Complexity of First- and Monadic Second-Order Logic

Stephan Kreutzer

Oxford University Computing Laboratory

MODNET Training Workshop – Model Theory and Applications
La Roche-en-Ardenne
20-25 April 2008

Introduction

Yesterday's tutorial:

- General introduction to Finite Model Theory
- Preservation theorems
- Well-behaved classes of finite structures
- Model-Checking or Evaluation Problem for First-Order Logic
  - Naïve algorithm for evaluating FO formulas
  - PSPACE-completeness
- Important tools: Ehrenfeucht-Fraïssé Games, Locality Theorems

In this talk: Model-Checking and Satisfiability for

- First-Order Logic
- Monadic Second-Order Logic

Focus: Efficient methods on well-behaved classes of structures

Introduction

Yesterday's tutorial:

- General introduction to Finite Model Theory
- Preservation theorems
- Well-behaved classes of finite structures
- Model-Checking or Evaluation Problem for First-Order Logic
  - Naïve algorithm for evaluating FO formulas
  - PSPACE-completeness
- Important tools: Ehrenfeucht-Fraïssé Games, Locality Theorems

In this talk: Model-Checking and Satisfiability for

- First-Order Logic
- Monadic Second-Order Logic

Focus: Efficient methods on well-behaved classes of structures

Introduction

In this talk: Model-Checking and Satisfiability for

- First-Order Logic
- Monadic Second-Order Logic

Main methods used:

- Feferman-Vaught style decomposition theorems
- Locality theorems (Gaifman locality)
- Interest in effective versions (complexity)

- Automata based methods
- Extension to infinite structures

A word of warning.

- This talk focuses on the logical methods underlying the results.
- Results from graph theory used are often very deep and sometimes much more involved than the logical methods.

Introduction

In this talk: Model-Checking and Satisfiability for

- First-Order Logic
- Monadic Second-Order Logic

Main methods used:

- Feferman-Vaught style decomposition theorems
- Locality theorems (Gaifman locality)
- Interest in effective versions (complexity)

- Automata based methods
- Extension to infinite structures

A word of warning.

- This talk focuses on the logical methods underlying the results.
- Results from graph theory used are often very deep and sometimes much more involved than the logical methods.

Recall: First-Order Model Checking

First-Order Model Checking:

Given: Finite structure $\mathfrak{A} := (A, \sigma)$
First-order formula φ

Problem: Decide $\mathfrak{A} \models \varphi$

We will often restrict the class of admissible structures.

Let $\mathcal{C}$ be a class of finite structures.

First-Order Model Checking on $\mathcal{C}$:

Given: Finite structure $\mathfrak{A} := (A, \sigma) \in \mathcal{C}$
First-order formula φ

Problem: Decide $\mathfrak{A} \models \varphi$

Definition:

MC(FO): FO-model-checking on the class of all finite structures
MC(FO, $\mathcal{C}$): FO-model-checking on the class of all structures $\mathfrak{A} \in \mathcal{C}$
MC(φ, $\mathcal{C}$): FO-model-checking for φ on the class $\mathfrak{A} \in \mathcal{C}$

Recall: First-Order Model Checking

First-Order Model Checking:

Given: Finite structure $\mathfrak{A} := (A, \sigma)$
First-order formula φ

Problem: Decide $\mathfrak{A} \models \varphi$

We will often restrict the class of admissible structures.

Let $\mathcal{C}$ be a class of finite structures.

First-Order Model Checking on $\mathcal{C}$:

Given: Finite structure $\mathfrak{A} := (A, \sigma) \in \mathcal{C}$
First-order formula φ

Problem: Decide $\mathfrak{A} \models \varphi$

Definition:

MC(FO): FO-model-checking on the class of all finite structures
MC(FO, $\mathcal{C}$): FO-model-checking on the class of all structures $\mathfrak{A} \in \mathcal{C}$
MC(φ, $\mathcal{C}$): FO-model-checking for φ on the class $\mathfrak{A} \in \mathcal{C}$

Recall: First-Order Model Checking

First-Order Model Checking:

Given: Finite structure $\mathfrak{A} := (A, \sigma)$
First-order formula φ

Problem: Decide $\mathfrak{A} \models \varphi$

We will often restrict the class of admissible structures.

Let $\mathcal{C}$ be a class of finite structures.

First-Order Model Checking on $\mathcal{C}$:

Given: Finite structure $\mathfrak{A} := (A, \sigma) \in \mathcal{C}$
First-order formula φ

Problem: Decide $\mathfrak{A} \models \varphi$

Definition:

MC(FO): FO-model-checking on the class of all finite structures
MC(FO, $\mathcal{C}$): FO-model-checking on the class of all structures $\mathfrak{A} \in \mathcal{C}$
MC(φ, $\mathcal{C}$): FO-model-checking for φ on the class $\mathfrak{A} \in \mathcal{C}$

Complexity of First-Order Model Checking

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

Naïve algorithm: Evaluation following the structure of the formula

- Existential quantification: $\varphi := \exists x \psi$

  for all $a \in A$ check whether

  $$(\mathfrak{A}, c \mapsto a) \models \varphi[x/c]$$

  where c is a new constant symbol.
- Boolean connectives: easy
- Atomic formulae: direct look up in the structure

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 $\mathfrak{A}$

Complexity of First-Order Model Checking

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

Naïve algorithm: Evaluation following the structure of the formula

- Existential quantification: $\varphi := \exists x \psi$

  for all $a \in A$ check whether

$$(\mathfrak{A}, c \mapsto a) \models \varphi[x/c]$$

  where c is a new constant symbol.
- Boolean connectives: easy
- Atomic formulae: direct look up in the structure

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 $\mathfrak{A}$

Complexity of First-Order Model-Checking

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 even for a fixed two element structure $\mathfrak{A}$.

