhere dense $\mathscr{C}$, edge-complements of $\mathscr{C}$;
- any class of bounded cliquewidth.

Q: Proposition for ?? in the following:

$$\mathrm{FO} \quad \longleftrightarrow \quad ??$$

Definition (Monadic dependence)

A class of graphs $\mathscr{C}$ is **monadically dependent** if one cannot FO-interpret all graphs in colored graphs from $\mathscr{C}$.

Monadic dependence

Monadic dependence

(Monadically) dependent (NIP) and **(monadically) stable** theories are a central object of studies in **model theory**.

Monadic dependence

(Monadically) dependent (NIP) and **(monadically) stable** theories are a central object of studies in **model theory**.

– There is a toolbox...

Monadic dependence

(Monadically) dependent (NIP) and **(monadically) stable** theories are a central object of studies in **model theory**.

- There is a toolbox... but applies to an inherently infinite setting.

Monadic dependence

(Monadically) dependent (NIP) and **(monadically) stable** theories are a central object of studies in **model theory**.

- There is a toolbox... but applies to an inherently infinite setting.
- For example, **nowhere dense** = **superflat**.

Monadic dependence

(Monadically) dependent (NIP) and **(monadically) stable** theories are a central object of studies in **model theory**.

- There is a toolbox... but applies to an inherently infinite setting.
- For example, **nowhere dense** = **superflat**.

Goal: Understand it and apply the toolbox to classes of finite graphs.

Monadic dependence

(Monadically) dependent (NIP) and **(monadically) stable** theories are a central object of studies in **model theory**.

- There is a toolbox... but applies to an inherently infinite setting.
- For example, **nowhere dense** = **superflat**.

Goal: Understand it and apply the toolbox to classes of finite graphs.

So far...

Intermediate logic

$$\begin{array}{ccc} \mathrm{MSO}_2 & \longleftrightarrow & \text{bnd treewidth} \\ \cup & & \cap \\ \mathrm{MSO}_1 & \longleftrightarrow & \text{bnd cliquewidth} \\ \cup & & \\ \mathrm{FO} & \longleftrightarrow & \text{nowhere dense} \end{array}$$

Intermediate logic

Can we find natural variants of logic between FO and MSO?

$$
\begin{array}{ccc}
\mathrm{MSO}_2 & \longleftrightarrow & \text{bnd treewidth} \\
\cup & & \cap \\
\mathrm{MSO}_1 & \longleftrightarrow & \text{bnd cliquewidth} \\
\cup & & \\
\mathrm{FO} & \longleftrightarrow & \text{nowhere dense}
\end{array}
$$

Intermediate logic

Can we find natural variants of logic between FO and MSO?

$$
\begin{array}{cccc}
\mathrm{MSO}_2 & \longleftrightarrow & & \text{bnd treewidth} \\
\cup & & & \cap \\
\mathrm{MSO}_1 & \longleftrightarrow & & \text{bnd cliquewidth} \\
\cup & & & \\
\mathrm{FO} + \mathrm{conn} & \longleftrightarrow & \text{top-minor-free} & \\
\cup & & \cap & \\
\mathrm{FO} & \longleftrightarrow & \text{nowhere dense} &
\end{array}
$$

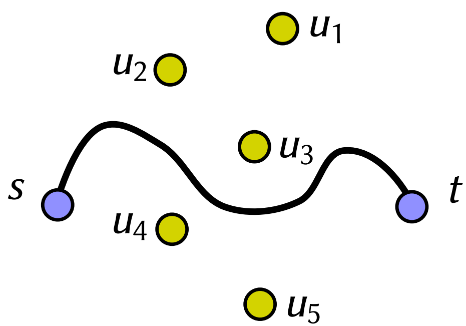
Proposition: logic $\mathrm{FO} + \mathrm{conn}$ corresponds to **topological-minor-free classes**.

FO + conn

FO + conn

Def: $G \models \mathrm{conn}(s, t, u_1, \ldots, u_k)$

if s and t can be **connected** by a **path** avoiding $u_1, \ldots, u_k$.

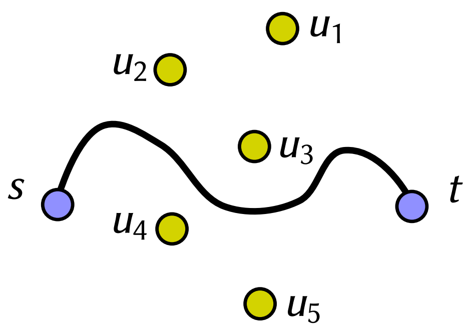


FO + conn

Def: $G \models \mathrm{conn}(s, t, u_1, \ldots, u_k)$

if s and t can be **connected** by a **path** avoiding $u_1, \ldots, u_k$.

FO + conn is FO where you can also use conn predicates.



