Towards a Structural Characterisation of the Complexity of Model-Checking Problems

Stephan Kreutzer

Technical University Berlin

Fall School on Logic and Complexity
19-23 September 2011
Prague

Finite Model Theory

This talk is motivated by applications of logic in computer science, in particular to computational complexity and algorithmic graph theory.

Finite Model Theory. We are interested in definability and model-checking in classes of finite structures.

Finite structures:
databases, transition systems, finite graphs as models in algorithms, ...

Classes of structures:

- We are interested in uniform definability in classes of structures, e.g. is a query definable within the class of all databases, etc.
- Similarly, we will study the problem of evaluating a formula within a class of structures, e.g. the class of all databases, the class of all finte graphs, etc.

Proviso. All structures in this talk will be finite (unless said otherwise)

Finite Model Theory

This talk is motivated by applications of logic in computer science, in particular to computational complexity and algorithmic graph theory.

Finite Model Theory. We are interested in definability and model-checking in classes of finite structures.

Finite structures:
databases, transition systems, finite graphs as models in algorithms, ...

Classes of structures:

- We are interested in uniform definability in classes of structures, e.g. is a query definable within the class of all databases, etc.
- Similarly, we will study the problem of evaluating a formula within a class of structures, e.g. the class of all databases, the class of all finte graphs, etc.

Proviso. All structures in this talk will be finite (unless said otherwise)

Model-Checking

In this talk we are primarily interested in evaluating formulas of a logic $\mathcal{L}$ in classes $\mathcal{C}$ of finite structures.

The Model-Checking Problem $\mathbf{MC}(\mathcal{L}, \mathcal{C})$*:*

Given:	Finite structure $\mathfrak{A} := (A, \sigma) \in \mathcal{C}$ Formula $\varphi \in \mathcal{L}$
Problem:	Decide $\mathfrak{A} \models \varphi$?

Note. In this talk we will only consider model-checking for formulas without free variables.

We write $\mathrm{MC}(\mathcal{L})$ if $\mathcal{C}$ is the class of all structures over some signature.

Applications.

Verification.	Model-checking is widely studied in computer-aided verification, where mostly temporal logics are used.
Databases.	Efficient evaluation of formulas/database queries.
Complexity Theory.	Formulas describe computational problems.

A Connection to Complexity Theory

Many standard computational problems on graphs are NP-complete, e.g.

- Dominating Set (find a min. set of vertices neighbours to all others)
- 3-Colourability (3-colour a graph without monochromatic edges)
- Hamiltonian path (find a path containing every vertex exactly once)

Study classes of graphs (planar graphs, graphs of bounded genus, ...) on which some of these problems become tractable.

Logical approach. Instead of designing algorithms for each problem individually, formulate the problems in a logical language and design model-checking algorithms on these classes of graphs.

A Connection to Complexity Theory

Many standard computational problems on graphs are NP-complete, e.g.

- Dominating Set (find a min. set of vertices neighbours to all others)
- 3-Colourability (3-colour a graph without monochromatic edges)
- Hamiltonian path (find a path containing every vertex exactly once)

Study classes of graphs (planar graphs, graphs of bounded genus, ...) on which some of these problems become tractable.

Logical approach. Instead of designing algorithms for each problem individually, formulate the problems in a logical language and design model-checking algorithms on these classes of graphs.

A Connection to Complexity Theory

Many standard computational problems on graphs are NP-complete, e.g.

- Dominating Set (find a min. set of vertices neighbours to all others)

  (Parametrized) definable in First-Order Logic (FO)
- 3-Colourability (3-colour a graph without monochromatic edges)

  definable in Monadic Second-Order Logic (MSO)
- Hamiltonian path (find a path containing every vertex exactly once)

  definable in Monadic Second-Order Logic (MSO_2)

Study classes of graphs (planar graphs, graphs of bounded genus, ...) on which some of these problems become tractable.

Logical approach. Instead of designing algorithms for each problem individually, formulate the problems in a logical language and design model-checking algorithms on these classes of graphs.

Model-Checking

Monadic Second-Order Logic.

First-Order Logic by quantification over sets of elements.

Formula building rules.

- $\exists X\,\varphi$, $\forall X\,\varphi$: there is a/for all sets of elements φ holds
- $\exists x\,\varphi$, $\forall x\,\varphi$: there is an/for all elements φ holds
- Boolean connectives and atomic formulas

Example. In the language $\sigma := \{E\}$ of graphs $G := (V, E)$ we can write

$$\underbrace{\exists C_1 \exists C_2 \exists C_3}_{\substack{\text{there are sets}\\ C_1, C_2, C_3}} \Big(\underbrace{\forall x \bigvee_{i=1}^{3} C_i(x)}_{\text{ev. node has a col.}} \wedge \underbrace{\forall x \forall y \big(E(x,y) \rightarrow \bigwedge_{i=1}^{3} \neg(C_i(x) \wedge C_i(y))\big)}_{\text{endpoints of edges have different colours}} \Big)$$

to say that a graph is 3-colourable.

Monadic Second-Order Logic on Graphs

There is a subtlety in how we encode graphs as logical structures.

Standard encoding. Signature $\sigma_g := \{E\}$.

A graph $G := (V, E)$ is encoded as σ_g-structure $\mathcal{G} := (V, E)$

Incidence encoding. Signature $\sigma_i := \{V, E, inc\}$.

A graph $G := (V, E)$ is encoded as σ_i-structure $\mathcal{G} := (V \cup E, \sigma)$ with

- $V^{\mathcal{G}} := V(G)$, $E^{\mathcal{G}} := E(G)$ and
- $(x, e) \in inc^{\mathcal{G}}$ if the vertex $x \in V(G)$ is incident to edge $e \in E(G)$

Over the incidence encoding we can say in MSO that a graph has a Hamiltonian cycle, which we cannot say in the standard encoding.

$$\exists P \subseteq E \big(P \text{ forms a path and every vertex occurs exactly once on } P\big)$$

We will refer to MSO as MSO_2 whenever we mean the incidence encoding and to MSO if we use the standard encoding.

Monadic Second-Order Logic on Graphs

There is a subtlety in how we encode graphs as logical structures.

Standard encoding. Signature $\sigma_g := \{E\}$.

A graph $G := (V, E)$ is encoded as σ_g-structure $\mathcal{G} := (V, E)$

Incidence encoding. Signature $\sigma_i := \{V, E, inc\}$.

A graph $G := (V, E)$ is encoded as σ_i-structure $\mathcal{G} := (V \cup E, \sigma)$ with

- $V^{\mathcal{G}} := V(G)$, $\quad E^{\mathcal{G}} := E(G)$ and
- $(x, e) \in inc^{\mathcal{G}}$ if the vertex $x \in V(G)$ is incident to edge $e \in E(G)$

Over the incidence encoding we can say in MSO that a graph has a Hamiltonian cycle, which we cannot say in the standard encoding.

$$\exists P \subseteq E\big(P \text{ forms a path and every vertex occurs exactly once on } P\big)$$

We will refer to MSO as MSO_2 whenever we mean the incidence encoding and to MSO if we use the standard encoding.

Complexity of Model-Checking Problems

Complexity of Monadic Second-Order Model-Checking

Given: Finite structure $\mathfrak{A} := (A, \sigma)$
MSO-formula φ
Problem: Decide $\mathfrak{A} \models \varphi$

Naïve algorithm: Evaluation following the structure of the formula

- Existential second-order quantification: $\varphi := \exists X \psi$
  for all $U \subseteq A$ check whether $(\mathfrak{A}, X \mapsto U) \models \psi$
- Existential first-order quantification: $\varphi := \exists x \psi$
  for all $a \in A$ check whether $(\mathfrak{A}, x \mapsto a)\psi$
- Boolean connectives $\wedge, \vee, \neg$: easy
- Atomic formulae: direct look up in the structure

Running time and space:

Time: exponential in $|\varphi|$ and $|A|$
Space: linear in both $|\varphi|$ and $|A|$

Complexity of Monadic Second-Order Model-Checking

Given: Finite structure $\mathfrak{A} := (A, \sigma)$
MSO-formula φ
Problem: Decide $\mathfrak{A} \models \varphi$

Naïve algorithm: Evaluation following the structure of the formula

- Existential second-order quantification: $\varphi := \exists X \psi$

  for all $U \subseteq A$ check whether $(\mathfrak{A}, X \mapsto U) \models \psi$
- Existential first-order quantification: $\varphi := \exists x \psi$

  for all $a \in A$ check whether $(\mathfrak{A}, x \mapsto a)\psi$
- Boolean connectives $\wedge, \vee, \neg$: easy
- Atomic formulae: direct look up in the structure

Running time and space:

Time: exponential in $|\varphi|$ and $|A|$

Space: linear in both $|\varphi|$ and $|A|$

Complexity of Monadic Second-Order Model-Checking

Theorem: Monadic Second-Order Model-Checking $\mathrm{MC}(\mathrm{MSO})$ is PSPACE-complete.

This is even true for $\mathrm{MC}(\mathrm{MSO}, \mathfrak{A})$ for a fixed two element structure $\mathfrak{A}$.

Proof. Reduction from satisfiability for Quantified Boolean Formulae

Data complexity.

Study the complexity of evaluating a fixed formula in input structures.

There are fixed formulas in MSO for which model-checking is NP-hard.

Complexity of First-Order Model-Checking

Naïve algorithm gives running time and space:

time:	$\mathcal{O}(l \cdot n^m)$	l: length of φ	m: quantifier rank of φ
space:	$\mathcal{O}(m \cdot \log n)$	n: size of $\mathfrak{A}$	

Theorem: First-Order Model-Checking MC(FO) is PSPACE-complete.

This is even true for $\mathrm{MC}(\mathrm{FO}, \mathfrak{A})$ for a fixed two element structure $\mathfrak{A}$.

Proof. Reduction from satisfiability for Quantified Boolean Formulae

Theorem. For any fixed φ, (data complexity)

$\mathrm{MC}(\varphi, \mathrm{Str}) \in \mathrm{AC}_0 \subseteq \textsc{Logspace} \subseteq \textsc{Ptime}$

However: Running time $\mathcal{O}(l \cdot n^m)$

Complexity of MSO revisited

Theorem: Monadic Second-Order Model-Checking MC(MSO) is PSPACE-complete.

There are fixed formulas in MSO for which model-checking is NP-hard.

On the other hand. For every fixed $\varphi \in \mathrm{MSO}$, deciding whether φ is true in a finite tree given as input can be done in linear time.

More precisely, MC(MSO, TREE) can be solved in time

$$f(|\varphi|) \cdot |T|,$$

where $|T|$ is the size of the tree and f is a computable function.

Hence, by restricting the class of admissible inputs we can achieve much better model-checking results.

Parametrized Complexity

Fixed-Parameter tractability. A model-checking problem is fixed-parameter tractable (fpt) if it can be solved in time

$$f(|\varphi|) \cdot |\mathfrak{A}|^c,$$

where c is a constant and f is a computable function.

Similarly, problems such as Dominating Set are fixed-parameter tractable on a class $\mathcal{C}$ of graphs if on input $G \in \mathcal{C}$ and k it can be decided in time $f(k) \cdot |G|^c$ whether G contains a dominating set of size k.

FPT is the class of all fixed-parameter tractable problems.

Comparable to PTIME in classical complexity.

The rôle of NP is played by a hierarchy of classes W[1], W[2], ...

Parametrized Complexity

Fixed-Parameter tractability. A model-checking problem is fixed-parameter tractable (fpt) if it can be solved in time

$$f(|\varphi|) \cdot |\mathfrak{A}|^c,$$

where c is a constant and f is a computable function.

Similarly, problems such as Dominating Set are fixed-parameter tractable on a class $\mathcal{C}$ of graphs if on input $G \in \mathcal{C}$ and k it can be decided in time $f(k) \cdot |G|^c$ whether G contains a dominating set of size k.

FPT is the class of all fixed-parameter tractable problems.

Comparable to PTIME in classical complexity.

The rôle of NP is played by a hierarchy of classes W[1], W[2], ...

Parametrized Complexity

Fixed-Parameter tractability. A model-checking problem is fixed-parameter tractable (fpt) if it can be solved in time

$$f(|\varphi|) \cdot |\mathfrak{A}|^c,$$

where c is a constant and f is a computable function.

Similarly, problems such as Dominating Set are fixed-parameter tractable on a class $\mathcal{C}$ of graphs if on input $G \in \mathcal{C}$ and k it can be decided in time $f(k) \cdot |G|^c$ whether G contains a dominating set of size k.

FPT is the class of all fixed-parameter tractable problems.

Comparable to PTIME in classical complexity.

The rôle of NP is played by a hierarchy of classes W[1], W[2], ...

Structural Characterisation of Model-Checking Problems

In the terminology of parametrized complexity:

MSO-model-checking is fpt on the class of finite trees.

Question. What are the largest/most general classes of graphs on which MSO becomes tractable?

And the same question applies to first-order logic.

Research programme. For each of the natural logics $\mathcal{L}$ such as FO or MSO, identify a structural property $\mathcal{P}$ of classes $\mathcal{C}$ of graphs such that MC$(\mathcal{L}, \mathcal{C})$ is tractable if, and only if, $\mathcal{C}$ has the property $\mathcal{P}$

We may not always get an exact characterisation, there may be gaps.

But such a characterisation would give an easy tool to assess whether MSO-model-checking is tractable on some class.

Structural Characterisation of Model-Checking Problems

In the terminology of parametrized complexity:

MSO-model-checking is fpt on the class of finite trees.

Question. What are the largest/most general classes of graphs on which MSO becomes tractable?

And the same question applies to first-order logic.

Research programme. For each of the natural logics $\mathcal{L}$ such as FO or MSO, identify a structural property $\mathcal{P}$ of classes $\mathcal{C}$ of graphs such that MC$(\mathcal{L}, \mathcal{C})$ is tractable if, and only if, $\mathcal{C}$ has the property $\mathcal{P}$

We may not always get an exact characterisation, there may be gaps.

But such a characterisation would give an easy tool to assess whether MSO-model-checking is tractable on some class.

Structural Characterisation of Model-Checking Problems

In the terminology of parametrized complexity:

MSO-model-checking is fpt on the class of finite trees.

Question. What are the largest/most general classes of graphs on which MSO becomes tractable?

And the same question applies to first-order logic.

Research programme. For each of the natural logics $\mathcal{L}$ such as FO or MSO, identify a structural property $\mathcal{P}$ of classes $\mathcal{C}$ of graphs such that MC$(\mathcal{L}, \mathcal{C})$ is tractable if, and only if, $\mathcal{C}$ has the property $\mathcal{P}$

We may not always get an exact characterisation, there may be gaps.

But such a characterisation would give an easy tool to assess whether MSO-model-checking is tractable on some class.

Structural Characterisation of Model-Checking Problems

In the terminology of parametrized complexity:

MSO-model-checking is fpt on the class of finite trees.

Question. What are the largest/most general classes of graphs on which MSO becomes tractable?

And the same question applies to first-order logic.

Research programme. For each of the natural logics $\mathcal{L}$ such as FO or MSO, identify a structural property $\mathcal{P}$ of classes $\mathcal{C}$ of graphs such that MC$(\mathcal{L}, \mathcal{C})$ is tractable if, and only if, $\mathcal{C}$ has the property $\mathcal{P}$ under suitable complexity theoretical assumptions.

We may not always get an exact characterisation, there may be gaps.

But such a characterisation would give an easy tool to assess whether MSO-model-checking is tractable on some class.

Structural Characterisation of Model-Checking Problems

To achieve such a characterisation we need

- upper bounds: tractability of model-checking on specific classes of graphs.

  Such results are known as algorithmic meta-theorems

- lower bounds: results establishing intractability of model-checking problems if certain structural parameters are not given.

Structural Characterisation of Model-Checking Problems

To achieve such a characterisation we need

- upper bounds: tractability of model-checking on specific classes of graphs.