(Reduce satisfiability for Quantified Boolean Formulae to FO Model-Checking)

Theorem. For any fixed φ, (data complexity)

$\text{MC}(\varphi, \text{Str}) \in$ PTIME and $\text{MC}(\varphi, \text{Str}) \in$ LOGSPACE

I.e. checking whether φ is true in a finite structure is in polynomial time and logarithmic space.

Complexity of First-Order Model-Checking

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 even for a fixed two element structure $\mathfrak{A}$.

(Reduce satisfiability for Quantified Boolean Formulae to FO Model-Checking)

Theorem. For any fixed φ, (data complexity)

$\text{MC}(\varphi, \text{Str}) \in$ PTIME and $\text{MC}(\varphi, \text{Str}) \in$ LOGSPACE

I.e. checking whether φ is true in a finite structure is in polynomial time and logarithmic space.

Parameterized Complexity

Theorem. For any fixed φ, checking whether φ is true in a finite structure is in polynomial time and logarithmic space.

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

This is bad unless φ is really small.

Better for moderately large φ:

$\mathcal{O}(2^{|\varphi|} \cdot |\mathfrak{A}|)$ or

$f(|\varphi|) \cdot |\mathfrak{A}|^c$ for some computable function f and fixed $c \in \mathbb{N}$.

Parameterized Complexity

Theorem. For any fixed φ, checking whether φ is true in a finite structure is in polynomial time and logarithmic space.

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

This is bad unless φ is really small.

Better for moderately large φ:

$\mathcal{O}(2^{|\varphi|} \cdot |\mathfrak{A}|)$ or

$f(|\varphi|) \cdot |\mathfrak{A}|^c$ for some computable function f and fixed $c \in \mathbb{N}$.

Parameterized Complexity

We look at parameterized problems of the form:

$p-\mathrm{MC}(\mathrm{FO}, \mathcal{C})$

Input:	Structure $\mathfrak{A} \in \mathcal{C}$, $\varphi \in \mathrm{FO}$.
Parameter:	$k := \|\varphi\|$.
Problem:	Decide $\mathfrak{A} \models \varphi$.

A problem is fixed-parameter tractable (fpt) if it can be solved in time

$$f(k) \cdot |\mathfrak{A}|^{\mathcal{O}(1)}$$

where $f : \mathbb{N} \to \mathbb{N}$ is any computable function.

Theorem. FO model checking is not fixed-parameter tractable on the class of all finite structures.

(under reasonable assumptions from complexity theory).

Identify classes of structures, where FO model checking is fpt

Parameterized Complexity

We look at parameterized problems of the form:

$p-\mathrm{MC}(\mathrm{FO}, \mathcal{C})$

Input:	Structure $\mathfrak{A} \in \mathcal{C}$, $\varphi \in \mathrm{FO}$.
Parameter:	$k := \lvert\varphi\rvert$.
Problem:	Decide $\mathfrak{A} \models \varphi$.

A problem is fixed-parameter tractable (fpt) if it can be solved in time

$$f(k) \cdot \lvert\mathfrak{A}\rvert^{\mathcal{O}(1)}$$

where $f : \mathbb{N} \to \mathbb{N}$ is any computable function.

Theorem. FO model checking is not fixed-parameter tractable on the class of all finite structures.

(under reasonable assumptions from complexity theory).

Identify classes of structures, where FO model checking is fpt

Parameterized Complexity

We look at parameterized problems of the form:

$p-\mathrm{MC}(\mathrm{FO}, \mathcal{C})$	
Input:	Structure $\mathfrak{A} \in \mathcal{C}$, $\varphi \in \mathrm{FO}$.
Parameter:	$k := \|\varphi\|$.
Problem:	Decide $\mathfrak{A} \models \varphi$.

