Lecture 3: Algorithmic Meta-Theorems
(A Short Introduction to Parameterized Complexity)

Clemens Grabmayer

Ph.D. Program, Advanced Courses Period
Gran Sasso Science Institute
L'Aquila, Italy

June 22, 2022

Tomorrow

Monday, June 19 10.00 – 12.00	Tuesday, June 20 10.00 – 12.00	Wednesday, June 21 10.00 – 12.00	Thursday, June 22 10.00 – 12.00	Friday, June 23 10.00 – 12.00
Introduction & basic FPT results	**Notions of bounded graph width**	**Guest Lecture Alessandro Aloisio**	**Algorithmic Meta-Theorems**	**FPT-Intractability Classes & Hierarchies**
motivation for FPT kernelization, Crown Lemma, Sunflower Lemma	path-, tree-, clique width, FPT-results by dynamic programming, transferring FPT results betw. widths	examples FPT-results: *firefighting problem*, *coverage in multi-interface networks*,	1st-order logic, monadic 2nd-order logic, FPT-results by Courcelle's Theorems for tree and clique-width	motivation for FP-intractability results, FPT-reductions, class XP (slicewise polynomial), W- and A-Hierarchies, placing problems on these hierarchies
Algorithmic Techniques			*Formal-Method & Algorithmic Techniques*	
				15.00 – 16.00
				Guest Exercise Class Alessandro Aloisio
				Intractability results on the firefighting problem

Overview

- ▶ Courcelle's theorem
  - ▶ FPT-results by model-checking MSO-formulas
    - ▶ for graphs / structures with bounded tree-width
    - ▶ for maximization problems over graphs of bounded tree-width
    - ▶ for graphs of bounded clique-width
  - ▶ applications to concrete problems

Overview

- ▶ logic preliminaries
    - ▶ first-order logic
        - ▶ expressing graph problems by f-o formulas
    - ▶ monadic second-order logic (MSO)
        - ▶ expressing graph problems by MSO formulas
    - ▶ complexity of evaluation and model checking problems
- ▶ Courcelle's theorem
    - ▶ FPT-results by model-checking MSO-formulas
        - ▶ for graphs / structures with bounded tree-width
        - ▶ for maximization problems over graphs of bounded tree-width
        - ▶ for graphs of bounded clique-width
    - ▶ applications to concrete problems

Overview

- ▶ logic preliminaries
    - ▶ first-order logic
        - ▶ expressing graph problems by f-o formulas
    - ▶ monadic second-order logic (MSO)
        - ▶ expressing graph problems by MSO formulas
    - ▶ complexity of evaluation and model checking problems
- ▶ Courcelle's theorem
    - ▶ FPT-results by model-checking MSO-formulas
        - ▶ for graphs / structures with bounded tree-width
        - ▶ for maximization problems over graphs of bounded tree-width
        - ▶ for graphs of bounded clique-width
    - ▶ applications to concrete problems
- ▶ graph minors

Overview

- ▶ logic preliminaries
    - ▶ first-order logic
        - ▶ expressing graph problems by f-o formulas
    - ▶ monadic second-order logic (MSO)
        - ▶ expressing graph problems by MSO formulas
    - ▶ complexity of evaluation and model checking problems
- ▶ Courcelle's theorem
    - ▶ FPT-results by model-checking MSO-formulas
        - ▶ for graphs / structures with bounded tree-width
        - ▶ for maximization problems over graphs of bounded tree-width
        - ▶ for graphs of bounded clique-width
    - ▶ applications to concrete problems
- ▶ graph minors
- ▶ meta-theorems for first-order model-checking: an example

Meta-theorems: idea, benefits and limitations

idea:

- ▶ express a problem P by a logical formula φ_P (of 'short' size)
- ▶ use model checking of φ_P
  on logical structures of bounded width k (tree-, clique-width, . . .)
  - ▶ is time bounded depending on k, size of φ_P, size of the structure
  - ▶ this often facilitates FPT-results

Meta-theorems: idea, benefits and limitations

idea:

- express a problem P by a logical formula φ_P (of 'short' size)
- use model checking of φ_P
  on logical structures of bounded width k (tree-, clique-width, . . .)
    - is time bounded depending on k, size of φ_P, size of the structure
    - this often facilitates FPT-results

benefits:

- a quick and easy way to show
  that [some problems] are fixed-parameter tractable,
- without working out the tedious details
  of a dynamic programming algorithm.