  Such results are known as algorithmic meta-theorems

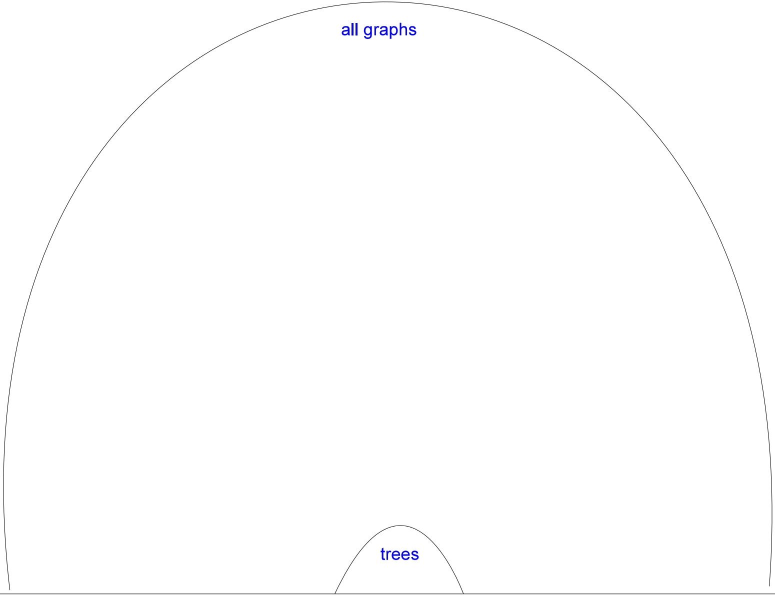
  Part I of this tutorial

- lower bounds: results establishing intractability of model-checking problems if certain structural parameters are not given.

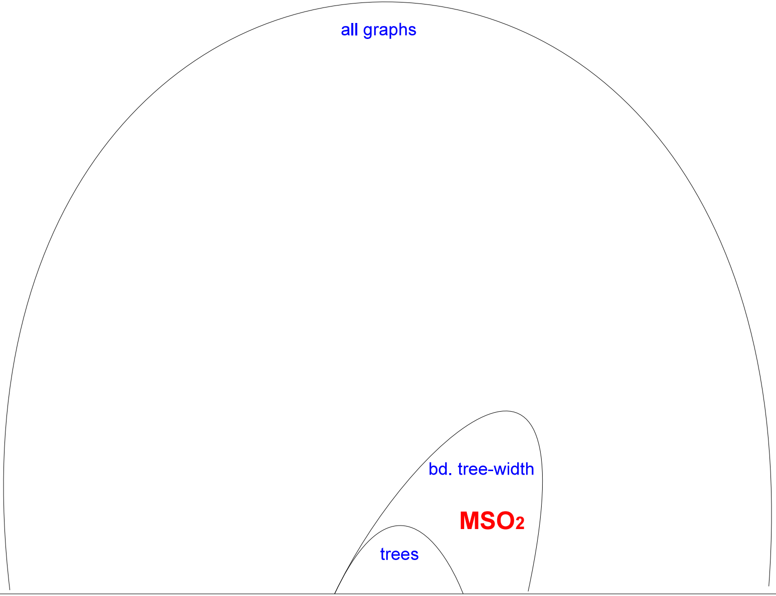
  Part II of this tutorial

Upper Bounds on the Complexity of Model-Checking Problems

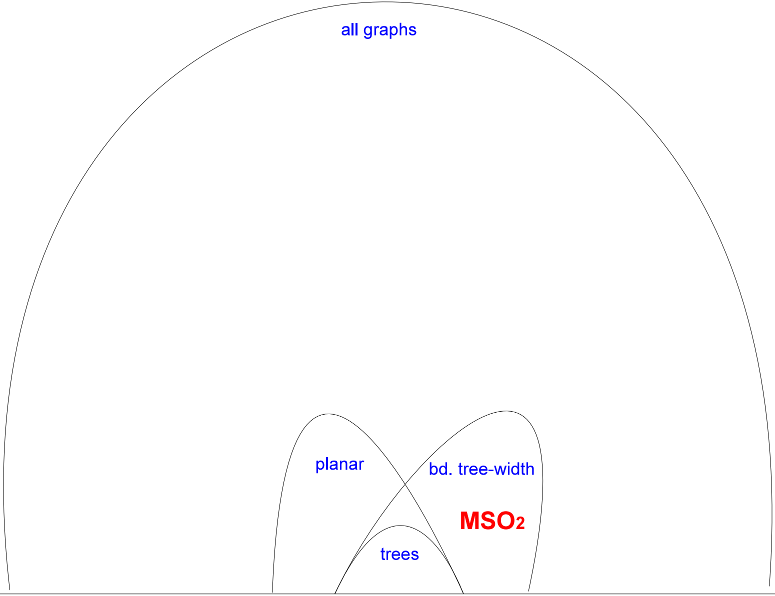
An Overview of Graph Parameters



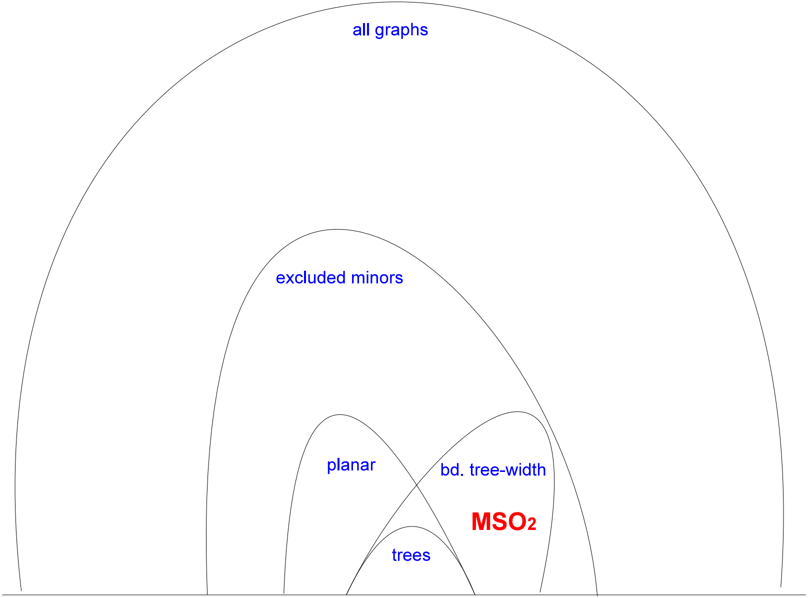
An Overview of Graph Parameters



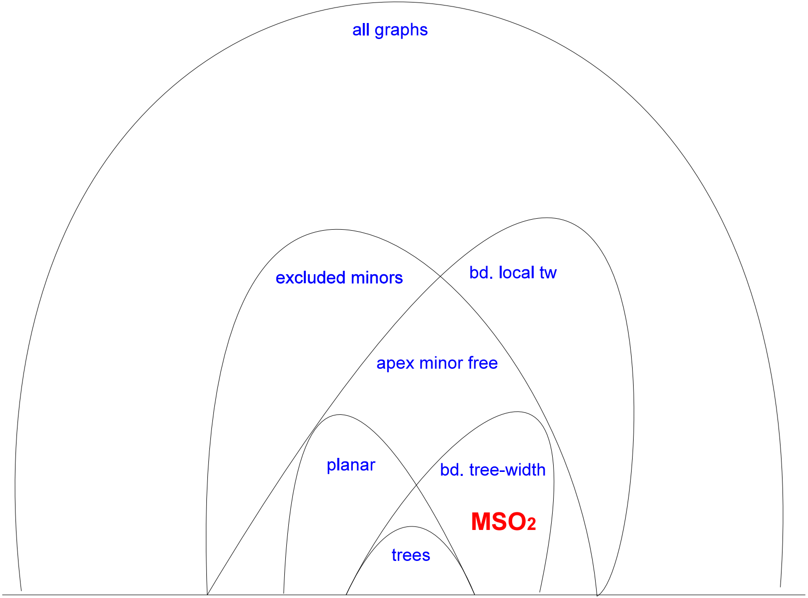
An Overview of Graph Parameters



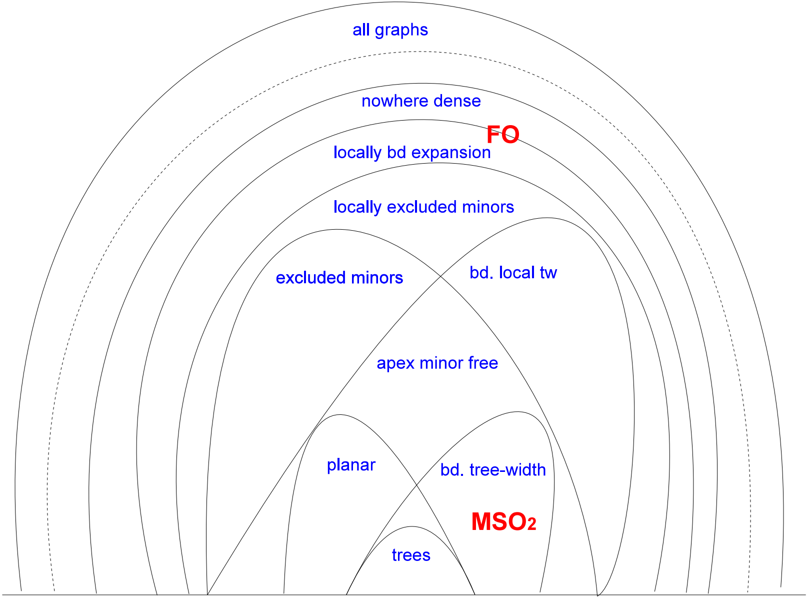
An Overview of Graph Parameters



An Overview of Graph Parameters



An Overview of Graph Parameters



The Composition Method

Feferman-Vaught Style Theorems

Notation:

G: graph $\quad\quad \overline{v}$: tuple of vertices

$\mathrm{tp}^{\mathrm{MSO}}(G, \overline{v})$: full MSO-type of $\overline{v}$ in G (all MSO-formulae true at $\overline{v}$)

$\mathrm{tp}_q^{\mathrm{MSO}}(G, \overline{v})$: class of MSO-formulae of quantifier-rank $\leq q$ true at $\overline{v}$

analogously for $\mathrm{tp}^{\mathrm{FO}}$ and $\mathrm{tp}_q^{\mathrm{FO}}$

Feferman-Vaught Style Theorems

Theorem. Let G, H be graphs

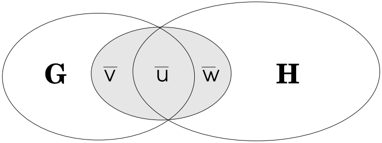
$\overline{v} \in V(G) \qquad \overline{w} \in V(H)$

$\overline{u} \in V(G)$ such that $\overline{u} = V(G) \cap V(H)$

For all $q \geq 0$,

$$\mathrm{tp}_q(G \cup H, \overline{uvw}) \text{ is determined by } \mathrm{tp}_q(G, \overline{uv}) \text{ and } \mathrm{tp}_q(\overline{uw})$$

Furthermore, there is an algorithm that computes $\mathrm{tp}_q(G \cup H, \overline{uvw})$ from $\mathrm{tp}_q(G, \overline{uv})$ and $\mathrm{tp}_q(\overline{uw})$.



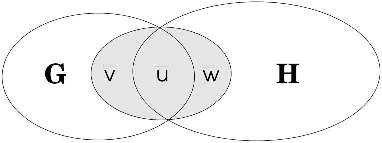
Feferman-Vaught Style Theorems

Theorem. Let G, H be graphs, $\overline{v} \in V(G)$ $\overline{w} \in V(H)$

$\overline{u} \in V(G)$ such that $\overline{u} = V(G) \cap V(H)$

For all $q \geq 0$, $\text{tp}_q(G \cup H, \overline{uvw})$ is determined by $\text{tp}_q(G, \overline{uv})$ and $\text{tp}_q(\overline{uw})$

Furthermore, there is an algorithm that computes $\text{tp}_q(G \cup H, \overline{uvw})$ from $\text{tp}_q(G, \overline{uv})$ and $\text{tp}_q(\overline{uw})$.



Feferman-Vaught Style Theorems

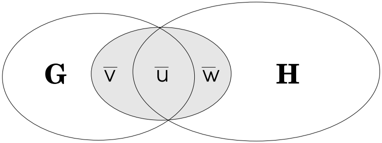
Theorem. Let G, H be graphs, $\overline{v} \in V(G)$ $\overline{w} \in V(H)$

$\overline{u} \in V(G)$ such that $\overline{u} = V(G) \cap V(H)$

For all $q \geq 0$, $\text{tp}_q(G \cup H, \overline{uvw})$ is determined by $\text{tp}_q(G, \overline{uv})$ and $\text{tp}_q(\overline{uw})$

Furthermore, there is an algorithm that computes $\text{tp}_q(G \cup H, \overline{uvw})$ from $\text{tp}_q(G, \overline{uv})$ and $\text{tp}_q(\overline{uw})$.

This suggests a model-checking algorithm on graphs which can recursively be decomposed into sub-graphs of constant size.



Feferman-Vaught Style Theorems

Theorem. Let G, H be graphs, $\overline{v} \in V(G)$ $\overline{w} \in V(H)$

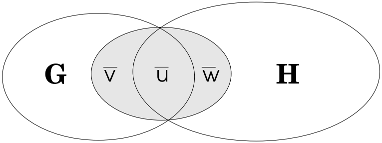
$\overline{u} \in V(G)$ such that $\overline{u} = V(G) \cap V(H)$

For all $q \geq 0$, $\mathrm{tp}_q(G \cup H, \overline{uvw})$ is determined by $\mathrm{tp}_q(G, \overline{uv})$ and $\mathrm{tp}_q(\overline{uw})$

Furthermore, there is an algorithm that computes $\mathrm{tp}_q(G \cup H, \overline{uvw})$ from $\mathrm{tp}_q(G, \overline{uv})$ and $\mathrm{tp}_q(\overline{uw})$.

This suggests a model-checking algorithm on graphs which can recursively be decomposed into sub-graphs of constant size.

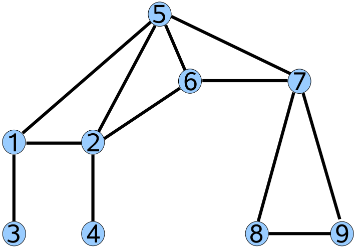
graphs of bounded tree-width



Tree-Width

The tree-width of a graph measures its similarity to a tree.

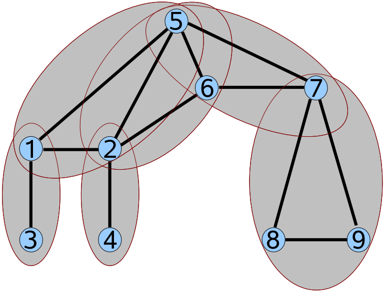
A graph has tree-width $\leq k$ if it can be covered by sub-graphs of size $\leq (k+1)$ in a tree-like fashion.



Tree-Width

The tree-width of a graph measures its similarity to a tree.

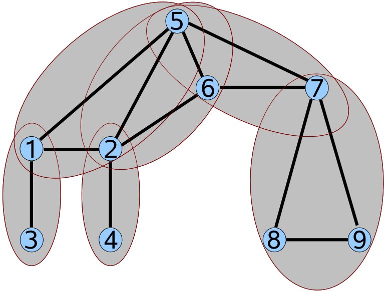
A graph has tree-width $\leq k$ if it can be covered by sub-graphs of size $\leq (k+1)$ in a tree-like fashion.

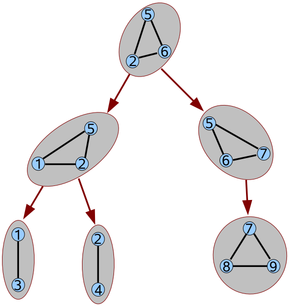


Tree-Width

The tree-width of a graph measures its similarity to a tree.

A graph has tree-width $\leq k$ if it can be covered by sub-graphs of size $\leq (k+1)$ in a tree-like fashion.