A problem is fixed-parameter tractable (fpt) if it can be solved in time

$$f(k) \cdot |\mathfrak{A}|^{\mathcal{O}(1)}$$

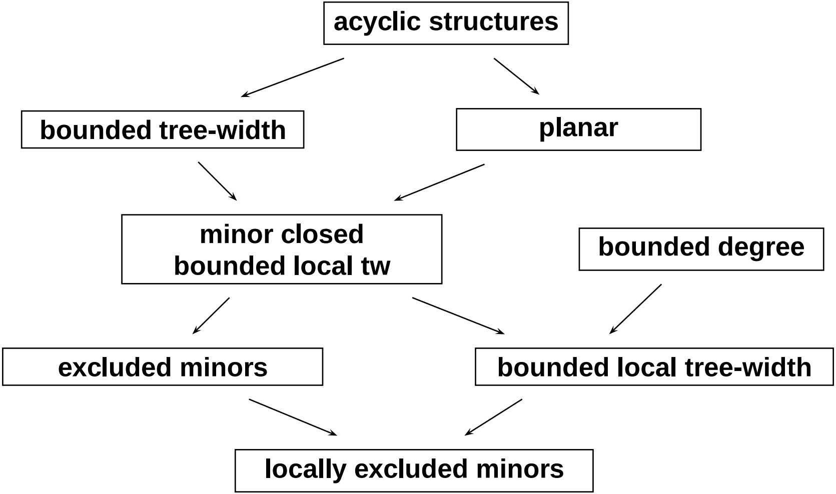
where $f : \mathbb{N} \to \mathbb{N}$ is any computable function.

Theorem. FO model checking is not fixed-parameter tractable on the class of all finite structures.

(under reasonable assumptions from complexity theory).

Identify classes of structures, where FO model checking is fpt

Well-Behaved Classes of Graphs

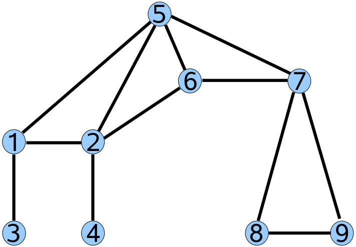


Graph Classes of Bounded Tree-Width

Tree-Width

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

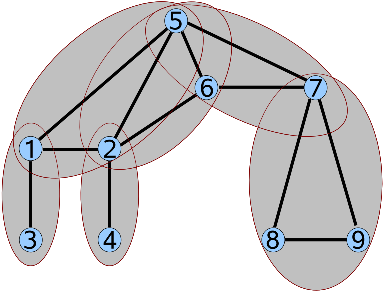
A graph has tree-width $\leq k$ if it can be covered by sub-graphs of size $\leq (k+1)$ in a tree-like fashion.



Tree-Width

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

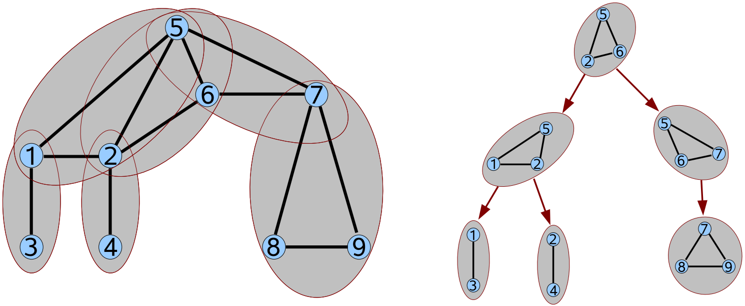
A graph has tree-width $\leq k$ if it can be covered by sub-graphs of size $\leq (k+1)$ in a tree-like fashion.



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.

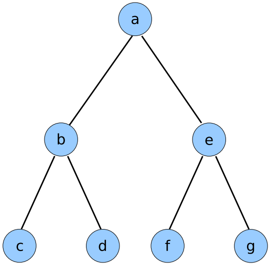
The **width** of $\mathcal{T}$ is $\max\{|B_t| - 1 : t \in V(T)\}$

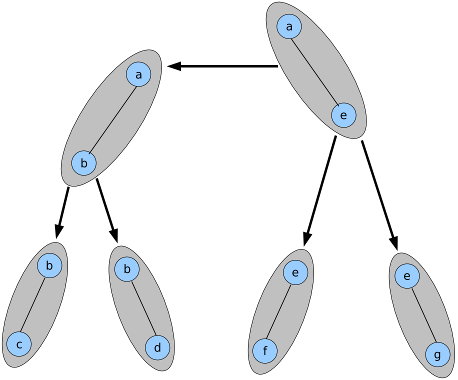
The **tree-width** $\mathrm{tw}(G)$ of G is the minimal width of any of its tree-dec.

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**

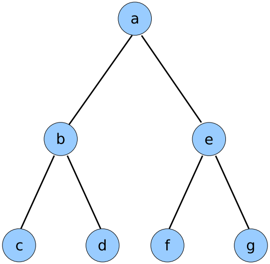


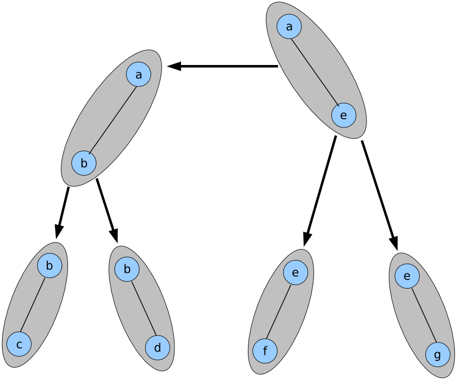


Proposition: Acyclic graphs are precisely the graphs of tree-width 1.