Meta-theorems: idea, benefits and limitations

idea:

- express a problem P by a logical formula φ_P (of 'short' size)
- use model checking of φ_P
  on logical structures of bounded width k (tree-, clique-width, ...)
  - is time bounded depending on k, size of φ_P, size of the structure
  - this often facilitates FPT-results

benefits:

- a quick and easy way to show
  that [some problems] are fixed-parameter tractable,
- without working out the tedious details
  of a dynamic programming algorithm.

limitations:

- algorithms obtained by meta-theorems
  cannot be expected to be optimal.
- a careful analysis of a specific problem at hand
  will usually yield more efficient fpt-algorithms

Logical preliminaries

First-order logic (formula example)

$$\varphi_3 := \exists x_1 \exists x_2 \exists x_3 \big(\neg(x_1 = x_2) \wedge \neg E(x_1, x_2)$$
$$\wedge \neg(x_1 = x_3) \wedge \neg E(x_1, x_3)$$
$$\wedge \neg(x_2 = x_3) \wedge \neg E(x_2, x_3) \big)$$

First-order logic (formula example)

$$\varphi_3 := \exists x_1 \exists x_2 \exists x_3 \big(\neg(x_1 = x_2) \land \neg E(x_1, x_2)$$
$$\land \neg(x_1 = x_3) \land \neg E(x_1, x_3)$$
$$\land \neg(x_2 = x_3) \land \neg E(x_2, x_3) \big)$$

$\mathcal{A}(\mathcal{G}) \vDash \varphi_3 \iff \mathcal{G}$ has a 3-element independent set.

$S \subseteq V$ is independent set in $\mathcal{G} = \langle V, E \rangle : \iff \forall e = \{u, v\} \in E \, (\, \neg(u \in S \land v \in S) \,)$
$$\iff \forall u, v \in S \, (\, u \neq v \Rightarrow \{u, v\} \notin E \,)$$

First-order logic (formula example)

$$\varphi_3 \; := \; \exists x_1 \exists x_2 \exists x_3 \big(\neg(x_1 = x_2) \wedge \neg E(x_1, x_2)$$
$$\wedge \, \neg(x_1 = x_3) \wedge \neg E(x_1, x_3)$$
$$\wedge \, \neg(x_2 = x_3) \wedge \neg E(x_2, x_3) \big)$$

$\mathcal{A}(\mathcal{G}) \vDash \varphi_3 \quad \Longleftrightarrow \quad \mathcal{G}$ has a 3-element independent set.

$$\varphi_k \; := \; \exists x_1 \ldots \exists x_k \big(\bigwedge_{1 \le i < j \le k} \big(\neg(x_i = x_j) \wedge \neg E(x_i, x_j) \big) \big)$$

$S \subseteq V$ is independent set in $\mathcal{G} = \langle V, E \rangle : \Longleftrightarrow \; \forall e = \{u, v\} \in E \, (\, \neg(u \in S \wedge v \in S) \,)$
$$\Longleftrightarrow \; \forall u, v \in S (\, u \ne v \Rightarrow \{u, v\} \notin E \,)$$