Tree-Width

Definition:

A **tree-decomposition** of a graph G is a pair $\mathcal{T} := (T, (B_t)_{t \in V^T})$ where

- T is a (directed) tree
- $B_t \subseteq V(G)$ for all $t \in V^T$

such that

1. for every edge $\{u, v\} \in E(G)$ there is $t \in V(T)$ with $u, v \in B_t$
2. for all $v \in V(G)$ the set $\{t : v \in B_t\}$ is non-empty and connected.

The **width** of $\mathcal{T}$ is $\max\{|B_t| - 1 : t \in V(T)\}$

The **tree-width** $\mathrm{tw}(G)$ of G is the minimal width of any of its tree-dec.

Definition: A class $\mathcal{C}$ has bounded tree-width if there is a constant $k \in \mathbb{N}$ such that $\mathrm{tw}(G) \leq k$ for all $G \in \mathcal{C}$.

Tree-Width

Definition:

A **tree-decomposition** of a graph G is a pair $\mathcal{T} := (T, (B_t)_{t \in V^T})$ where

- T is a (directed) tree
- $B_t \subseteq V(G)$ for all $t \in V^T$

such that

1. for every edge $\{u, v\} \in E(G)$ there is $t \in V(T)$ with $u, v \in B_t$
2. for all $v \in V(G)$ the set $\{t : v \in B_t\}$ is non-empty and connected.

The **width** of $\mathcal{T}$ is $\max\{|B_t| - 1 : t \in V(T)\}$

The **tree-width** $\mathrm{tw}(G)$ of G is the minimal width of any of its tree-dec.

Definition: A class $\mathcal{C}$ has bounded tree-width if there is a constant $k \in \mathbb{N}$ such that $\mathrm{tw}(G) \leq k$ for all $G \in \mathcal{C}$.

Tree-Width

Definition:

A **tree-decomposition** of a graph G is a pair $\mathcal{T} := (T, (B_t)_{t \in V^T})$ where

- T is a (directed) tree
- $B_t \subseteq V(G)$ for all $t \in V^T$

such that

1. for every edge $\{u, v\} \in E(G)$ there is $t \in V(T)$ with $u, v \in B_t$
2. for all $v \in V(G)$ the set $\{t : v \in B_t\}$ is non-empty and connected.

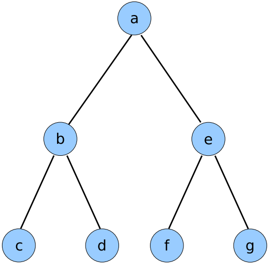
The **width** of $\mathcal{T}$ is $\max\{|B_t| - 1 : t \in V(T)\}$

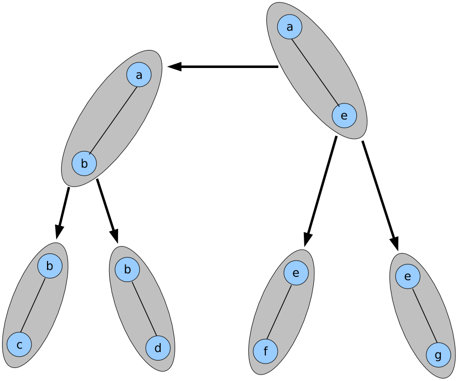
The **tree-width** $\mathrm{tw}(G)$ of G is the minimal width of any of its tree-dec.

Definition: A class $\mathcal{C}$ has bounded tree-width if there is a constant $k \in \mathbb{N}$ such that $\mathrm{tw}(G) \leq k$ for all $G \in \mathcal{C}$.

Examples

Example 1: **Trees/Forests have tree-width 1**

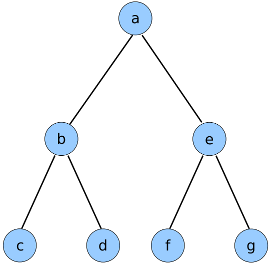


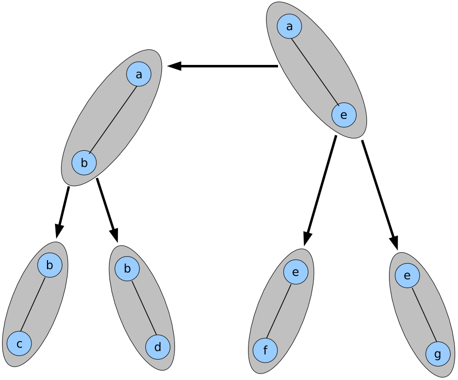


Proposition: Acyclic graphs are precisely the graphs of tree-width 1.

Examples

Example 1: **Trees/Forests have tree-width 1**

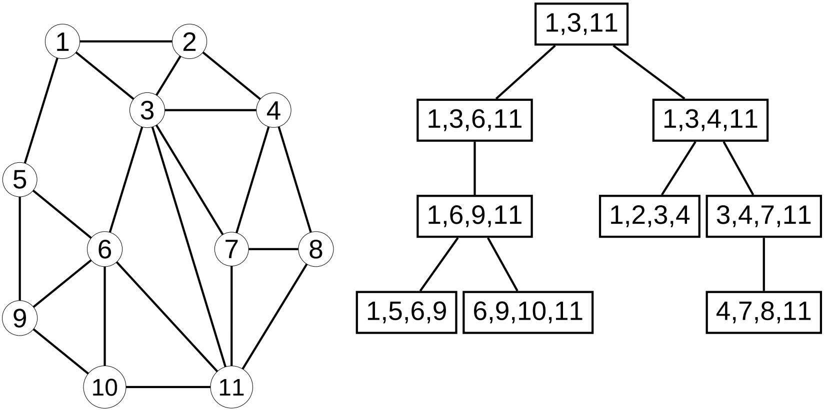




Proposition: Acyclic graphs are precisely the graphs of tree-width 1.

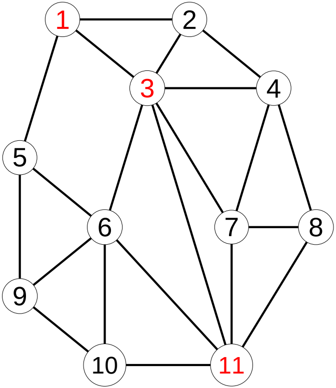
Examples

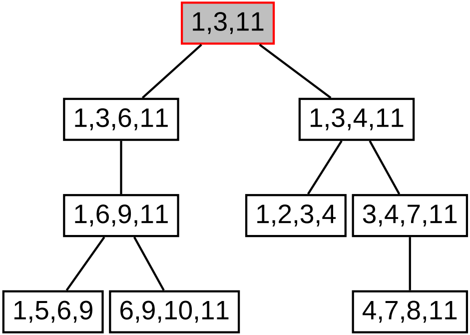
Example 2:



Examples

Example 2:





Courcelle's Theorem

Theorem: (Courcelle 1990)

For any class $\mathcal{C}$ of bounded tree-width

> MC(MSO$_2$, $\mathcal{C}$)
> *Input:* Graph $G \in \mathcal{C}$, $\varphi \in \mathrm{MSO}_2$
> *Parameter:* $|\varphi|$
> *Problem:* Decide $G \models \varphi$

is fixed-parameter tractable (linear time for each fixed φ).

MSO_2: tree-width of a graph equals tree-width of its incidence encoding.

Example: 3-COLOURABILITY

$$\underbrace{\exists C_1 \exists C_2 \exists C_3}_{\substack{\text{there are sets}\\ C_1, C_2, C_3}} \Big(\underbrace{\forall x \bigvee_{i=1}^{3} C_i(x)}_{\text{ev. node has a col.}} \quad \wedge \quad \underbrace{\forall x \forall y (E(x,y) \rightarrow \bigwedge_{i=1}^{3} \neg(C_i(x) \wedge C_i(y)))}_{\text{endpoints of edges have different colours}} \Big)$$

First Ingredient: Computing Tree-Decompositions

Theorem: (Arnborg, Corneil, Proskurowski, 1987)

The problem

TREE-WIDTH
Input: Graph G and $k \in \mathbb{N}$
Problem: tree-width$(G) \leq k$?

is NP-complete.

Theorem: (Bodlaender 1996)

There is an algorithm that, given a graph G constructs a tree-decomposition of minimal width in time

$$\mathcal{O}(2^{\mathrm{tw}(G)^3}|G|).$$

Hence, if $\mathcal{C}$ is a class of graphs of tree-width at most k then for all $G \in \mathcal{C}$ we can compute an optimal tree-decomposition in linear time.

First Ingredient: Computing Tree-Decompositions

Theorem: (Arnborg, Corneil, Proskurowski, 1987)

The problem

TREE-WIDTH	
Input:	Graph G and $k \in \mathbb{N}$
Problem:	tree-width$(G) \leq k$?

is NP-complete.

Theorem: (Bodlaender 1996)

There is an algorithm that, given a graph G constructs a tree-decomposition of minimal width in time

$$\mathcal{O}(2^{\text{tw}(G)^3}|G|).$$

Hence, if $\mathcal{C}$ is a class of graphs of tree-width at most k then for all $G \in \mathcal{C}$ we can compute an optimal tree-decomposition in linear time.

Second Ingredient: Feferman-Vaught Style Theorems

Theorem. Let G, H be graphs

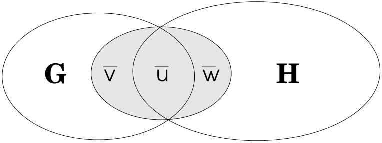
$\overline{v} \in V(G) \qquad \overline{w} \in V(H)$

$\overline{u} \in V(G)$ such that $\overline{u} = V(G) \cap V(H)$

For all $q \geq 0$,

$$\mathrm{tp}_q(G \cup H, \overline{uvw}) \text{ is determined by } \mathrm{tp}_q(G, \overline{uv}) \text{ and } \mathrm{tp}_q(\overline{uw})$$

Furthermore, there is an algorithm that computes $\mathrm{tp}_q(G \cup H, \overline{uvw})$ from $\mathrm{tp}_q(G, \overline{uv})$ and $\mathrm{tp}_q(\overline{uw})$.

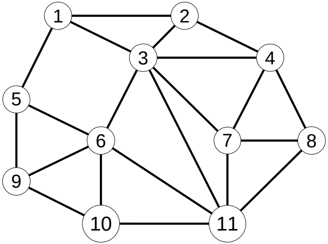


Courcelle's Theorem: Algorithm

Given: Graph G of tree-width $\leq k$ fixed MSO-formula φ of q.r. q

1. Compute a tree-decomposition $\mathcal{T} := (T, (B_t)_{t \in V^T})$ of G
2. Compute the MSO_q-type $\mathrm{tp}^{\mathrm{MSO}}(B_t)$ for each leaf t
3. Bottom up, compute $\mathrm{tp}_q^{\mathrm{MSO}}(G[\bigcup_{t \prec s} B_s], B_t)$ for each $t \in V(T)$

   MSO_q-type of B_t in $G[\bigcup_{t \prec s} B_s]$ (graph induced by $\bigcup_{t \prec s} B_s$)
4. Check whether $\varphi \in \mathrm{tp}_q^{\mathrm{MSO}}(G, B_r)$ at the root r of G

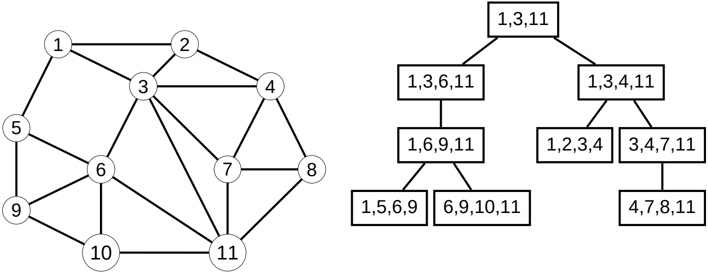


Courcelle's Theorem: Algorithm

Given: Graph G of tree-width $\leq k$ fixed MSO-formula φ of q.r. q

1. Compute a tree-decomposition $\mathcal{T} := (T, (B_t)_{t \in V^T})$ of G
2. Compute the MSO_q-type $\mathrm{tp}^{\mathrm{MSO}}(B_t)$ for each leaf t
3. Bottom up, compute $\mathrm{tp}_q^{\mathrm{MSO}}(G[\bigcup_{t \prec s} B_s], B_t)$ for each $t \in V(T)$

   MSO_q-type of B_t in $G[\bigcup_{t \prec s} B_s]$ (graph induced by $\bigcup_{t \prec s} B_s$)
4. Check whether $\varphi \in \mathrm{tp}_q^{\mathrm{MSO}}(G, B_r)$ at the root r of G

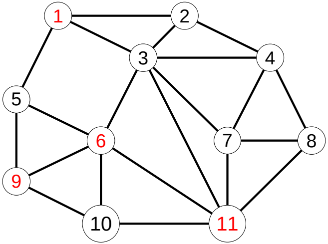


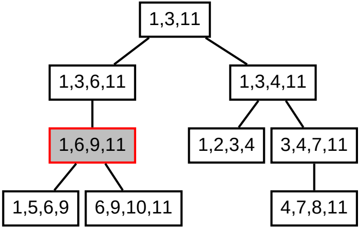
Courcelle's Theorem: Algorithm

Given: Graph G of tree-width $\leq k$ fixed MSO-formula φ of q.r. q

1. Compute a tree-decomposition $\mathcal{T} := (T, (B_t)_{t \in V^T})$ of G
2. Compute the MSO_q-type $\mathrm{tp}^{\mathrm{MSO}}(B_t)$ for each leaf t
3. Bottom up, compute $\mathrm{tp}_q^{\mathrm{MSO}}(G[\bigcup_{t \prec s} B_s], B_t)$ for each $t \in V(T)$

   MSO_q-type of B_t in $G[\bigcup_{t \prec s} B_s]$ (graph induced by $\bigcup_{t \prec s} B_s$)
4. Check whether $\varphi \in \mathrm{tp}_q^{\mathrm{MSO}}(G, B_r)$ at the root r of G





Courcelle's Theorem

Theorem: (Courcelle 1990)

For any class $\mathcal{C}$ of bounded tree-width

$\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$	
Input:	Graph $G \in \mathcal{C}$, $\varphi \in \mathrm{MSO}$
Parameter:	$\lvert\varphi\rvert$
Problem:	Decide $G \models \varphi$

is fixed-parameter tractable (linear time for each fixed φ).

What about the parameter dependence?

Theorem: (Frick, Grohe, 01)

1. Unless P=NP, there is no fpt-algorithm for MSO model checking on trees with elementary parameter dependence.
2. Unless FPT=W[1], there is no fpt-algorithm for FO model checking on trees with elementary parameter dependence.

Courcelle's Theorem

Theorem: (Courcelle 1990)

For any class $\mathcal{C}$ of bounded tree-width

$\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$

Input:	Graph $G \in \mathcal{C}$, $\varphi \in \mathrm{MSO}$
Parameter:	$\lvert\varphi\rvert$
Problem:	Decide $G \models \varphi$

is fixed-parameter tractable (linear time for each fixed φ).

What about the parameter dependence?

Theorem: (Frick, Grohe, 01)

1. Unless P=NP, there is no fpt-algorithm for MSO model checking on trees with elementary parameter dependence.
2. Unless FPT=W[1], there is no fpt-algorithm for FO model checking on trees with elementary parameter dependence.