Examples

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

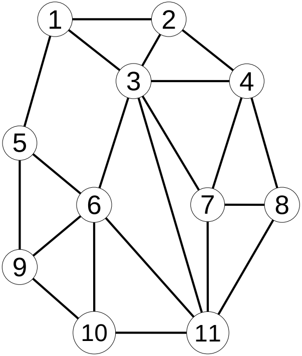


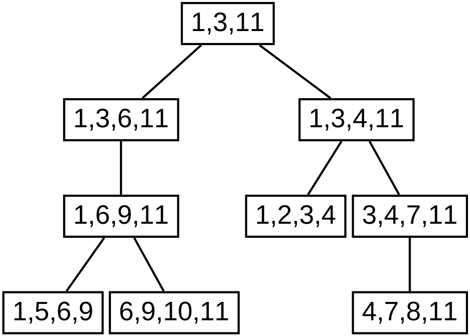


Proposition: Acyclic graphs are precisely the graphs of tree-width 1.

Examples

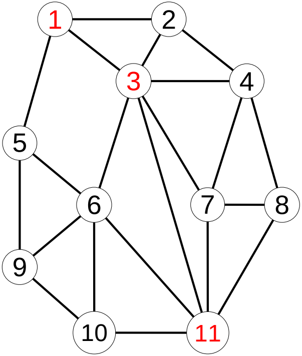
Example 2:

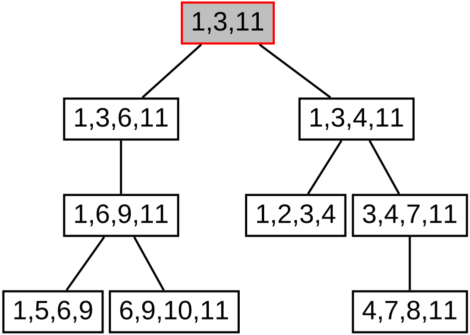




Examples

Example 2:

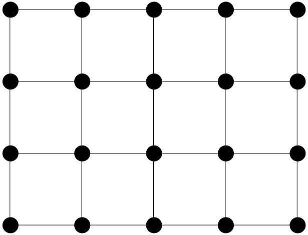




Examples

Example 3: Grids

(4×5)-grid



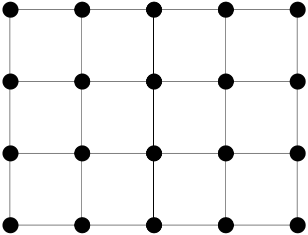
Lemma: For all $k > 1$, the $(n \times m)$-grid has tree-width $\min\{n, m\}$.

Theorem: (Excluded Grid Theorem) (Robertson, Seymour)
There is a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$, such that every graph of tree-width at least $f(k)$ has a $(k \times k)$-grid as a minor.

Examples

Example 3: Grids

(4×5)-grid



Lemma: For all $k > 1$, the $(n \times m)$-grid has tree-width $\min\{n, m\}$.

Theorem: (Excluded Grid Theorem) (Robertson, Seymour)

There is a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$, such that every graph of tree-width at least $f(k)$ has a $(k \times k)$-grid as a minor.

Courcelle's Theorem

Theorem: (Courcelle 1990)

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

MC(MSO, $\mathcal{C}$)	
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).

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

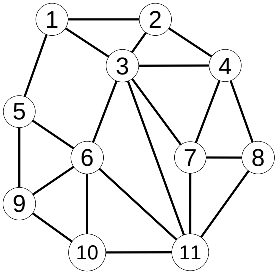
$$\mathcal{O}(2^{\text{tw}(G)^3}|G|).$$

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: Separators

$\mathcal{C}$: Class $\mathcal{C}$ of graphs of bounded tree-width
φ: Fixed MSO-sentence

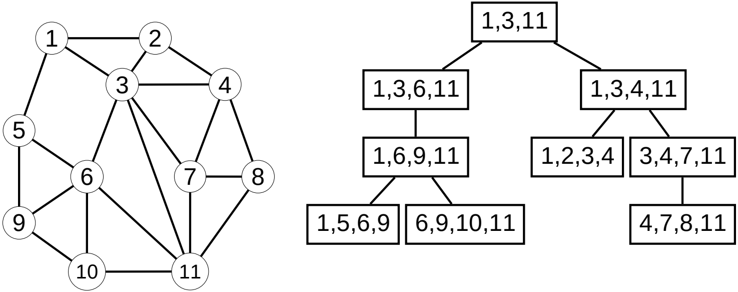
Given: graph $G \in \mathcal{C}$
Problem: decide whether $G \models \varphi$



Second Ingredient: Separators

$\mathcal{C}$: Class $\mathcal{C}$ of graphs of bounded tree-width
φ: Fixed MSO-sentence

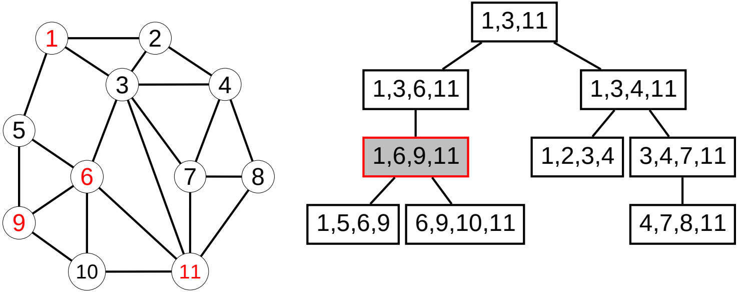
Given: graph $G \in \mathcal{C}$
Problem: decide whether $G \models \varphi$