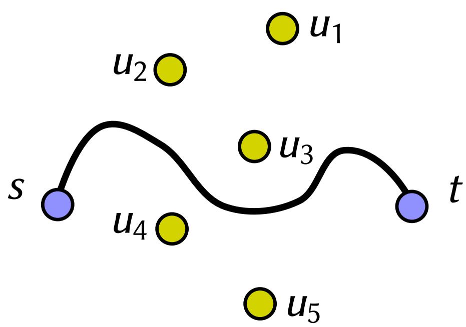
FO + conn

Def: $G \models \mathrm{conn}(s, t, u_1, \ldots, u_k)$

if s and t can be **connected** by a **path** avoiding $u_1, \ldots, u_k$.

FO + conn is FO where you can also use conn predicates.

See [Bojańczyk; 21^+] and [Schirrmacher, Siebertz, Vigny; **this CSL!**].



FO + conn

Def: $G \models \mathrm{conn}(s, t, u_1, \ldots, u_k)$

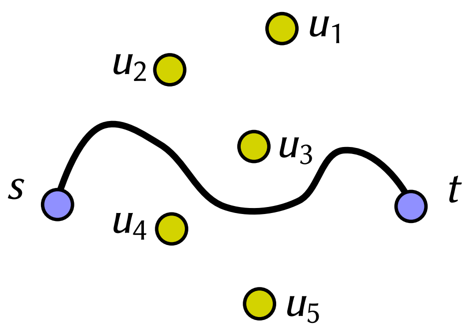
if s and t can be **connected** by a **path** avoiding $u_1, \ldots, u_k$.

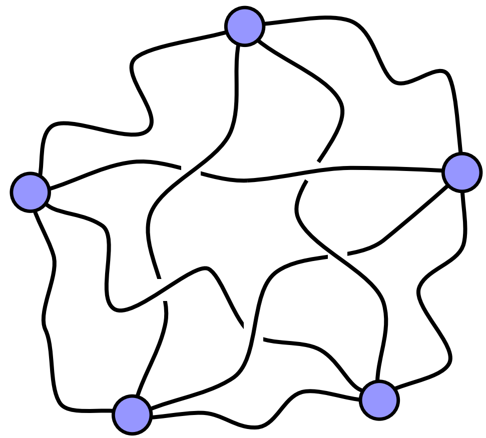
FO + conn is FO where you can also use conn predicates.

See [Bojańczyk; 21$^+$] and [Schirrmacher, Siebertz, Vigny; **this CSL!**].

Def: $\mathscr{C}$ is **top-minor-free** if there is $t \in \mathbb{N}$ such that

no graph $G \in \mathscr{C}$ contains a subdivision of K_t.





FO + conn

Def: $G \models \mathrm{conn}(s, t, u_1, \ldots, u_k)$

if s and t can be **connected** by a **path** avoiding $u_1, \ldots, u_k$.

FO + conn is FO where you can also use conn predicates.

See [Bojańczyk; 21^+] and [Schirrmacher, Siebertz, Vigny; **this CSL!**].

Def: $\mathscr{C}$ is **top-minor-free** if there is $t \in \mathbb{N}$ such that

no graph $G \in \mathscr{C}$ contains a subdivision of K_t.

Theorem (P, Schirrmacher, Siebertz, Toruńczyk, Vigny; 21^+)

Suppose $\mathscr{C}$ is a class of graphs closed under taking subgraphs. Then:

– $\mathscr{C}$ **top-minor-free** $\Rightarrow$ MC FO + conn in time $f(\varphi) \cdot n$.

– $\mathscr{C}$ **not top-minor-free** $\Rightarrow^\star$ MC FO + conn hard as on general graphs.

FO + conn

Def: $G \models \mathrm{conn}(s, t, u_1, \ldots, u_k)$

if s and t can be **connected** by a **path** avoiding $u_1, \ldots, u_k$.

FO + conn is FO where you can also use conn predicates.

See [Bojańczyk; 21$^+$] and [Schirrmacher, Siebertz, Vigny; **this CSL!**].

Def: $\mathscr{C}$ is **top-minor-free** if there is $t \in \mathbb{N}$ such that

no graph $G \in \mathscr{C}$ contains a subdivision of K_t.

Theorem (P, Schirrmacher, Siebertz, Toruńczyk, Vigny; 21$^+$)

Suppose $\mathscr{C}$ is a class of graphs closed under taking subgraphs. Then:

– $\mathscr{C}$ **top-minor-free** $\Rightarrow$ MC FO + conn in time $f(\varphi) \cdot n$.

– $\mathscr{C}$ **not top-minor-free** $\Rightarrow^\star$ MC FO + conn hard as on general graphs.

Key: A known decomposition into parts where FO + conn reduces to FO.

Wrap-up

Wrap-up

Start: questions originating from **graph algorithms**.

Wrap-up

Start: questions originating from **graph algorithms**.