Courcelle's Theorem

Theorem: (Courcelle 1990)

For any class $\mathcal{C}$ of bounded tree-width

$\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$	
Input:	Graph $G \in \mathcal{C}$, $\varphi \in \mathrm{MSO}$
Parameter:	$\lvert\varphi\rvert$
Problem:	Decide $G \models \varphi$

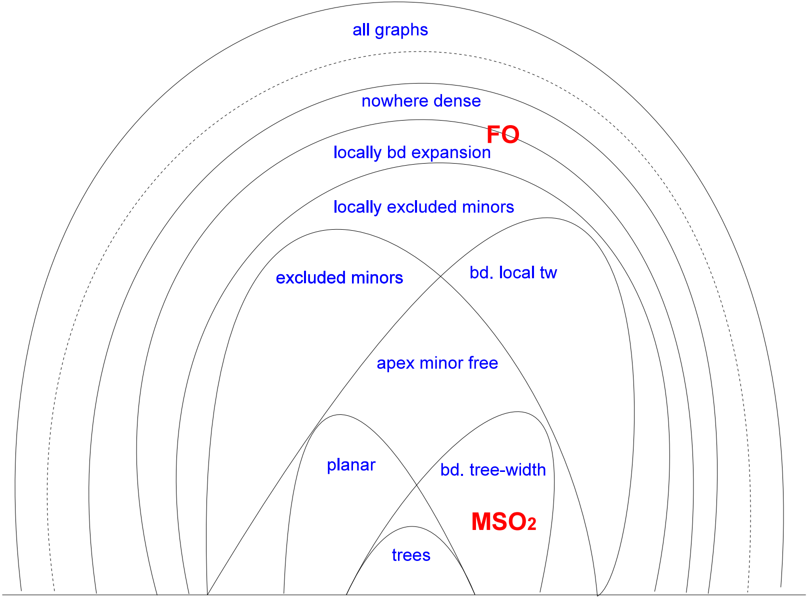
is fixed-parameter tractable (linear time for each fixed φ).

What about the parameter dependence?

Theorem: (Frick, Grohe, 01)

1. Unless P=NP, there is no fpt-algorithm for MSO model checking on trees with elementary parameter dependence.
2. Unless FPT=W[1], there is no fpt-algorithm for FO model checking on trees with elementary parameter dependence.

An Overview of Graph Parameters



The Locality Method for First-Order Logic

Locality of First-Order Logic

Notation: Let G be a graph e.g. the Gaifman graph of a structure

$\text{dist}^G(u,v)$: length of the shortest path between u and v

$$N_r^G(v) := \{u \in V(G) : \text{dist}^G(u,v) \leq r\}$$

$N_r^G(v)$: r-neighbourhood of v in G.

Definition:

A formula $\varphi(x) \in \text{FO}$ is r-local if for all graphs G and all $v \in V(G)$

$$G \models \varphi(v) \iff G[N_r(v)] \models \varphi(v).$$

Hence, truth at v only depends on the vertices around v.

Gaifman's Theorem

Theorem: (Gaifman, 1982)

Every first-order sentence $\varphi \in \mathrm{FO}$ is equivalent to a Boolean combination of basic local sentences.

Basic local sentence:

$$\varphi := \exists x_1 \dots \exists x_m \bigwedge_{i \neq j} \mathrm{dist}(x_i, x_j) > 2r \wedge \bigwedge_{i=1}^{k} \psi(x_i).$$

where ψ is r-local.

Remark: Gaifman's proof is constructive.

Theorem: (Dawar, Grohe, K., Schweikardt, 07)

For each $k \geq 1$ there is $\varphi_k \in \mathrm{FO}[\{E\}]$ of length $\mathcal{O}(k^4)$ such that every equivalent sentence in Gaifman-NF has length at least $tower(k)$.

(similar lower bounds for Feferman-Vaught and preservation thms)

Gaifman's Theorem

Theorem: (Gaifman, 1982)

Every first-order sentence $\varphi \in \mathrm{FO}$ is equivalent to a Boolean combination of basic local sentences.

Basic local sentence:

$$\varphi := \exists x_1 \dots \exists x_m \bigwedge_{i \neq j} \mathrm{dist}(x_i, x_j) > 2r \wedge \bigwedge_{i=1}^{k} \psi(x_i).$$

where ψ is r-local.

Remark: Gaifman's proof is constructive.

Theorem: (Dawar, Grohe, K., Schweikardt, 07)

For each $k \geq 1$ there is $\varphi_k \in \mathrm{FO}[\{E\}]$ of length $\mathcal{O}(k^4)$ such that every equivalent sentence in Gaifman-NF has length at least $tower(k)$.

(similar lower bounds for Feferman-Vaught and preservation thms)

First-Order Logic on Bounded Degree Graphs

Theorem: (Seese, 1996)

Let $\mathcal{C}$ be a class of graphs of maximum degree at most $d \geq 1$.

MC(FO, $\mathcal{C}$)	
Input:	Graph $G \in \mathcal{C}$, $\varphi \in$ FO
Parameter:	$\lvert\varphi\rvert$
Problem:	Decide $G \models \varphi$

is fixed-parameter tractable (linear time fpt algorithm).

Proof. By Gaifman's theorem it suffices to consider formulae of the form

$$\exists x_1 \ldots \exists x_m \bigwedge_{1 \leq i < j \leq m} \mathrm{dist}(x_i, x_j) > 2r \wedge \bigwedge_{i=1}^{k} \psi(x_i)$$

for some r-local formula $\psi(x)$.

First-Order Logic on Bounded Degree Graphs

Theorem: (Seese, 1996)

Let $\mathcal{C}$ be a class of graphs of maximum degree at most $d \geq 1$.

$\mathrm{MC}(\mathrm{FO}, \mathcal{C})$	
Input:	Graph $G \in \mathcal{C}$, $\varphi \in \mathrm{FO}$
Parameter:	$\lvert\varphi\rvert$
Problem:	Decide $G \models \varphi$

is fixed-parameter tractable (linear time fpt algorithm).

Proof. By Gaifman's theorem it suffices to consider formulae of the form

$$\exists x_1 \ldots \exists x_m \bigwedge_{1 \leq i < j \leq m} \mathrm{dist}(x_i, x_j) > 2r \wedge \bigwedge_{i=1}^{k} \psi(x_i)$$

for some r-local formula $\psi(x)$.

Proof of Theorem

Suppose

$$\varphi := \exists x_1 \dots \exists x_m \bigwedge_{1 \le i < j \le m} \mathrm{dist}(x_i, x_j) > 2r \wedge \bigwedge_{i=1}^{m} \psi(x_i)$$

for some r-local formula $\psi(x)$.

Let G be a graph of maximum degree d.

Find m vertices of distance $> 2r$ whose r-neighbourhoods satisfy ψ.

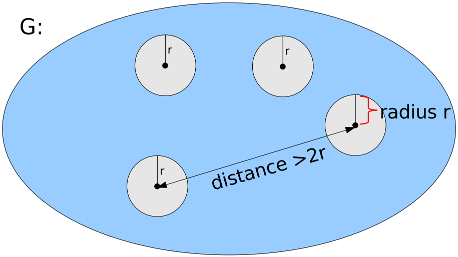
Proof of Theorem

Suppose

$$\varphi := \exists x_1 \dots \exists x_m \bigwedge_{1 \le i < j \le m} \mathrm{dist}(x_i, x_j) > 2r \wedge \bigwedge_{i=1}^{m} \psi(x_i)$$

for some r-local formula $\psi(x)$.

Let G be a graph of maximum degree d.



Find m vertices of distance $> 2r$ whose r-neighbourhoods satisfy ψ.

First Step

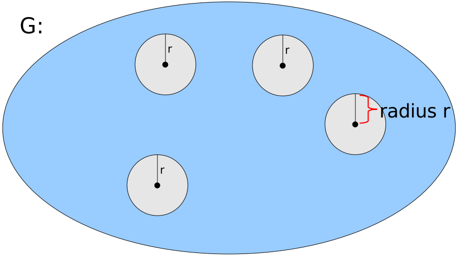
Algorithm: First Step

for all $v \in V(G)$ $\mathcal{O}(n)$

- compute $N_r(v)$ $\mathcal{O}(d^r) = \mathcal{O}(1)$
- test whether $N_r(v) \models \psi(v)$ $\mathcal{O}(1)$
  (constant size neighbourhood)

  if it does, colour the vertex red

Running time: $\mathcal{O}(n)$



First Step

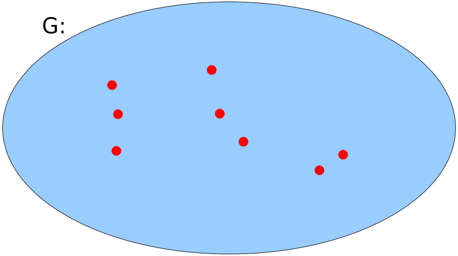
Algorithm: First Step

for all $v \in V(G)$ $\mathcal{O}(n)$

- compute $N_r(v)$ $\mathcal{O}(d^r) = \mathcal{O}(1)$
- test whether $N_r(v) \models \psi(v)$ $\mathcal{O}(1)$
  (constant size neighbourhood)

  if it does, colour the vertex red

Running time: $\mathcal{O}(n)$



First Step

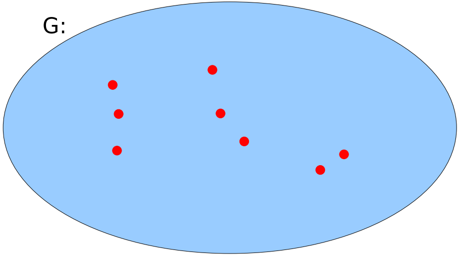
Algorithm: First Step

for all $v \in V(G)$ — $\mathcal{O}(n)$

- compute $N_r(v)$ — $\mathcal{O}(d^r) = \mathcal{O}(1)$
- test whether $N_r(v) \models \psi(v)$ — $\mathcal{O}(1)$
  (constant size neighbourhood)

  if it does, colour the vertex red

Running time: $\mathcal{O}(n)$



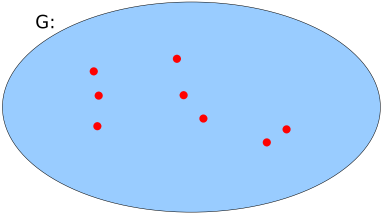
Second Step: Greedy Approach

Let Q be the set of red vertices.

Algorithm: Second Step

$L := \emptyset$
while $Q \neq \emptyset$ **do**
  choose $v \in Q$
  $L := L \cup \{v\}$
  $Q := Q \setminus N_{2r}(v)$
od
if $|L| \geq m$ **then** accept
else
  all red vertices are within a $2r$-neighbourhood of an element of L
  if $G[N_{2r}(L)] \models \exists x_1 \dots x_m (\bigwedge_{i \neq j} \text{dist}(x_i, x_j) > 2r \wedge \bigwedge_i \text{“}x_i \text{ is } red\text{”})$
  accept **else** reject

Assume $m = 4$



Running time: $\mathcal{O}(n)$

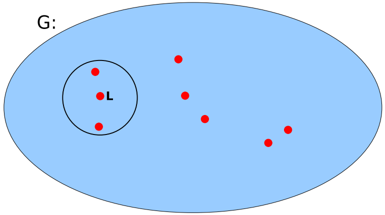
Second Step: Greedy Approach

Let Q be the set of red vertices.

Algorithm: Second Step

$L := \emptyset$
while $Q \neq \emptyset$ **do**
  choose $v \in Q$
  $L := L \cup \{v\}$
  $Q := Q \setminus N_{2r}(v)$
od
if $|L| \geq m$ **then** accept
else
  all red vertices are within a $2r$-neighbourhood of an element of L
  if $G[N_{2r}(L)] \models \exists x_1 \dots x_m (\bigwedge_{i \neq j} \text{dist}(x_i, x_j) > 2r \wedge \bigwedge_i \text{"}x_i \text{ is } red\text{"})$
  accept **else** reject

Assume $m = 4$



Running time: $\mathcal{O}(n)$

Second Step: Greedy Approach

Let Q be the set of red vertices.

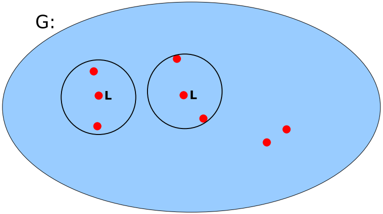
Algorithm: Second Step

$L := \emptyset$
while $Q \neq \emptyset$ **do**
choose $v \in Q$
$L := L \cup \{v\}$
$Q := Q \setminus N_{2r}(v)$
od
if $|L| \geq m$ **then** accept
else
all red vertices are within a $2r$-neighbourhood of an element of L

if $G[N_{2r}(L)] \models \exists x_1 \dots x_m (\bigwedge_{i \neq j} \mathrm{dist}(x_i, x_j) > 2r \wedge \bigwedge_i \text{“}x_i \text{ is } red\text{”})$

accept **else** reject

Assume $m = 4$



Running time: $\mathcal{O}(n)$

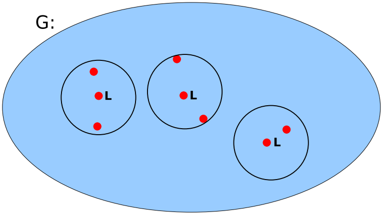
Second Step: Greedy Approach

Let Q be the set of red vertices.

Algorithm: Second Step

$L := \emptyset$
while $Q \neq \emptyset$ **do**
choose $v \in Q$
$L := L \cup \{v\}$
$Q := Q \setminus N_{2r}(v)$
od
if $|L| \geq m$ **then** accept
else
all red vertices are within a $2r$-neighbourhood of an element of L
if $G[N_{2r}(L)] \models \exists x_1 \dots x_m (\bigwedge_{i \neq j} \text{dist}(x_i, x_j) > 2r \wedge \bigwedge_i \text{“}x_i \text{ is } red\text{”})$
accept **else** reject

Assume $m = 4$



Running time: $\mathcal{O}(n)$

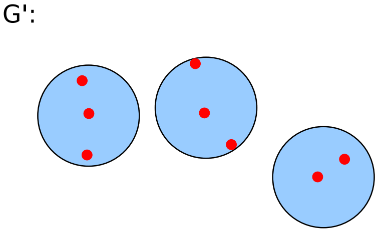
Second Step: Greedy Approach

Let Q be the set of red vertices.

Algorithm: Second Step

$L := \emptyset$
while $Q \neq \emptyset$ **do**
  choose $v \in Q$
  $L := L \cup \{v\}$
  $Q := Q \setminus N_{2r}(v)$
od
if $|L| \geq m$ **then** accept
else
  all red vertices are within a $2r$-neighbourhood of an element of L
  if $G[N_{2r}(L)] \models \exists x_1 \dots x_m (\bigwedge_{i \neq j} \text{dist}(x_i, x_j) > 2r \wedge \bigwedge_i \text{“}x_i \text{ is } red\text{”})$
  accept **else** reject

Assume $m = 4$



Running time: $\mathcal{O}(n)$

Second Step: Greedy Approach

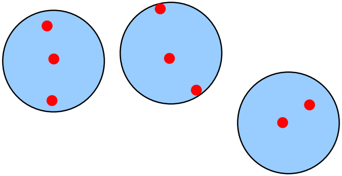
Let Q be the set of red vertices.

Algorithm: Second Step

$L := \emptyset$
while $Q \neq \emptyset$ **do**
  choose $v \in Q$
  $L := L \cup \{v\}$
  $Q := Q \setminus N_{2r}(v)$
od
if $|L| \geq m$ **then** accept
else
  all red vertices are within a $2r$-neighbourhood of an element of L
  if $G[N_{2r}(L)] \models \exists x_1 \dots x_m (\bigwedge_{i \neq j} \text{dist}(x_i, x_j) > 2r \wedge \bigwedge_i \text{"}x_i \text{ is } red\text{"})$
  accept **else** reject

Assume $m = 4$

G':



Running time: $\mathcal{O}(n)$

First-Order Logic on Bounded Degree Graphs

Theorem: (Seese, 1996)

Let $\mathcal{C}$ be a class of graphs of maximum degree at most $d \geq 1$.

MC(FO, $\mathcal{C}$)

Input:	Graph $G \in \mathcal{C}$, $\varphi \in$ FO
Parameter:	$\lvert\varphi\rvert$
Problem:	Decide $G \models \varphi$

is fixed-parameter tractable (linear time fpt algorithm).

But wait:

The proof shows much more ...

... for, where did we use bounded degree?

First-Order Logic on Bounded Degree Graphs

Theorem: (Seese, 1996)

Let $\mathcal{C}$ be a class of graphs of maximum degree at most $d \geq 1$.

MC(FO, $\mathcal{C}$)	
Input:	Graph $G \in \mathcal{C}$, $\varphi \in$ FO
Parameter:	$\|\varphi\|$
Problem:	Decide $G \models \varphi$

is fixed-parameter tractable (linear time fpt algorithm).

But wait:

The proof shows much more ...