Second Ingredient: Separators

$\mathcal{C}$: Class $\mathcal{C}$ of graphs of bounded tree-width
φ: Fixed MSO-sentence

Given: graph $G \in \mathcal{C}$
Problem: decide whether $G \models \varphi$



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

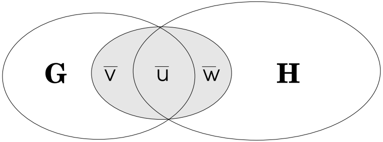
$\overline{v} \in V(G) \qquad \overline{w} \in V(H)$

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

For all $q \geq 0$,

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

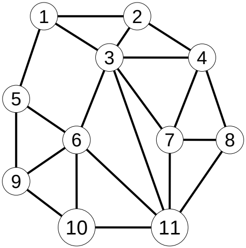
Furthermore, there is an algorithm that computes $\text{tp}_q(G \cup H, \overline{uvw})$ from $\text{tp}_q(G, \overline{uv})$ and $\text{tp}_q(\overline{uw})$.



Courcelle's Theorem: Algorithm

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

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

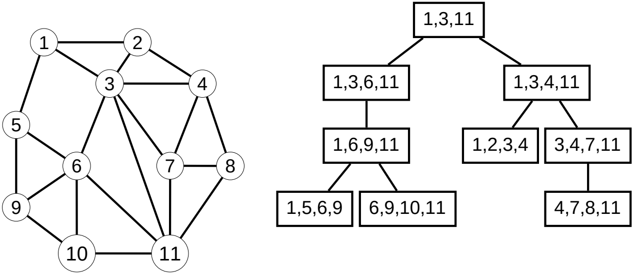


Courcelle's Theorem: Algorithm

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

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

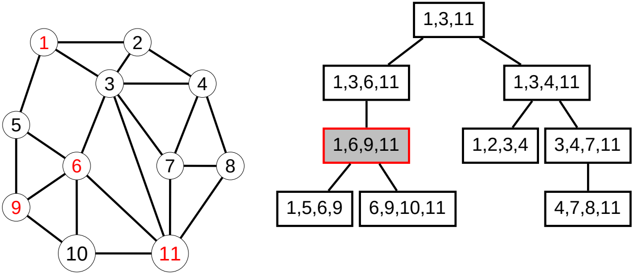


Courcelle's Theorem: Algorithm

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

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



Courcelle's Theorem

Theorem: (Courcelle 1990)

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

MC(MSO, $\mathcal{C}$)
Input: Graph $G \in \mathcal{C}$, $\varphi \in \mathrm{MSO}$.
Parameter: $|\varphi|$.
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

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

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

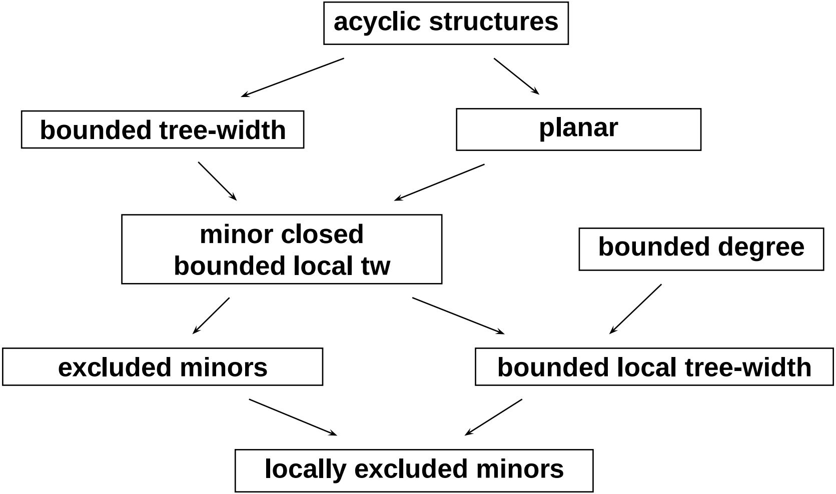
is fixed-parameter tractable (linear time for each fixed φ).

What about the parameter dependence?