We formulated them in **logic**. $\rightsquigarrow$ Complexity of **model-checking**.

Wrap-up

Start: questions originating from **graph algorithms**.

We formulated them in **logic**. $\rightsquigarrow$ Complexity of **model-checking**.

For MSO_2, a fundamental division line in **graph theory**

$\rightsquigarrow$ division lines in **complexity** and in **finite model theory**.

Wrap-up

Start: questions originating from **graph algorithms**.

We formulated them in **logic**. $\rightsquigarrow$ Complexity of **model-checking**.

For MSO_2, a fundamental division line in **graph theory**

$\rightsquigarrow$ division lines in **complexity** and in **finite model theory**.

For FO, a **logic**-motivated branch of **graph theory**: **Sparsity**.

Wrap-up

Start: questions originating from **graph algorithms**.

We formulated them in **logic**. $\rightsquigarrow$ Complexity of **model-checking**.

For MSO_2, a fundamental division line in **graph theory**

$\rightsquigarrow$ division lines in **complexity** and in **finite model theory**.

For FO, a **logic**-motivated branch of **graph theory**: **Sparsity**.

Beyond subgraph-closed classes $\rightsquigarrow$ Notions from **model theory**.

Wrap-up

Start: questions originating from **graph algorithms**.

We formulated them in **logic**. $\rightsquigarrow$ Complexity of **model-checking**.

For MSO_2, a fundamental division line in **graph theory**

$\rightsquigarrow$ division lines in **complexity** and in **finite model theory**.

For FO, a **logic**-motivated branch of **graph theory**: **Sparsity**.

Beyond subgraph-closed classes $\rightsquigarrow$ Notions from **model theory**.

A fundamental puzzle with many facets.

Wrap-up

Start: questions originating from **graph algorithms**.

We formulated them in **logic**. $\rightsquigarrow$ Complexity of **model-checking**.

For MSO_2, a fundamental division line in **graph theory**

$\rightsquigarrow$ division lines in **complexity** and in **finite model theory**.

For FO, a **logic**-motivated branch of **graph theory**: **Sparsity**.

Beyond subgraph-closed classes $\rightsquigarrow$ Notions from **model theory**.

A fundamental puzzle with many facets.

Understanding the puzzle requires understanding all the facets.

Wrap-up

Start: questions originating from **graph algorithms**.

We formulated them in **logic**. $\rightsquigarrow$ Complexity of **model-checking**.

For MSO_2, a fundamental division line in **graph theory**

$\rightsquigarrow$ division lines in **complexity** and in **finite model theory**.

For FO, a **logic**-motivated branch of **graph theory**: **Sparsity**.

Beyond subgraph-closed classes $\rightsquigarrow$ Notions from **model theory**.

A fundamental puzzle with many facets.

Understanding the puzzle requires understanding all the facets.

Mathematics is not only about **depth**, but also about **connections**.

Wrap-up

Start: questions originating from **graph algorithms**.

We formulated them in **logic**. $\rightsquigarrow$ Complexity of **model-checking**.

For MSO_2, a fundamental division line in **graph theory**

$\rightsquigarrow$ division lines in **complexity** and in **finite model theory**.

For FO, a **logic**-motivated branch of **graph theory**: **Sparsity**.

Beyond subgraph-closed classes $\rightsquigarrow$ Notions from **model theory**.

A fundamental puzzle with many facets.

Understanding the puzzle requires understanding all the facets.

Mathematics is not only about **depth**, but also about **connections**.

— Don't be afraid to try multiple new areas.

Wrap-up

Start: questions originating from **graph algorithms**.

We formulated them in **logic**. $\rightsquigarrow$ Complexity of **model-checking**.

For MSO_2, a fundamental division line in **graph theory**

$\rightsquigarrow$ division lines in **complexity** and in **finite model theory**.

For FO, a **logic**-motivated branch of **graph theory**: **Sparsity**.

Beyond subgraph-closed classes $\rightsquigarrow$ Notions from **model theory**.

A fundamental puzzle with many facets.

Understanding the puzzle requires understanding all the facets.

Mathematics is not only about **depth**, but also about **connections**.

– Don't be afraid to try multiple new areas.

– Learning by trying.

Wrap-up

Start: questions originating from **graph algorithms**.

We formulated them in **logic**. $\rightsquigarrow$ Complexity of **model-checking**.

For MSO_2, a fundamental division line in **graph theory**

$\rightsquigarrow$ division lines in **complexity** and in **finite model theory**.

For FO, a **logic**-motivated branch of **graph theory**: **Sparsity**.

Beyond subgraph-closed classes $\rightsquigarrow$ Notions from **model theory**.

A fundamental puzzle with many facets.

Understanding the puzzle requires understanding all the facets.

Mathematics is not only about **depth**, but also about **connections**.

– Don't be afraid to try multiple new areas.

– Learning by trying.

Thanks for attention!