... for, where did we use bounded degree?

First-Order Logic on Bounded Degree Graphs

Theorem: (Seese, 1996)

Let $\mathcal{C}$ be a class of graphs of maximum degree at most $d \geq 1$.

MC(FO, $\mathcal{C}$)	
Input:	Graph $G \in \mathcal{C}$, $\varphi \in$ FO
Parameter:	$\lvert\varphi\rvert$
Problem:	Decide $G \models \varphi$

is fixed-parameter tractable (linear time fpt algorithm).

But wait:

The proof shows much more ...

... for, where did we use bounded degree?

First Step

Algorithm: First Step

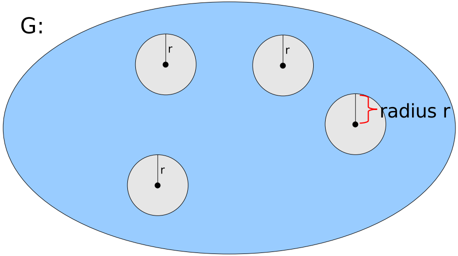
for all $v \in V(G)$ — $\mathcal{O}(n)$

- compute $N_r(v)$ — $\mathcal{O}(d^r) = \mathcal{O}(1)$
- test whether $N_r(v) \models \psi(v)$ — $\mathcal{O}(1)$

  (constant size neighbourhood)

  if it does, colour the vertex red

Running time: $\mathcal{O}(n)$



Second Step: Greedy Approach

Let Q be the set of red vertices.

Algorithm: Second Step

$L := \emptyset$

while $Q \neq \emptyset$ **do**

choose $v \in Q$

$L := L \cup \{v\}$

$Q := Q \setminus N_{2r}(v)$

od

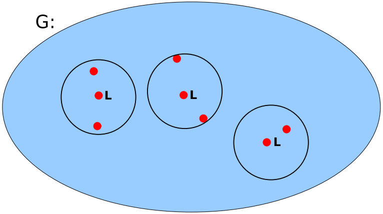
if $|L| \geq m$ **then** accept

else

all red vertices are within a $2r$-neighbourhood of an element of L

if $G[N_{2r}(L)] \models \exists x_1 \dots x_k (\bigwedge_{i \neq j} \mathrm{dist}(x_i, x_j) > 2r \wedge \bigwedge_i \text{“}x_i \text{ is } red\text{”})$

accept **else** reject



Running time: $\mathcal{O}(n)$

Local Model Checking

Essentially:

- We need to be able to test r-local formulae $\psi(x)$ in r'-neighbourhoods

Here: r, r' depend on the original formula φ and hence are constant (part of the parameter).

Local Model Checking

Theorem: Let $\mathcal{C}$ be a class of graphs such that the following is fpt:

LOCAL-FO-MC($\mathcal{C}$)	
Input:	$\varphi \in \mathrm{FO}$, Graph $G \in \mathcal{C}$, $v_1, \dots, v_k \in V(G)$, and $r \in \mathbb{N}$
Parameter:	$r + k + \lvert\varphi\rvert$
Problem:	Decide $G\left[N_r^G(v_1, \dots, v_k)\right] \models \varphi$

Then first-order model checking is fixed-parameter tractable on $\mathcal{C}$.

Consequences: For efficient first-order model checking, it suffices if every neighbourhood in a graph is "well-behaved".

Not the whole graph needs to have small tree-width, but only its neighbourhoods.

Local Model Checking

Theorem: Let $\mathcal{C}$ be a class of graphs such that the following is fpt:

LOCAL-FO-MC($\mathcal{C}$)	
Input:	$\varphi \in \mathrm{FO}$, Graph $G \in \mathcal{C}, v_1, \dots, v_k \in V(G)$, and $r \in \mathbb{N}$
Parameter:	$r + k + \lvert\varphi\rvert$
Problem:	Decide $G\left[N_r^G(v_1, \dots, v_k)\right] \models \varphi$

Then first-order model checking is fixed-parameter tractable on $\mathcal{C}$.

Consequences: For efficient first-order model checking, it suffices if every neighbourhood in a graph is "well-behaved".

Not the whole graph needs to have small tree-width, but only its neighbourhoods.

Localisation of Graph Invariants

Graph Invariants

Definition:

A graph invariant is a function $f : \text{GRAPHS} \to \mathbb{N}$.

A class $\mathcal{C}$ has bounded f, if there is a constant $k : \mathbb{N}$ such that $f(G) \leq k$ for all $G \in \mathcal{C}$.

Examples:

- $f : G \mapsto \Delta(G)$ (max. degree in G)
  classes of bounded degree
- $f : G \mapsto \mathrm{tw}(G)$ (tree-width of G)
  classes of bounded tree-width
- $f : G \mapsto mec(G)$ ($mec(G)$: minimal order of a clique $K_m \not\preceq G$)
  classes excluding a minor

Localisation of Graph Invariants

Definition:

Let f : GRAPHS $\rightarrow \mathbb{N}$ be a graph invariant.

We define its localisation loc_f : GRAPHS $\times \mathbb{N} \rightarrow \mathbb{N}$ as

$$loc_f(G, r) := \max \Big\{ f\Big(G\big[N_r(v)\big]\Big) : v \in V(G) \Big\}.$$

A class $\mathcal{C}$ of graphs has bounded local f, if there is a computable function $h : \mathbb{N} \rightarrow \mathbb{N}$ such that $loc_f(G, r) \leq h(r)$ for all $G \in \mathcal{C}$ and $r \in \mathbb{N}$.

Example: $f : G \mapsto \mathrm{tw}(G)$ tree-width of graphs

$$\rightsquigarrow loc_f(G, r) := \max \Big\{ \mathrm{tw}\Big(G\big[N_r(v)\big]\Big) : v \in V(G) \Big\}$$

Bounded local tree-width

Localisation of Graph Invariants

Definition:

Let $f : \text{GRAPHS} \to \mathbb{N}$ be a graph invariant.

We define its localisation $loc_f : \text{GRAPHS} \times \mathbb{N} \to \mathbb{N}$ as

$$loc_f(G, r) := \max\Big\{f\Big(G\big[N_r(v)\big]\Big) : v \in V(G)\Big\}.$$

A class $\mathcal{C}$ of graphs has bounded local f, if there is a computable function $h : \mathbb{N} \to \mathbb{N}$ such that $loc_f(G, r) \leq h(r)$ for all $G \in \mathcal{C}$ and $r \in \mathbb{N}$.

Example: $f : G \mapsto \mathrm{tw}(G)$ tree-width of graphs

$$\rightsquigarrow loc_f(G, r) := \max\Big\{\mathrm{tw}\Big(G\big[N_r(v)\big]\Big) : v \in V(G)\Big\}$$

Bounded local tree-width

Bounded Local Tree-Width

Bounded local tree-width:

$f : G \mapsto \mathrm{tw}(G)$ tree-width of graphs

$$\rightsquigarrow loc_f(G, r) := \max\left\{\mathrm{tw}\Big(G\big[N_r(v)\big]\Big) : v \in V(G)\right\}$$

Example: Every class of graphs of bounded degree has bounded local tree-width.

Example: The class of planar graphs has bounded local tree-width.

Theorem: (Baker)

Every planar graph of diameter r has tree-width at most $3r$.

Localisation of Graph Invariants

Let f : GRAPHS $\rightarrow \mathbb{N}$ be a induced subgraph monotone graph invariant.

Theorem: Let $\mathcal{C}$ be a class of graphs such that the following is fpt:

MC(FO, f)	
Input:	$\varphi \in$ FO, Graph $G \in \mathcal{C}$
Parameter:	$\|\varphi\| + f(G)$
Problem:	Decide $G \models \varphi$

Then first-order model checking is fixed-parameter tractable on $\mathcal{C}$.

Follows immediately from the following theorem proved before.

Theorem: Let $\mathcal{C}$ be a class of graphs such that the following is fpt:

LOCAL-FO-MC	
Input:	$\varphi \in$ FO, Graph $G \in \mathcal{C}$,$v_1, \dots, v_k \in V(G)$, and $r \in \mathbb{N}$
Parameter:	$r + k + \|\varphi\|$
Problem:	Decide $G[N_r^G(v_1, \dots, v_k)] \models \varphi$

Then first-order model checking is fixed-parameter tractable on $\mathcal{C}$.

Localisation of Graph Invariants

Let $f : \text{GRAPHS} \to \mathbb{N}$ be a induced subgraph monotone graph invariant.

Theorem: Let $\mathcal{C}$ be a class of graphs such that the following is fpt:

MC(FO, f)	
Input:	$\varphi \in \text{FO}$, Graph $G \in \mathcal{C}$
Parameter:	$\lvert\varphi\rvert + f(G)$
Problem:	Decide $G \models \varphi$

Then first-order model checking is fixed-parameter tractable on $\mathcal{C}$.

Follows immediately from the following theorem proved before.

Theorem: Let $\mathcal{C}$ be a class of graphs such that the following is fpt:

LOCAL-FO-MC	
Input:	$\varphi \in \text{FO}$, Graph $G \in \mathcal{C}$,$v_1, \dots, v_k \in V(G)$, and $r \in \mathbb{N}$
Parameter:	$r + k + \lvert\varphi\rvert$
Problem:	Decide $G[N_r^G(v_1, \dots, v_k)] \models \varphi$

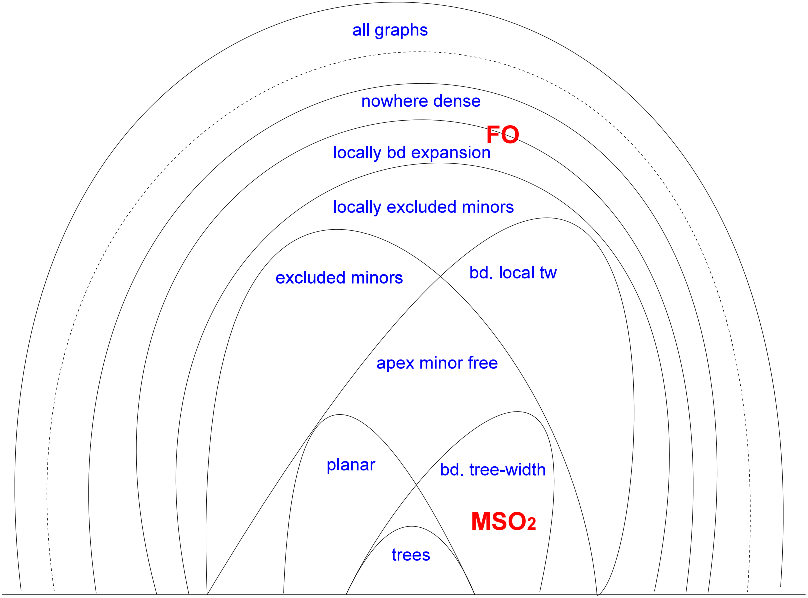
Then first-order model checking is fixed-parameter tractable on $\mathcal{C}$.

Localisation of Graph Invariants

Theorem: First-order model checking is fixed-parameter tractable on

- planar graphs (Frick, Grohe 01)
- graphs of locally bounded tree-width (Frick, Grohe 01)

An Overview of Graph Parameters



Localisation of Graph Invariants

Theorem: First-order model checking is fixed-parameter tractable on

- planar graphs (Frick, Grohe 01)
- graphs of locally bounded tree-width (Frick, Grohe 01)

Theorem: (Flum, Grohe 01)
First-order model-checking is fixed-parameter tractable on graph classes excluding a minor.

Theorem: (Dawar, Grohe, K. 07)
First-order model-checking is fixed-parameter tractable on graph classes locally excluding a minor.

Theorem: (Dvořák, Kral, Thomas 11)
First-order model-checking is fixed-parameter tractable on graph classes of locally bounded expansion.

Localisation of Graph Invariants

Theorem: First-order model checking is fixed-parameter tractable on

- planar graphs (Frick, Grohe 01)
- graphs of locally bounded tree-width (Frick, Grohe 01)

Theorem: (Flum, Grohe 01)
First-order model-checking is fixed-parameter tractable on graph classes excluding a minor.

Theorem: (Dawar, Grohe, K. 07)
First-order model-checking is fixed-parameter tractable on graph classes locally excluding a minor.

Theorem: (Dvořák, Kral, Thomas 11)
First-order model-checking is fixed-parameter tractable on graph classes of locally bounded expansion.

Localisation of Graph Invariants

Theorem: First-order model checking is fixed-parameter tractable on

- planar graphs (Frick, Grohe 01)
- graphs of locally bounded tree-width (Frick, Grohe 01)

Theorem: (Flum, Grohe 01)
First-order model-checking is fixed-parameter tractable on graph classes excluding a minor.

Theorem: (Dawar, Grohe, K. 07)
First-order model-checking is fixed-parameter tractable on graph classes locally excluding a minor.

Theorem: (Dvořák, Kral, Thomas 11)
First-order model-checking is fixed-parameter tractable on graph classes of locally bounded expansion.

Localisation of Graph Invariants

Theorem: First-order model checking is fixed-parameter tractable on

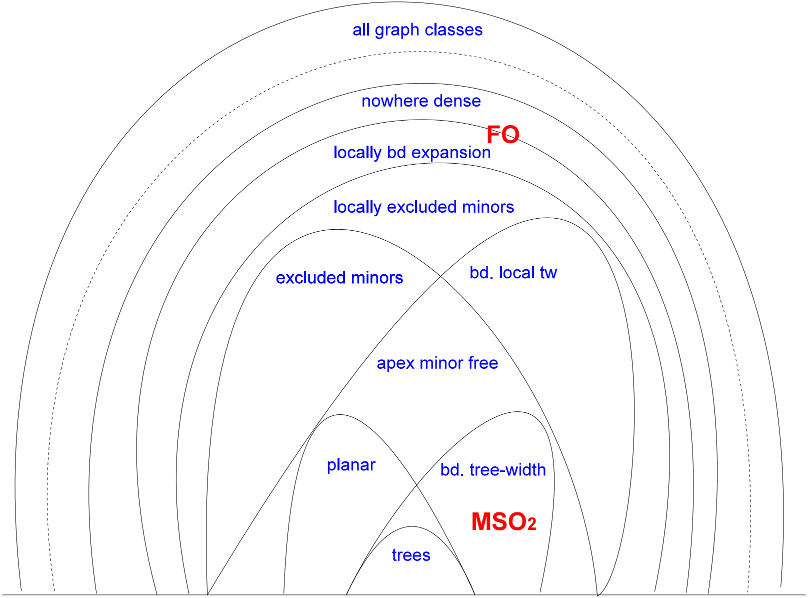
- planar graphs (Frick, Grohe 01)
- graphs of locally bounded tree-width (Frick, Grohe 01)

Theorem: (Flum, Grohe 01)
First-order model-checking is fixed-parameter tractable on graph classes excluding a minor.

Theorem: (Dawar, Grohe, K. 07)
First-order model-checking is fixed-parameter tractable on graph classes locally excluding a minor.

Theorem: (Dvořák, Kral, Thomas 11)
First-order model-checking is fixed-parameter tractable on graph classes of locally bounded expansion.

An Overview of Graph Parameters



Part II: Lower Bounds

Lower Bounds for Monadic Second-Order Logic

We would like to show. If a class $\mathcal{C}$ of graphs has unbounded tree-width then $\text{MC}(\text{MSO}_2, \mathcal{C})$ is not fixed-parameter tractable.

Sadly, in this generality this is not true.

Theorem. (Makowsky, Mariño 04)

There are classes $\mathcal{C}$ of graphs of unbounded tree-width on which $\text{MC}(\text{MSO}_2, \mathcal{C})$ is tractable.

But something similar is true.

Unbounded Tree-Width.

We first need to classify the *unboundedness* of tree-width.

Classes of Unbounded Tree-Width

Definition. Let $f : \mathbb{N} \to \mathbb{N}$ be a non-decreasing function.
A class $\mathcal{C}$ of graphs has f-bounded tree-width if $\mathrm{tw}(G) \leq f(|G|)$ for all $G \in \mathcal{C}$.

Examples.