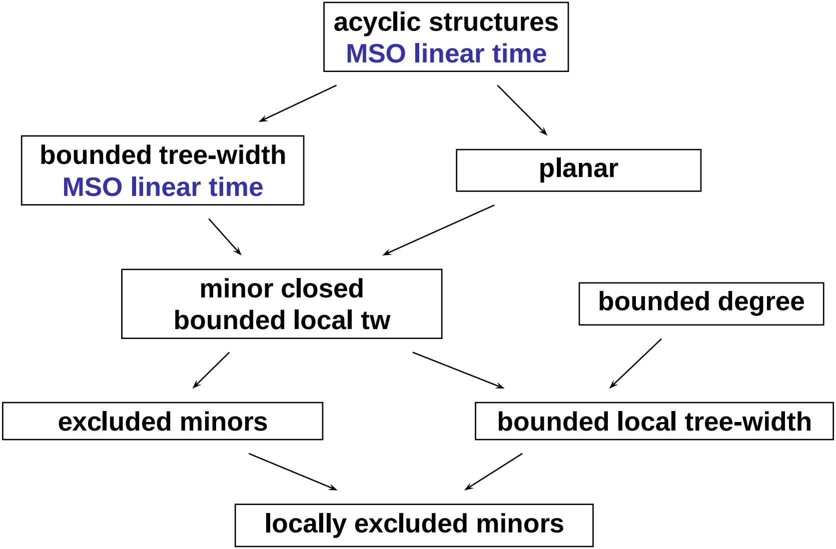
Theorem: (Frick, Grohe, 01)

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.

Well-Behaved Classes of Graphs



Well-Behaved Classes of Graphs



Beyond Bounded Tree-Width

Can Courcelle's theorem be generalised to other classes of graphs?

Theorem: (Garey, Johnson, Stockmeyer, 1976)

3-COLOURABILITY is NP-complete on planar graphs of degree at most 4.

Corollary:

MSO model-checking is NP-hard on the class of planar graphs and the class of graphs of degree at most k for some $k \geq 4$.

What about first-order model-checking?

Beyond Bounded Tree-Width

Can Courcelle's theorem be generalised to other classes of graphs?

Theorem: (Garey, Johnson, Stockmeyer, 1976)

3-Colourability is NP-complete on planar graphs of degree at most 4.

Corollary:

MSO model-checking is NP-hard on the class of planar graphs and the class of graphs of degree at most k for some $k \geq 4$.

What about first-order model-checking?

First-Order Model Checking

Bounded Degree Graphs

Theorem: (Seese, 1996)

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

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

(linear time ftp-algorithm)

Proof. (by Frick and Grohe.)

The proof is based on locality of first-order logic.

Locality of First-Order Logic

Notation: Let G be a graph.

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

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

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

Definition:

A formula $\varphi(x) \in \mathrm{FO}$ 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:

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

Remark: Gaifman's proof is constructive.

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

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

(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:

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

Remark: Gaifman's proof is constructive.

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

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

(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

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

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 $d \geq 1$.

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

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

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 such that $G \models \varphi$.

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

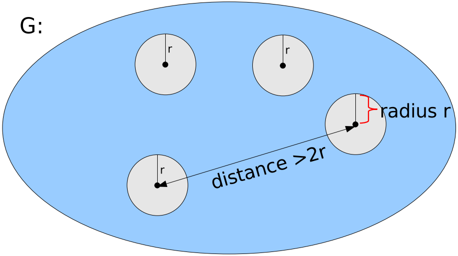
Proof of Theorem

Suppose

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

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

Let G be a graph of maximum degree d such that $G \models \varphi$.



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

First Step

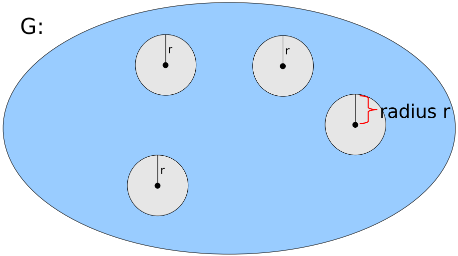
Algorithm: First Step

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

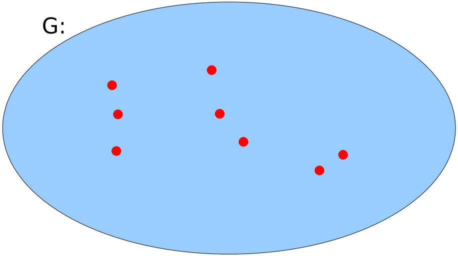
Algorithm: First Step

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: $\mathcal{O}(n)$



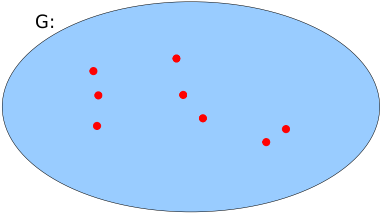
Second Step: Greedy Approach

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

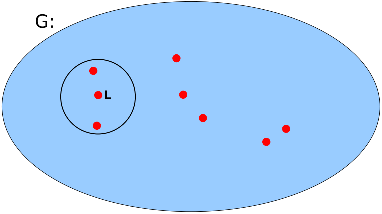
else

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

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

accept **else** reject

Assume $m = 4$



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

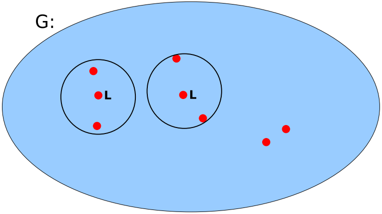
Second Step: Greedy Approach

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 $v \in Q$

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

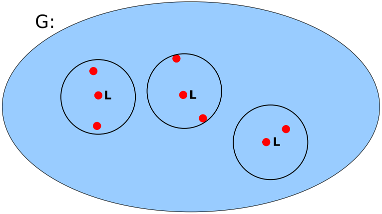
$Q := Q \setminus N_{2r}(v)$

od

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

else

Assume $m = 4$



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

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

accept **else** reject

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

Second Step: Greedy Approach

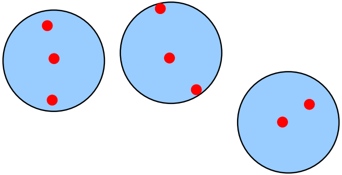
Let Q be the set of red vertices.