First-order logic (formula example)

$$\varphi_3 \;:=\; \exists x_1 \exists x_2 \exists x_3 \big(\neg(x_1 = x_2) \wedge \neg E(x_1, x_2)$$
$$\wedge \neg(x_1 = x_3) \wedge \neg E(x_1, x_3)$$
$$\wedge \neg(x_2 = x_3) \wedge \neg E(x_2, x_3) \big)$$

$$\mathcal{A}(\mathcal{G}) \vDash \varphi_3 \quad \Longleftrightarrow \quad \mathcal{G} \text{ has a } 3\text{-element independent set}.$$

$$\varphi_k \;:=\; \exists x_1 \ldots \exists x_k \Big(\bigwedge_{1 \le i < j \le k} \big(\neg(x_i = x_j) \wedge \neg E(x_i, x_j) \big) \Big)$$

$$\mathcal{A}(\mathcal{G}) \vDash \varphi_k \quad \Longleftrightarrow \quad \mathcal{G} \text{ has a } k\text{-element independent set}.$$

$$S \subseteq V \text{ is independent set in } \mathcal{G} = \langle V, E \rangle : \Longleftrightarrow \quad \forall e = \{u, v\} \in E \, (\, \neg(u \in S \wedge v \in S) \,)$$
$$\Longleftrightarrow \quad \forall u, v \in S (\, u \ne v \Rightarrow \{u, v\} \notin E \,)$$

First-order logic: syntax (language)

- language based on:
  - a *vocabulary* $\tau = \{R_1, \ldots, R_n\}$ of *predicate symbols* R_i together with arity *ar*$(R_i) \in \mathbb{N}$

First-order logic: syntax (language)

- ▶ language based on:
  - ▶ a *vocabulary* $\tau = \{R_1, \ldots, R_n\}$ of *predicate symbols* R_i together with arity $ar(R_i) \in \mathbb{N}$
  - ▶ the binary equality predication $=$

First-order logic: syntax (language)

- ▶ language based on:
  - ▶ a *vocabulary* $\tau = \{R_1, \ldots, R_n\}$ of *predicate symbols* R_i together with arity *ar*$(R_i) \in \mathbb{N}$
  - ▶ the binary equality predication $=$
  - ▶ (first-order) variable symbols: $x, y, z, w, x_1, y_1, z_1, w_1, x_2, \ldots$

First-order logic: syntax (language)

- ▶ language based on:
  - ▶ a *vocabulary* $\tau = \{R_1, \ldots, R_n\}$ of *predicate symbols* R_i together with arity $ar(R_i) \in \mathbb{N}$
  - ▶ the binary equality predication $=$
  - ▶ (first-order) variable symbols: $x, y, z, w, x_1, y_1, z_1, w_1, x_2, \ldots$
  - ▶ propositional connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$

First-order logic: syntax (language)

- ▶ language based on:
  - ▶ a *vocabulary* $\tau = \{R_1, \ldots, R_n\}$ of *predicate symbols* R_i together with arity $ar(R_i) \in \mathbb{N}$
  - ▶ the binary equality predication $=$
  - ▶ (first-order) variable symbols: $x, y, z, w, x_1, y_1, z_1, w_1, x_2, \ldots$
  - ▶ propositional connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$
  - ▶ existential quantifier $\exists$, universal quantifier $\forall$

First-order logic: syntax (language)

- ▶ language based on:
  - ▶ a *vocabulary* $\tau = \{R_1, \ldots, R_n\}$ of *predicate symbols* R_i together with arity $ar(R_i) \in \mathbb{N}$
  - ▶ the binary equality predication $=$
  - ▶ (first-order) variable symbols: $x, y, z, w, x_1, y_1, z_1, w_1, x_2, \ldots$
  - ▶ propositional connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$
  - ▶ existential quantifier $\exists$, universal quantifier $\forall$
- ▶ *atomic formulas (atoms)*: a formula $x = y$ or $R(x_1 \ldots x_n)$ for $R \in \tau$

First-order logic: syntax (language)