- In Courcelle's theorem, $f(n) := c$ is constant.
- $f(n) := n$ is the maximal function that makes sense here.
- We will look at $f(n) := \log^c n$ for a small constant $c > 0$.

Theorem by Makowsky, Mariño.

There are classes $\mathcal{C}$ of graphs of logarithmic tree-width on which $\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$ is tractable.

What we would like to show. If the tree-width of $\mathcal{C}$ is not bounded by $\log^c n$, for small constant c, then $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is not FPT.

Classes of Unbounded Tree-Width

Definition. Let $f : \mathbb{N} \to \mathbb{N}$ be a non-decreasing function.

A class $\mathcal{C}$ of graphs has f-bounded tree-width if $\mathrm{tw}(G) \leq f(|G|)$ for all $G \in \mathcal{C}$.

Examples.

- In Courcelle's theorem, $f(n) := c$ is constant.
- $f(n) := n$ is the maximal function that makes sense here.
- We will look at $f(n) := \log^c n$ for a small constant $c > 0$.

Theorem by Makowsky, Mariño.

There are classes $\mathcal{C}$ of graphs of logarithmic tree-width on which $\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$ is tractable.

What we would like to show. If the tree-width of $\mathcal{C}$ is not bounded by $\log^c n$, for small constant c, then $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is not FPT.

Classes of Unbounded Tree-Width

Definition. Let $f : \mathbb{N} \to \mathbb{N}$ be a non-decreasing function.
A class $\mathcal{C}$ of graphs has f-bounded tree-width if $\mathrm{tw}(G) \leq f(|G|)$ for all $G \in \mathcal{C}$.

Examples.

- In Courcelle's theorem, $f(n) := c$ is constant.
- $f(n) := n$ is the maximal function that makes sense here.
- We will look at $f(n) := \log^c n$ for a small constant $c > 0$.

Theorem by Makowsky, Mariño.

There are classes $\mathcal{C}$ of graphs of logarithmic tree-width on which $\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$ is tractable.

What we would like to show. If the tree-width of $\mathcal{C}$ is not bounded by $\log^c n$, for small constant c, then $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is not FPT.

Classes of Unbounded Tree-Width

What we would like to show. If the tree-width of $\mathcal{C}$ is not bounded by $\log^c n$, for small constant c, then $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is not FPT.

Problem. Any such result would separate PTIME from PSPACE.

For, $\mathrm{MC}(\mathrm{MSO}_2)$ is in PSPACE. Hence, if PSPACE collapses to PTIME then MSO is fixed-parameter tractable on the class of all graphs.

We will therefore show hardness of $\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$ by reducing a hard problem to it.

For this to work we need to

1. understand what structural information we can draw from the fact that the tree-width of graphs is high $\rightsquigarrow$ obstructions
2. use this information to reduce a hard problem to $\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$.

This requires some further technical conditions.

Classes of Unbounded Tree-Width

Definition. Let $f : \mathbb{N} \to \mathbb{N}$ be a function and $p(n)$ be a polynomial.

The tree-width of a class $\mathcal{C}$ is (f, p)-unbounded if there is an $\epsilon < 1$ such that for all $n \in \mathbb{N}$ there is a graph $G_n \in \mathcal{C}$ with

1. $n \leq \text{tw}(G_n) \leq p(n)$ and $\text{tw}(G_n) \geq f(|G_n|)$
2. given n (in unary), G_n can be constructed in time 2^{n^ϵ}.

The tree-width of $\mathcal{C}$ is f-unbounded if it is (f, p)-unbounded for some $p(n)$.

Theorem. (K., Tazari 10)

Let $\mathcal{C}$ be a class of graph closed under sub-graphs and let $p(n)$ be a polynomial of degree $< \gamma$.

If the tree-width of $\mathcal{C}$ is $(\log^{28+\gamma} n, p)$-unbounded then $\text{MC}(\text{MSO}, \mathcal{C})$ is not fpt unless SAT can be solved in sub-exponential time.

(fpt: with parameter $|\varphi|$)

Classes of Unbounded Tree-Width

Definition. Let $f : \mathbb{N} \to \mathbb{N}$ be a function and $p(n)$ be a polynomial.

The tree-width of a class $\mathcal{C}$ is (f,p)-unbounded if there is an $\epsilon < 1$ such that for all $n \in \mathbb{N}$ there is a graph $G_n \in \mathcal{C}$ with

1. $n \leq \mathrm{tw}(G_n) \leq p(n)$ and $\mathrm{tw}(G_n) \geq f(|G_n|)$
2. given n (in unary), G_n can be constructed in time 2^{n^ϵ}.

The tree-width of $\mathcal{C}$ is f-unbounded if it is (f,p)-unbounded for some $p(n)$.

Theorem. (K., Tazari 10)

Let $\mathcal{C}$ be a class of graph closed under sub-graphs and let $p(n)$ be a polynomial of degree $< \gamma$.

If the tree-width of $\mathcal{C}$ is $(\log^{28+\gamma} n, p)$-unbounded then $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is not fpt unless SAT can be solved in sub-exponential time.

(fpt: with parameter $|\varphi|$)

General Proof Idea

We reduce the propositional satisfiability problem (SAT) to $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$.

Given: $(X_1 \vee X_2 \vee \neg X_3) \wedge (X_4 \vee \neg X_5 \vee X_6)\ldots \quad \hat{=} \quad w \in \{0,1\}^*$
Problem: Decide if w is satisfiable.

Reduction.

1. Construct $G_w \in \mathcal{C}$ of tree-width $|w|^c$ with $\mathrm{tw}(G_w) > \log^d |G_w|$

   Condition 1: G_w exists in $\mathcal{C}$
   Condition 2: G_w can be computed efficiently

2. Somehow encode w in a sub-graph of G_w

   Use obstructions to tree-width.

   use closure under sub-graphs

3. Define an MSO-formula φ (independent of w) which is true in G_w iff w is satisfiable.

   φ decodes w in G_w and decides whether w is satisfiable.

Outline of the Intractability Proof

1. Some simple intractability results
2. Grid-Like Minors
   (or: why the excluded grid theorem is useless (in this context))
3. Tree Labelled Webs
4. Intractability of MSO

Of Grids and Walls

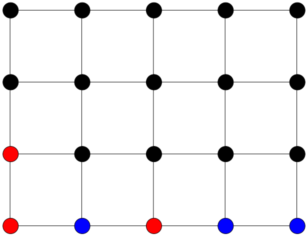
First Example: Coloured Grids

Coloured Grid. A grid whose vertices may be coloured red or blue.

Let $\mathcal{G}$ be the class of all finite coloured grids.

Theorem. Let $\mathcal{G}$ be the class of coloured grids. Then MC$(\mathrm{MSO}, \mathcal{G})$ is not fixed-parameter tractable unless P=NP.

(4×5)-grid



Intractability on Grids

Theorem. Let $\mathcal{G}$ be the class of coloured grids. Then MC$(\text{MSO}, \mathcal{G})$ is not fixed-parameter tractable unless P=NP.

Proof. Let SAT be the NP-complete propositional satisfiability problem.

SAT can be solved in quadratic time by an NTM $\mathcal{M}$.

Given: SAT-instance $w := (X_1 \vee \neg X_3 ...) \hat{=} 010...$

Look at the time-space diagram of an acc. run of $\mathcal{M}$ on w.

0	**1**	**0**	□	□	□

Intractability on Grids

Theorem. Let $\mathcal{G}$ be the class of coloured grids. Then $\mathrm{MC}(\mathrm{MSO}, \mathcal{G})$ is not fixed-parameter tractable unless P=NP.

Proof. Let SAT be the NP-complete propositional satisfiability problem.

SAT can be solved in quadratic time by an NTM $\mathcal{M}$.

Given: SAT-instance $w := (X_1 \vee \neg X_3 ...) \hat{=} 010...$

Look at the time-space diagram of an acc. run of $\mathcal{M}$ on w.

1	1	1	□ q_f	□	□
1	1	0 q_2	□	□	□
1	1 q_1	0	□	□	□
0 q_0	1	0	□	□	□

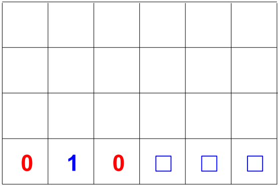
Intractability on Grids

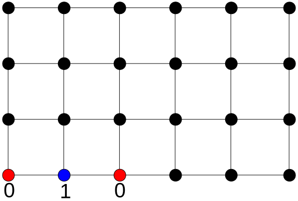
Theorem. Let $\mathcal{G}$ be the class of coloured grids. Then $\mathrm{MC}(\mathrm{MSO}, \mathcal{G})$ is not fixed-parameter tractable unless P=NP.

Proof. We reduce SAT to $\mathrm{MC}(\mathrm{MSO}, \mathcal{G})$ as follows.

1. Given SAT instance w of length n, construct an $n^2 \times n^2$-grid G_w and colour its bottom row by w.
2. Construct a formula $\varphi_{\mathcal{M}} \in \mathrm{MSO}$ which guesses a colouring of the grid and checks that this encodes a successful run of $\mathcal{M}$ on input w.

Then $w \in$ SAT if, and only if, $G_w \models \varphi_{\mathcal{M}}$.





$$\exists X_0 \exists X_1 \exists X_\Box \exists X_{q_0} \dots X_{q_k} \psi \dots \in \mathrm{MSO}_1$$

Intractability on Grids

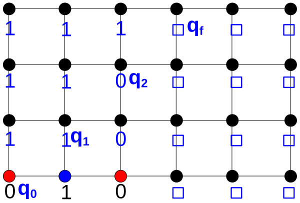
Theorem. Let $\mathcal{G}$ be the class of coloured grids. Then $\mathrm{MC}(\mathrm{MSO}, \mathcal{G})$ is not fixed-parameter tractable unless P=NP.

Proof. We reduce SAT to $\mathrm{MC}(\mathrm{MSO}, \mathcal{G})$ as follows.

1. Given SAT instance w of length n, construct an $n^2 \times n^2$-grid G_w and colour its bottom row by w.
2. Construct a formula $\varphi_{\mathcal{M}} \in \mathrm{MSO}$ which guesses a colouring of the grid and checks that this encodes a successful run of $\mathcal{M}$ on input w.

Then $w \in$ SAT if, and only if, $G_w \models \varphi_{\mathcal{M}}$.

1	**1**	**1**	$\square_{q_f}$	$\square$	$\square$
1	**1**	**0** $_{q_2}$	$\square$	$\square$	$\square$
1	**1** $_{q_1}$	**0**	$\square$	$\square$	$\square$
0 $_{q_0}$	**1**	**0**	$\square$	$\square$	$\square$



$$\exists X_0 \exists X_1 \exists X_\square \exists X_{q_0} \dots X_{q_k} \psi \dots \in \mathrm{MSO}_1$$

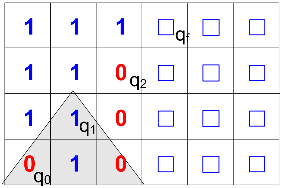
Intractability on Grids

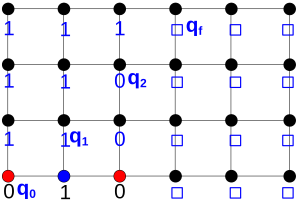
Theorem. Let $\mathcal{G}$ be the class of coloured grids. Then $\mathrm{MC}(\mathrm{MSO}, \mathcal{G})$ is not fixed-parameter tractable unless P=NP.

Proof. We reduce SAT to $\mathrm{MC}(\mathrm{MSO}, \mathcal{G})$ as follows.

1. Given SAT instance w of length n, construct an $n^2 \times n^2$-grid G_w and colour its bottom row by w.
2. Construct a formula $\varphi_{\mathcal{M}} \in \mathrm{MSO}$ which guesses a colouring of the grid and checks that this encodes a successful run of $\mathcal{M}$ on input w.

Then $w \in$ SAT if, and only if, $G_w \models \varphi_{\mathcal{M}}$.





$$\exists X_0 \exists X_1 \exists X_\square \exists X_{q_0} \dots X_{q_k} \psi \dots \in \mathrm{MSO}_1$$

Intractability on Grids

Theorem. Let $\mathcal{G}$ be the class of coloured grids. Then $\text{MC}(\text{MSO}, \mathcal{G})$ is not fixed-parameter tractable unless P=NP.

Proof. We reduce SAT to $\text{MC}(\text{MSO}, \mathcal{G})$ as follows.

1. Given a propositional logic formula w of length n in CNF, construct an $n^2 \times n^2$-grid G_w and colour its bottom row by w.
2. Construct a formula $\varphi_{\mathcal{M}} \in \text{MSO}$ which guesses a colouring of the grid and checks that this encodes a successful run of $\mathcal{M}$ on input w.

Then $w \in \text{SAT}$ if, and only if, $G_w \models \varphi_{\mathcal{M}}$.

Hence, if "$G_w \models \varphi_{\mathcal{M}}$?" could be decided in time $f(|\varphi_{\mathcal{M}}|) \cdot |G_w|^c$ then "$w \in \text{SAT}$" could be decided in time

$$f(|\varphi_{\mathcal{M}}|) \cdot |G_w|^c \quad = \quad f(|\varphi_{\mathcal{M}}|) \cdot |w|^{2c} \quad = \quad \mathcal{O}(|w|^{2c}),$$

as $\mathcal{M}$ and hence $\varphi_{\mathcal{M}}$ is fixed. □

Intractability on Grids

Theorem. Let $\mathcal{G}$ be the class of coloured grids. Then $\mathsf{MC}(\mathrm{MSO}, \mathcal{G})$ is not fixed-parameter tractable unless P=NP.

Proof. We reduce SAT to $\mathsf{MC}(\mathrm{MSO}, \mathcal{G})$ as follows.

1. Given a propositional logic formula w of length n in CNF, construct an $n^2 \times n^2$-grid G_w and colour its bottom row by w.
2. Construct a formula $\varphi_{\mathcal{M}} \in \mathrm{MSO}$ which guesses a colouring of the grid and checks that this encodes a successful run of $\mathcal{M}$ on input w.

Then $w \in \text{SAT}$ if, and only if, $G_w \models \varphi_{\mathcal{M}}$.

Hence, if "$G_w \models \varphi_{\mathcal{M}}$?" could be decided in time $f(|\varphi_{\mathcal{M}}|) \cdot |G_w|^c$ then "$w \in \text{SAT}$" could be decided in time

$$f(|\varphi_{\mathcal{M}}|) \cdot |G_w|^c \quad = \quad f(|\varphi_{\mathcal{M}}|) \cdot |w|^{2c} \quad = \quad \mathcal{O}(|w|^{2c}),$$

as $\mathcal{M}$ and hence $\varphi_{\mathcal{M}}$ is fixed. □

Example: Grids

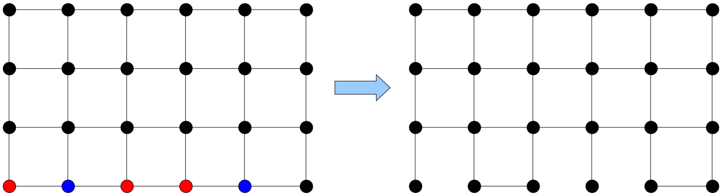
Theorem. Let $\mathcal{G}$ be the class of coloured grids. Then $\text{MC}(\text{MSO}, \mathcal{G})$ is not fixed-parameter tractable unless P=NP.

Theorem. Let $\mathcal{C}$ be the class of sub-graphs of grids. Then $\text{MC}(\text{MSO}, \mathcal{C})$ is not fixed-parameter tractable unless P=NP.

Example: Grids

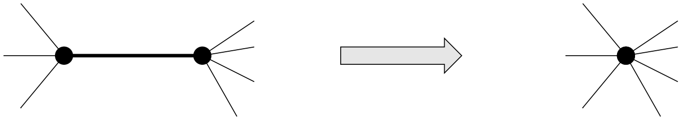
Theorem. Let $\mathcal{G}$ be the class of coloured grids. Then MC$(\mathrm{MSO}, \mathcal{G})$ is not fixed-parameter tractable unless P=NP.