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

Assume $m = 4$

G':



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

Second Step: Greedy Approach

Let Q be the set of red vertices.

Algorithm: Second Step

$L := \emptyset$

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

choose $v \in Q$

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

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

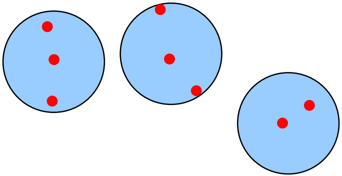
od

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

else

Assume $m = 4$

G':



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

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:	$\|\varphi\|$.
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-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

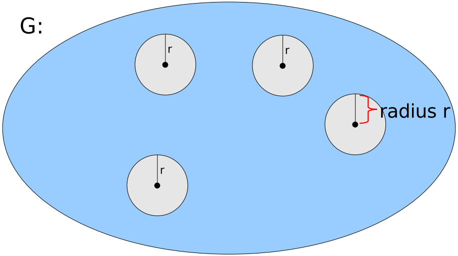
Algorithm: First Step

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

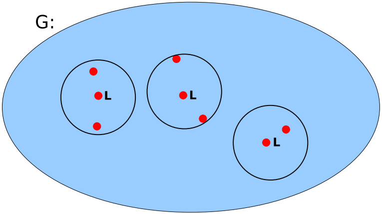
if $|L| \geq m$ **then** accept

else

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

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

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

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

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

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 $f : \text{GRAPHS} \rightarrow \mathbb{N}$.

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

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

Bounded local tree-width

Localisation of Graph Invariants

Definition:

Let $f : \textsc{Graphs} \to \mathbb{N}$ be a graph invariant.

We define its localisation $loc_f : \textsc{Graphs} \times \mathbb{N} \to \mathbb{N}$ 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:

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

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

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:	$\lvert\varphi\rvert + f(G)$.
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:	$\varphi \in$ 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

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

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

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

Theorem: (Frick, 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.

But let's not get carried away: In Frick, Grohe's theorem, the exponent depends on the excluded minor.

For locally excluded minors, the excluded minor must be made part of the parameter.
This requires significant work in the graph theoretical part of the proof.

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)
- graphs of locally bounded clique-width

Theorem: (Frick, 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.

But let's not get carried away: In Frick, Grohe's theorem, the exponent depends on the excluded minor.

For locally excluded minors, the excluded minor must be made part of the parameter.
This requires significant work in the graph theoretical part of the proof.

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)
- graphs of locally bounded clique-width

Theorem: (Frick, 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.

But let's not get carried away: In Frick, Grohe's theorem, the exponent depends on the excluded minor.

For locally excluded minors, the excluded minor must be made part of the parameter.

This requires significant work in the graph theoretical part of the proof.

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)
- graphs of locally bounded clique-width

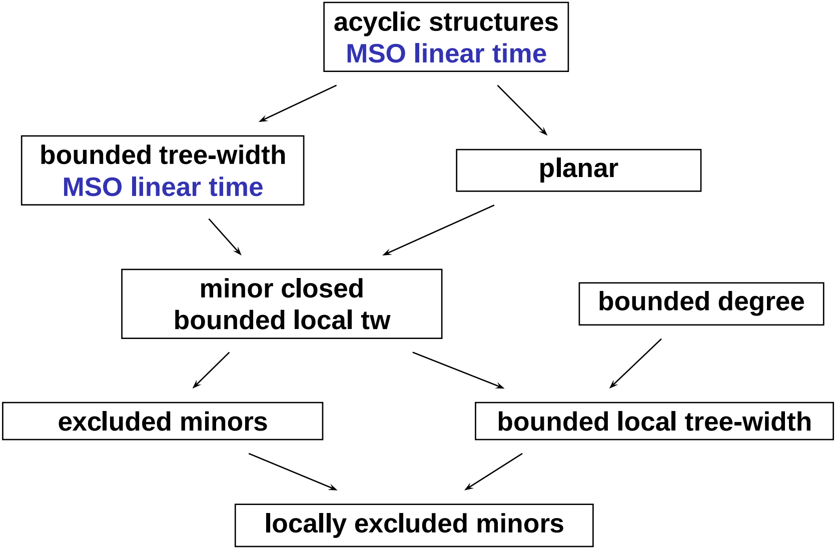
Theorem: (Frick, Grohe 01)
First-order model-checking is fixed-parameter tractable on graph classes excluding a minor.

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

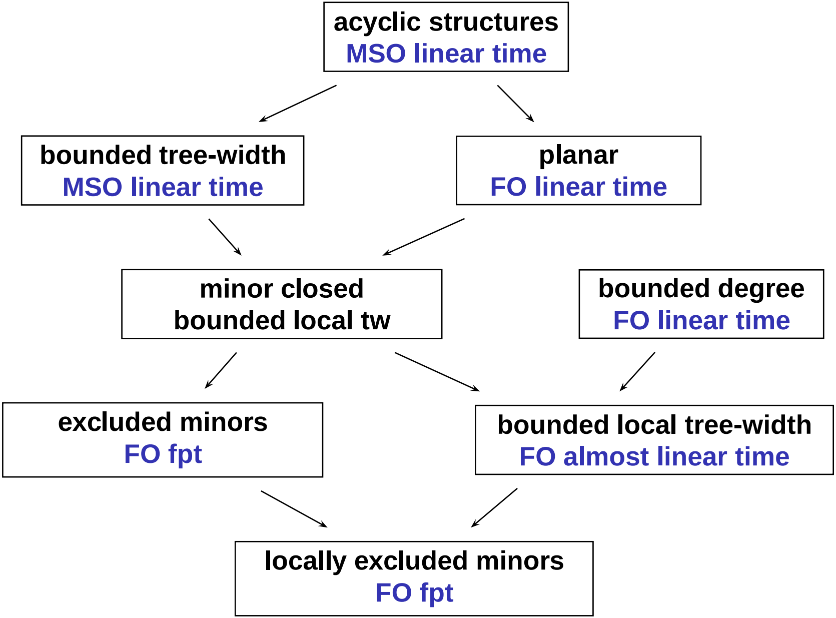
But let's not get carried away: In Frick, Grohe's theorem, the exponent depends on the excluded minor.

For locally excluded minors, the excluded minor must be made part of the parameter.
This requires significant work in the graph theoretical part of the proof.

Well-Behaved Classes of Graphs



Well-Behaved Classes of Graphs



Further Work

Satisfiability

Satisfiability problem:
Let $\mathcal{C}$ be a graph of structures

SAT(FO, $\mathcal{C}$), SAT(MSO, $\mathcal{C}$)	
Input:	MSO- or FO-sentence φ.
Problem:	Does φ have a model in $\mathcal{C}$?.

Trakhtenbrot's theorem: SAT(FO, FIN), i.e. satisfiability in the finite, is undecidable.

For $k \in \mathbb{N}$, let $\mathfrak{T}_k$ be the class of finite graphs of tree-width $\leq k$.

Theorem. (Seese)

For any $k \geq 0$, SAT(MSO, $\mathfrak{T}_k$) is decidable.

Seese's Conjecture. For any class $\mathcal{C}$ of finite graphs, SAT(MSO, $\mathcal{C}$) is decidable if, and only if, $\mathcal{C}$ has bounded clique width.

Satisfiability

Satisfiability problem:
Let $\mathcal{C}$ be a graph of structures

SAT(FO, $\mathcal{C}$), SAT(MSO, $\mathcal{C}$)	
Input:	MSO- or FO-sentence φ.
Problem:	Does φ have a model in $\mathcal{C}$?.

Trakhtenbrot's theorem: SAT(FO, FIN), i.e. satisfiability in the finite, is undecidable.

For $k \in \mathbb{N}$, let $\mathfrak{T}_k$ be the class of finite graphs of tree-width $\leq k$.

Theorem. (Seese)

For any $k \geq 0$, SAT(MSO, $\mathfrak{T}_k$) is decidable.

Seese's Conjecture. For any class $\mathcal{C}$ of finite graphs, SAT(MSO, $\mathcal{C}$) is decidable if, and only if, $\mathcal{C}$ has bounded clique width.

Satisfiability

Satisfiability problem:
Let $\mathcal{C}$ be a graph of structures

SAT(FO, $\mathcal{C}$), SAT(MSO, $\mathcal{C}$)	
Input:	MSO- or FO-sentence φ.
Problem:	Does φ have a model in $\mathcal{C}$?.

Trakhtenbrot's theorem: SAT(FO, FIN), i.e. satisfiability in the finite, is undecidable.

For $k \in \mathbb{N}$, let $\mathfrak{T}_k$ be the class of finite graphs of tree-width $\leq k$.

Theorem. (Seese)

For any $k \geq 0$, SAT(MSO, $\mathfrak{T}_k$) is decidable.

Seese's Conjecture. For any class $\mathcal{C}$ of finite graphs, SAT(MSO, $\mathcal{C}$) is decidable if, and only if, $\mathcal{C}$ has bounded clique width.

Finitely Representable Structures

Structures as input: If structures are input to algorithms, they must be finite or at least permit a finite presentation.

Finitely representable structures: Much work on finding large classes of (infinite) structures which

- still have a decidable MSO-theory or FO-theory
- admit a finite presentation

Examples:

- Automatic structures
- Infinite binary tree
- Tree-unravellings of structures with decidable MSO-theory
- Structures interpretable in the infinite binary tree

Automata theory: All these results use the strong connection between MSO and finite (tree) automata.