Theorem. Let $\mathcal{C}$ be the class of sub-graphs of grids. Then MC$(\mathrm{MSO}, \mathcal{C})$ is not fixed-parameter tractable unless P=NP.



Minors

Definition. A graph H is a minor of G, denoted $H \preceq G$, if it can be obtained from a subgraph G' of G by contracting edges.

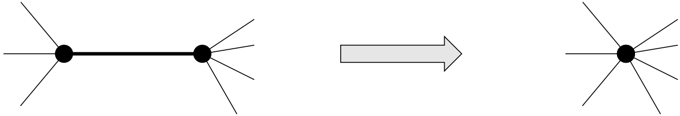


Equivalently. $H \preceq G$ if for every $v \in V(H)$ there is a connected $G_v \subseteq G$ such that

- if $u \neq v \in V(H)$ then $G_u \cap G_v = \emptyset$ and
- if $\{u, v\} \in E(H)$ then there is an edge in G joining G_u and G_v.

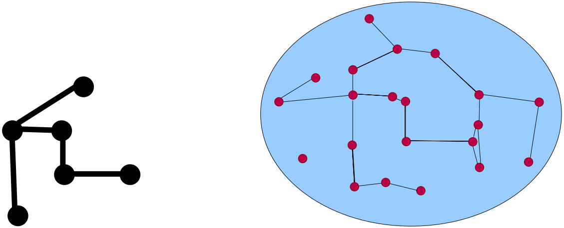
Minors

Definition. A graph H is a minor of G, denoted $H \preceq G$, if it can be obtained from a subgraph G' of G by contracting edges.



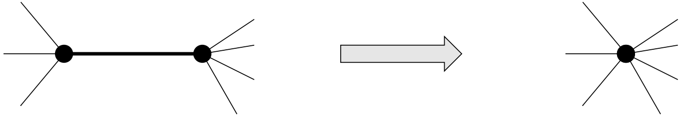
Equivalently. $H \preceq G$ if for every $v \in V(H)$ there is a connected $G_v \subseteq G$ such that

- if $u \neq v \in V(H)$ then $G_u \cap G_v = \emptyset$ and
- if $\{u, v\} \in E(H)$ then there is an edge in G joining G_u and G_v.



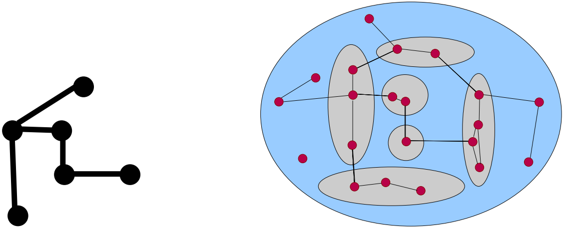
Minors

Definition. A graph H is a minor of G, denoted $H \preceq G$, if it can be obtained from a subgraph G' of G by contracting edges.



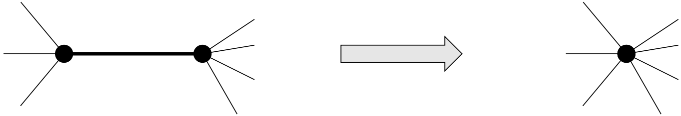
Equivalently. $H \preceq G$ if for every $v \in V(H)$ there is a connected $G_v \subseteq G$ such that

- if $u \neq v \in V(H)$ then $G_u \cap G_v = \emptyset$ and
- if $\{u, v\} \in E(H)$ then there is an edge in G joining G_u and G_v.



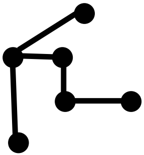
Minors

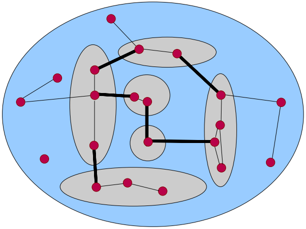
Definition. A graph H is a minor of G, denoted $H \preceq G$, if it can be obtained from a subgraph G' of G by contracting edges.



Equivalently. $H \preceq G$ if for every $v \in V(H)$ there is a connected $G_v \subseteq G$ such that

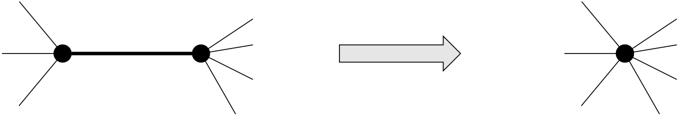
- if $u \neq v \in V(H)$ then $G_u \cap G_v = \emptyset$ and
- if $\{u, v\} \in E(H)$ then there is an edge in G joining G_u and G_v.





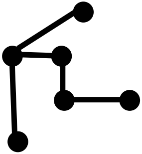
Minors

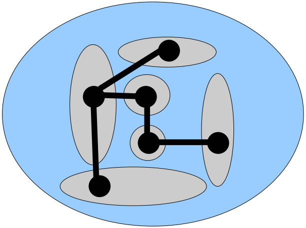
Definition. A graph H is a minor of G, denoted $H \preceq G$, if it can be obtained from a subgraph G' of G by contracting edges.



Equivalently. $H \preceq G$ if for every $v \in V(H)$ there is a connected $G_v \subseteq G$ such that

- if $u \neq v \in V(H)$ then $G_u \cap G_v = \emptyset$ and
- if $\{u, v\} \in E(H)$ then there is an edge in G joining G_u and G_v.





The Excluded Grid Theorem

Theorem. (Robertson, Seymour)

There is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that for all graphs G and all $k \in \mathbb{N}$, if $\mathrm{tw}(G) > f(k)$ then G contains a $k \times k$ grid as a minor.

Theorem. (Makowsky, Mariño 04)

If $\mathcal{C}$ is a class of graphs of unbounded tree-width closed under taking minors, then $\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$ is not fixed-parameter tractable unless P=NP.

Proof. As $\mathcal{C}$ has unbounded tree-width but is closed under taking minors, it contains all sub-graphs of grids. □

This result can be strengthened to closure under topological minors using walls instead of grids.

The Excluded Grid Theorem

Theorem. (Robertson, Seymour)

There is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that for all graphs G and all $k \in \mathbb{N}$, if $\mathrm{tw}(G) > f(k)$ then G contains a $k \times k$ grid as a minor.

Theorem. (Makowsky, Mariño 04)

If $\mathcal{C}$ is a class of graphs of unbounded tree-width closed under taking minors, then $\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$ is not fixed-parameter tractable unless P=NP.

Proof. As $\mathcal{C}$ has unbounded tree-width but is closed under taking minors, it contains all sub-graphs of grids. $\square$

This result can be strengthened to closure under topological minors using walls instead of grids.

Minors that look like grids

Recall: Main Result

Theorem. (K., Tazari 10)

Let $\mathcal{C}$ be a class of graph closed under sub-graphs and let $p(n)$ be a polynomial of degree $< \gamma$.

If the tree-width of $\mathcal{C}$ is $(\log^{28+\gamma} n, p)$-unbounded then $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is not fpt unless SAT can be solved in sub-exponential time.

(fpt: with parameter $|\varphi|$)

Definition. Let $f : \mathbb{N} \to \mathbb{N}$ be a function and $p(n)$ be a polynomial.

The tree-width of a class $\mathcal{C}$ is (f, p)-unbounded if there is an $\epsilon < 1$ such that for all $n \in \mathbb{N}$ there is a graph $G_n \in \mathcal{C}$ with

1. $n \leq \mathrm{tw}(G_n) \leq p(n)$ and $\mathrm{tw}(G_n) \geq f(|G_n|)$
2. given n, G_n can be constructed in time 2^{n^ϵ}.

The tree-width of $\mathcal{C}$ is f-unbounded if it is (f, p)-unbounded for some $p(n)$.

Good Idea, Sadly Wrong

First and wrong proof idea. Use the excluded grid theorem.

Theorem. (Robertson, Seymour)
There is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that all graphs of tree-width $\geq f(k)$ contain a $k \times k$-grid (as a minor).

Proof Idea: given a propositional logic formula w construct G_w so that G_w contains $|w|^2 \times |w|^2$-grid and proceed as before.

Problem. $f(n) := 20^{2 \cdot k^5}$ (Robertson, Seymour, Thomas)

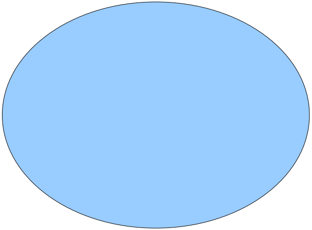
Good Idea, Sadly Wrong

First and wrong proof idea. Use the excluded grid theorem.

Theorem. (Robertson, Seymour)
There is a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$ such that all graphs of tree-width $\geq f(k)$ contain a $k \times k$-grid (as a minor).

Proof Idea: given a propositional logic formula w construct G_w so that G_w contains $|w|^2 \times |w|^2$-grid and proceed as before.

Problem. $f(n) := 20^{2 \cdot k^5}$ (Robertson, Seymour, Thomas)



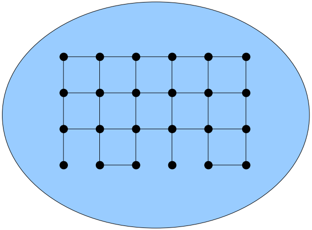
Good Idea, Sadly Wrong

First and wrong proof idea. Use the excluded grid theorem.

Theorem. (Robertson, Seymour)
There is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that all graphs of tree-width $\geq f(k)$ contain a $k \times k$-grid (as a minor).

Proof Idea: given a propositional logic formula w construct G_w so that G_w contains $|w|^2 \times |w|^2$-grid and proceed as before.

Problem. $f(n) := 20^{2 \cdot k^5}$ (Robertson, Seymour, Thomas)



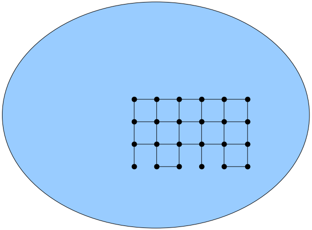
Good Idea, Sadly Wrong

First and wrong proof idea. Use the excluded grid theorem.

Theorem. (Robertson, Seymour)
There is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that all graphs of tree-width $\geq f(k)$ contain a $k \times k$-grid (as a minor).

Proof Idea: given a propositional logic formula w construct G_w so that G_w contains $|w|^2 \times |w|^2$-grid and proceed as before.

Problem. $f(n) := 20^{2 \cdot k^5}$ (Robertson, Seymour, Thomas)



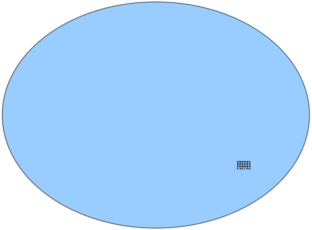
Good Idea, Sadly Wrong

First and wrong proof idea. Use the excluded grid theorem.

Theorem. (Robertson, Seymour)
There is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that all graphs of tree-width $\geq f(k)$ contain a $k \times k$-grid (as a minor).

Proof Idea: given a propositional logic formula w construct G_w so that G_w contains $|w|^2 \times |w|^2$-grid and proceed as before.

Problem. $f(n) := 20^{2 \cdot k^5}$ (Robertson, Seymour, Thomas)

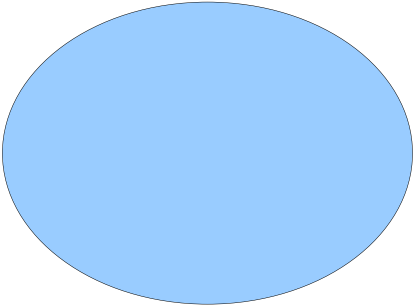


Grid-Like Minors

We will therefore use grid-like minors instead of grids.

Theorem. (Reed, Wood)

Any graph G of tree-width $\geq k^5$ contains two sets $\mathcal{P}, \mathcal{Q}$ of disjoint paths such that their intersection graph $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ contains a K_k-minor.

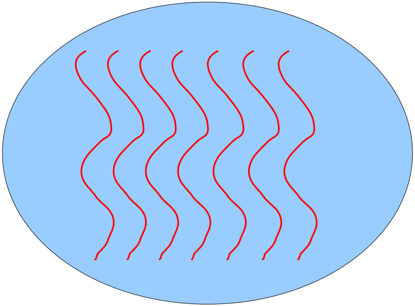


Grid-Like Minors

We will therefore use grid-like minors instead of grids.

Theorem. (Reed, Wood)

Any graph G of tree-width $\geq k^5$ contains two sets $\mathcal{P}, \mathcal{Q}$ of disjoint paths such that their intersection graph $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ contains a K_k-minor.

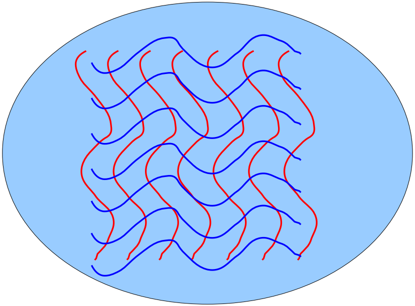


Grid-Like Minors

We will therefore use grid-like minors instead of grids.

Theorem. (Reed, Wood)

Any graph G of tree-width $\geq k^5$ contains two sets $\mathcal{P}, \mathcal{Q}$ of disjoint paths such that their intersection graph $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ contains a K_k-minor.

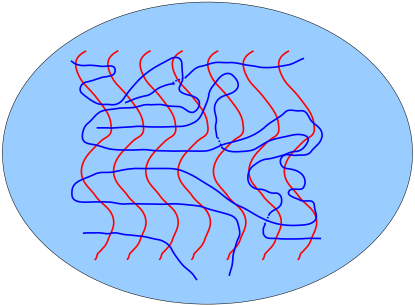


Grid-Like Minors

We will therefore use grid-like minors instead of grids.

Theorem. (Reed, Wood)

Any graph G of tree-width $\geq k^5$ contains two sets $\mathcal{P}, \mathcal{Q}$ of disjoint paths such that their intersection graph $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ contains a K_k-minor.

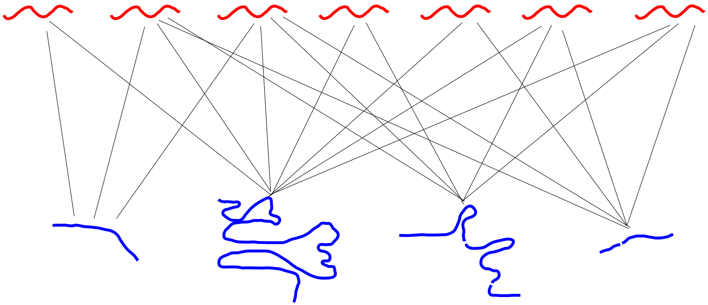


Grid-Like Minors

We will therefore use grid-like minors instead of grids.

Theorem. (Reed, Wood)

Any graph G of tree-width $\geq k^5$ contains two sets $\mathcal{P}, \mathcal{Q}$ of disjoint paths such that their intersection graph $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ contains a K_k-minor.



Grid-Like Minors

We will therefore use grid-like minors instead of grids.

Theorem. (Reed, Wood)

Any graph G of tree-width $\geq k^5$ contains two sets $\mathcal{P}, \mathcal{Q}$ of disjoint paths such that their intersection graph $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ contains a K_k-minor.

Theorem. (K., Tazari 10)

There is a constant c and a polynomial-time algorithm which, given a graph G with $\mathrm{tw}(G) > c \cdot k^{12}$, computes a (topological) grid-like minor of order k in G.

If we allow randomised algorithms we can reduce the tree-width to $\mathrm{tw}(G) > c' \cdot l^5$ to either

- find a model of K_l in G or
- a (top.) grid-like minor $(\mathcal{P}, \mathcal{Q})$ of order l.

Grid-Like Minors

We will therefore use grid-like minors instead of grids.

Theorem. (Reed, Wood)

Any graph G of tree-width $\geq k^5$ contains two sets $\mathcal{P}, \mathcal{Q}$ of disjoint paths such that their intersection graph $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ contains a K_k-minor.

Theorem. (K., Tazari 10)

There is a constant c and a polynomial-time algorithm which, given a graph G with $\mathrm{tw}(G) > c \cdot k^{12}$, computes a (topological) grid-like minor of order k in G.

If we allow randomised algorithms we can reduce the tree-width to $\mathrm{tw}(G) > c' \cdot l^5$ to either

- find a model of K_l in G or
- a (top.) grid-like minor $(\mathcal{P}, \mathcal{Q})$ of order l.

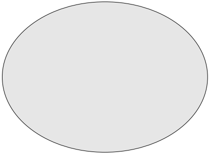
Another Good Idea, A Little Less Wrong

We reduce the propositional satisfiability problem (SAT) to $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$.

Given: $w := (X_1 \vee X_2 \vee \neg X_3) \wedge (X_4 \vee \neg X_5 \vee X_6)\ldots$
Problem: Decide if w is satisfiable.

Reduction.

Compute G_w
$|w|^{28} < \mathrm{tw}(G_w)$



MSO***-definable.***

There is MSO_2-formula $\varphi(P, Q)$ saying (P, Q) are grid-like minor.

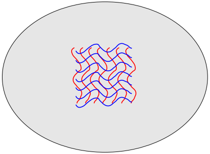
Another Good Idea, A Little Less Wrong

We reduce the propositional satisfiability problem (SAT) to $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$.

Given: $w := (X_1 \vee X_2 \vee \neg X_3) \wedge (X_4 \vee \neg X_5 \vee X_6) \ldots$
Problem: Decide if w is satisfiable.

Reduction.

Compute G_w
$|w|^{28} < \mathrm{tw}(G_w)$
Comp. GM $(\mathcal{P}, \mathcal{Q})$



MSO***-definable.***

There is MSO_2-formula $\varphi(P, Q)$ saying (P, Q) are grid-like minor.

Another Good Idea, A Little Less Wrong

We reduce the propositional satisfiability problem (SAT) to $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$.

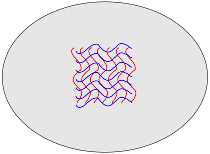
Given: $w := (X_1 \vee X_2 \vee \neg X_3) \wedge (X_4 \vee \neg X_5 \vee X_6)\ldots$
Problem: Decide if w is satisfiable.

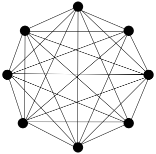
Reduction.

Compute G_w

$|w|^{28} < \mathrm{tw}(G_w)$ $\rightsquigarrow$ $\mathcal{I}(\mathcal{P}, \mathcal{Q})$

Comp. GM $(\mathcal{P}, \mathcal{Q})$





MSO-***definable.***

There is MSO_2-formula $\varphi(P, Q)$ saying (P, Q) are grid-like minor.

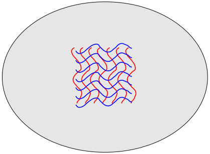
Another Good Idea, A Little Less Wrong

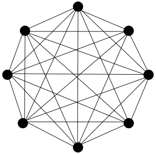
We reduce the propositional satisfiability problem (SAT) to $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$.

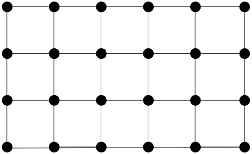
Given: $w := (X_1 \vee X_2 \vee \neg X_3) \wedge (X_4 \vee \neg X_5 \vee X_6) \ldots$
Problem: Decide if w is satisfiable.

Reduction.

Compute G_w
$|w|^{28} < \mathrm{tw}(G_w)$ $\rightsquigarrow$ $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ $\rightsquigarrow$ $G_{m,m}$ in $\mathcal{I}(\mathcal{P}, \mathcal{Q})$
Comp. GM $(\mathcal{P}, \mathcal{Q})$







MSO-***definable.***

There is MSO_2-formula $\varphi(P, Q)$ saying (P, Q) are grid-like minor.

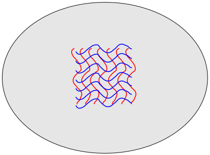
Another Good Idea, A Little Less Wrong

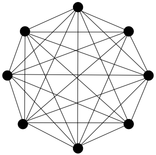
We reduce the propositional satisfiability problem (SAT) to $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$.

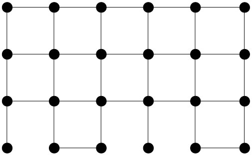
Given: $w := (X_1 \vee X_2 \vee \neg X_3) \wedge (X_4 \vee \neg X_5 \vee X_6)\ldots$
Problem: Decide if w is satisfiable.

Reduction.

Compute G_w
$|w|^{28} < \mathrm{tw}(G_w)$ $\rightsquigarrow$ $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ $\rightsquigarrow$ $G_{m,m}$ in $\mathcal{I}(\mathcal{P}, \mathcal{Q})$
Comp. GM $(\mathcal{P}, \mathcal{Q})$







MSO-***definable.***

There is MSO_2-formula $\varphi(P, Q)$ saying (P, Q) are grid-like minor.

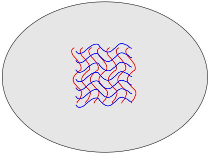
Another Good Idea, A Little Less Wrong

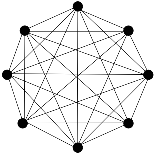
We reduce the propositional satisfiability problem (SAT) to $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$.

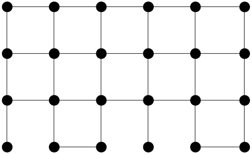
Given: $w := (X_1 \vee X_2 \vee \neg X_3) \wedge (X_4 \vee \neg X_5 \vee X_6)\ldots$
Problem: Decide if w is satisfiable.

Reduction.

Compute G_w
$|w|^{28} < \mathrm{tw}(G_w)$ $\rightsquigarrow$ $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ $\rightsquigarrow$ $G_{m,m}$ in $\mathcal{I}(\mathcal{P}, \mathcal{Q})$
Comp. GM $(\mathcal{P}, \mathcal{Q})$







MSO-***definable.***

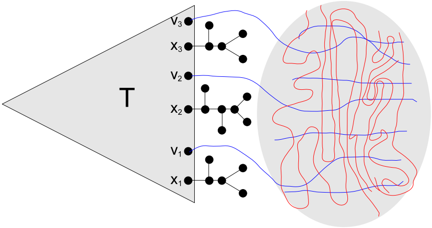
There is MSO_2-formula $\varphi(P, Q)$ saying (P, Q) are grid-like minor.

Catch. We cannot delete edges in $\mathcal{I}(\mathcal{P}, \mathcal{Q})$!

This means deleting vertices in G destroying the grid-like minors.

Labelled Tree-Ordered Webs Encoding Words

Definition. Labelled Tree-Ordered Webs encoding w

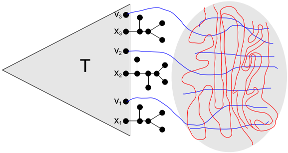


Theorem. (K., Tazari 10)

Let $w \in \{0,1\}^*$ be a word of length l and let d be a constant.

There is a constant c and a polynomial time algorithm which, given a graph G of tree-width $\geq cl^{14d}$ computes a sub-graph $G_w \subseteq G$ which is a labelled tree-ordered web encoding w (with power d).

Defining Labelled Tree-Ordered Webs

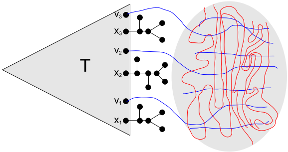


1. T is uniquely MSO definable in $G' \subseteq G$ by $\varphi_T(V, E)$
2. The order defined by T is MSO-definable by $\varphi_\leq$
3. The encoding of w is MSO-definable by φ_0, φ_1
4. The grid-like minor $(\mathcal{P}, \mathcal{Q})$ is MSO-definable by $\varphi(P, Q)$
5. A grid (wall) as sub-graph of $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ respecting the order of T is MSO-definable by $\varphi(H, V, P, Q)$

Theorem. There is an MSO_2-formula φ such that for every labelled tree-ordered web H encoding a SAT-instance w

$$H_w \models \varphi \quad \Longleftrightarrow \quad w \text{ is satisfiable}$$

Defining Labelled Tree-Ordered Webs



1. T is uniquely MSO definable in $G' \subseteq G$ by $\varphi_T(V, E)$
2. The order defined by T is MSO-definable by $\varphi_{\leq}$
3. The encoding of w is MSO-definable by φ_0, φ_1
4. The grid-like minor $(\mathcal{P}, \mathcal{Q})$ is MSO-definable by $\varphi(P, Q)$
5. A grid (wall) as sub-graph of $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ respecting the order of T is MSO-definable by $\varphi(H, V, P, Q)$

Theorem. There is an MSO_2-formula φ such that for every labelled tree-ordered web H encoding a SAT-instance w

$$H_w \models \varphi \qquad \Longleftrightarrow \qquad w \text{ is satisfiable}$$

Main Result

Theorem. (K., Tazari 10)

Let $\mathcal{C}$ be a class of graph closed under sub-graphs and let $p(n)$ be a polynomial of degree $< \gamma$.

If the tree-width of $\mathcal{C}$ is $(\log^{28+\gamma} n, p)$-unbounded then $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is not fpt unless SAT can be solved in sub-exponential time.

Proof. For simplicity, assume $\gamma = 0$.

Given SAT-instance $w \in \{0,1\}^*$ of length l.

1. As the tree-width of $\mathcal{C}$ is $(\log^{28+\gamma} n, p)$-unbounded, there is $\epsilon < 1$ such that $\mathcal{C}$ contains $G \in \mathcal{C}$ with $\mathrm{tw}(G) > \log^{28}|G|$ and $\mathrm{tw}(G) = 2cl^{28}$.
2. Compute G in time $2^{|w|^\epsilon}$. This implies $|G| \leq 2^{|w|^\delta}$ for some $\delta < 1$.
3. Compute in pol. time a labelled tree-ordered web $H \subseteq G$ encoding w.

   $H \models \varphi$ iff w is satisfiable.
4. Hence, if $H \models \varphi$ was decidable in

$$|H|^{f(|\varphi|)} \leq |G|^{f(|\varphi|)} \leq (2^{|w|^\delta})^{f(|\varphi|)} = 2^{f(|\varphi|)|w|^\delta} = 2^{o(|w|)}$$

The Gap

Comparing our result with Courcelle's theorem, there is a gap.

- If the tree-width of $\mathcal{C}$ is $(\log^{28+\gamma} n, p)$-unbounded then $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is not fpt unless SAT can be solved in sub-exponential time.
- The model-checking problem $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is fixed-parameter tractable on any class $\mathcal{C}$ of graphs of bounded tree-width.

(Courcelle '90)

What can we say about the gap?

It seems impossible to close the gap.

- Makowsky and Mariño give classes of graphs closed under sub-graphs of logarithmic tree-width with tractable MSO-model-checking.
- There are examples of classes of graphs closed under sub-graphs of logarithmic tree-width where it becomes intractable (presumably).

The Gap

Comparing our result with Courcelle's theorem, there is a gap.

- If the tree-width of $\mathcal{C}$ is $(\log^{28+\gamma} n, p)$-unbounded then $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is not fpt unless SAT can be solved in sub-exponential time.

- The model-checking problem $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is fixed-parameter tractable on any class $\mathcal{C}$ of graphs of bounded tree-width.

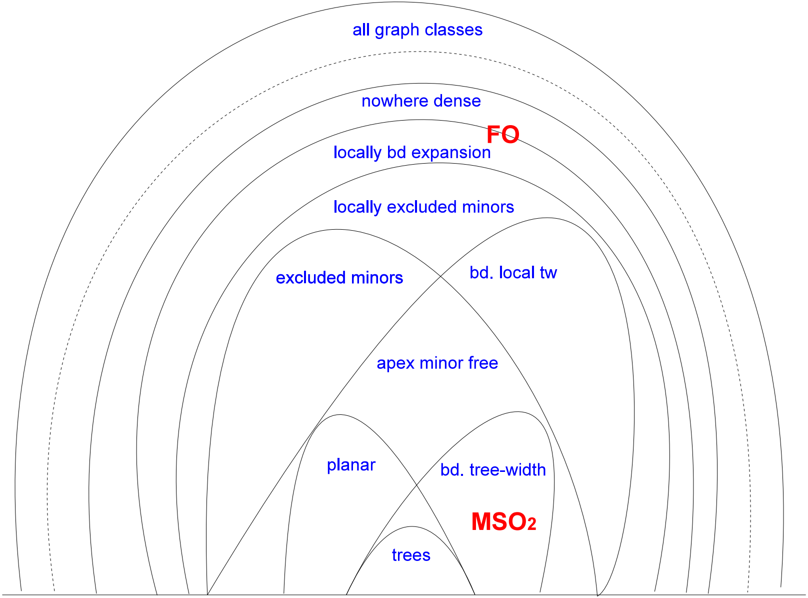
(Courcelle '90)

What can we say about the gap?

It seems impossible to close the gap.

- Makowsky and Mariño give classes of graphs closed under sub-graphs of logarithmic tree-width with tractable MSO-model-checking.
- There are examples of classes of graphs closed under sub-graphs of logarithmic tree-width where it becomes intractable (presumably).

An Overview of Graph Parameters



Lower Bounds for First-Order Logic

Theorem. (Dvořák, Kral, Thomas 10)

First-Order Model-Checking is fpt on any class of graphs of (locally) bounded expansion.

Theorem. (K. 09)

If $\mathcal{C}$ is not nowhere dense, closed under sub-graphs and satisfies some technical condition, then $\text{MC}(\text{FO}, \mathcal{C})$ is not fpt unless P=NP.

Lower Bounds for First-Order Logic

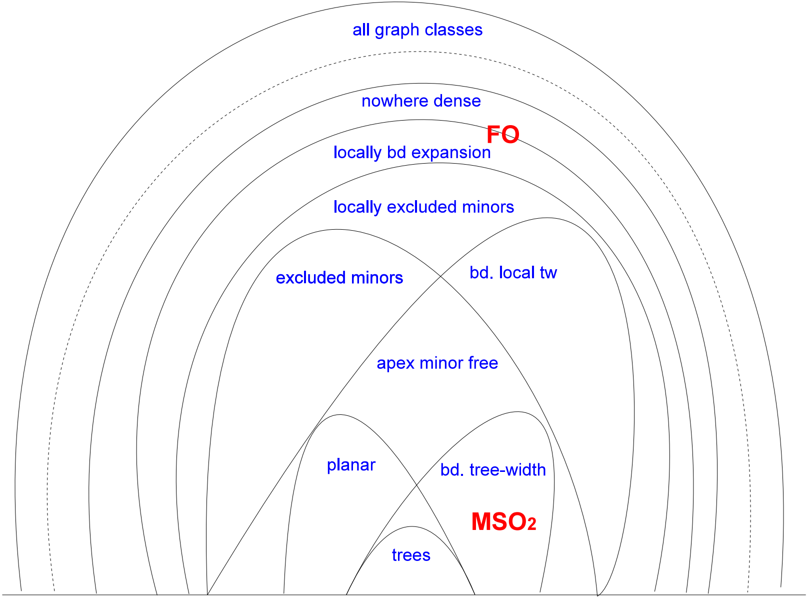
Theorem. (Dvořák, Kral, Thomas 10)

First-Order Model-Checking is fpt on any class of graphs of (locally) bounded expansion.

Theorem. (K. 09)

If $\mathcal{C}$ is not nowhere dense, closed under sub-graphs and satisfies some technical condition, then $\mathrm{MC}(\mathrm{FO}, \mathcal{C})$ is not fpt unless P=NP.

An Overview of Graph Parameters



Conclusion

Structural Characterisation of Model-Checking Problems

Research programme. For each of the natural logics $\mathcal{L}$ such as FO or MSO, identify a structural property $\mathcal{P}$ of classes $\mathcal{C}$ of graphs such that $\text{MC}(\mathcal{L}, \mathcal{C})$ is tractable if, and only if, $\mathcal{C}$ has the property $\mathcal{P}$ under suitable complexity theoretical assumptions.

We may not always get an exact characterisation, there may be gaps.

But such a characterisation would give an easy tool to assess whether MSO-model-checking is tractable on some class.