- ▶ language based on:
  - ▶ a *vocabulary* $\tau = \{R_1, \ldots, R_n\}$ of *predicate symbols* R_i together with arity $ar(R_i) \in \mathbb{N}$
  - ▶ the binary equality predication $=$
  - ▶ (first-order) variable symbols: $x, y, z, w, x_1, y_1, z_1, w_1, x_2, \ldots$
  - ▶ propositional connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$
  - ▶ existential quantifier $\exists$, universal quantifier $\forall$
- ▶ *atomic formulas (atoms)*: a formula $\boldsymbol{x} = \boldsymbol{y}$ or $R(\boldsymbol{x}_1 \ldots \boldsymbol{x}_n)$ for $R \in \tau$
- ▶ *quantifier-free formula*: atoms, literals (= negated atoms), formulas built up from atoms by using propositional connectives

First-order logic: syntax (language)

- ▶ language based on:
  - ▸ a *vocabulary* $\tau = \{R_1, \ldots, R_n\}$ of *predicate symbols* R_i together with arity $ar(R_i) \in \mathbb{N}$
  - ▸ the binary equality predication $=$
  - ▸ (first-order) variable symbols: $x, y, z, w, x_1, y_1, z_1, w_1, x_2, \ldots$
  - ▸ propositional connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$
  - ▸ existential quantifier $\exists$, universal quantifier $\forall$
- ▶ *atomic formulas (atoms)*: a formula $\boldsymbol{x} = \boldsymbol{y}$ or $R(\boldsymbol{x}_1 \ldots \boldsymbol{x}_n)$ for $R \in \tau$
- ▶ *quantifier-free formula*: atoms, literals (= negated atoms), formulas built up from atoms by using propositional connectives
- ▶ *quantifications* over (first-order variables):
  - ▸ existential quantifications $\exists x$ and universal quantifications $\forall x$

First-order logic: syntax (language)

- ▶ language based on:
  - ▶ a *vocabulary* $\tau = \{R_1, \ldots, R_n\}$ of *predicate symbols* R_i together with arity $ar(R_i) \in \mathbb{N}$
  - ▶ the binary equality predication $=$
  - ▶ (first-order) variable symbols: $x, y, z, w, x_1, y_1, z_1, w_1, x_2, \ldots$
  - ▶ propositional connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$
  - ▶ existential quantifier $\exists$, universal quantifier $\forall$
- ▶ *atomic formulas (atoms)*: a formula $\boldsymbol{x} = \boldsymbol{y}$ or $R(\boldsymbol{x}_1 \ldots \boldsymbol{x}_n)$ for $R \in \tau$
- ▶ *quantifier-free formula*: atoms, literals (= negated atoms), formulas built up from atoms by using propositional connectives
- ▶ *quantifications* over (first-order variables):
  - ▶ existential quantifications $\exists x$ and universal quantifications $\forall x$
- ▶ *formulas*:
$$\varphi ::= \boldsymbol{x} = \boldsymbol{y} \mid R(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{ar(R)}) \qquad (\text{where } R \in \tau)$$
$$\mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \rightarrow \varphi_2 \mid \varphi_1 \leftrightarrow \varphi_2$$
$$\mid \exists x \varphi \mid \forall x \varphi$$

First-order logic: syntax (language)

- ▶ language based on:
    - ▶ a *vocabulary* $\tau = \{R_1, \ldots, R_n\}$ of *predicate symbols* R_i together with arity $ar(R_i) \in \mathbb{N}$
    - ▶ the binary equality predication $=$
    - ▶ (first-order) variable symbols: $x, y, z, w, x_1, y_1, z_1, w_1, x_2, \ldots$
    - ▶ propositional connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$
    - ▶ existential quantifier $\exists$, universal quantifier $\forall$
- ▶ *atomic formulas (atoms)*: a formula $\boldsymbol{x} = \boldsymbol{y}$ or $R(\boldsymbol{x}_1 \ldots \boldsymbol{x}_n)$ for $R \in \tau$
- ▶ *quantifier-free formula*: atoms, literals (= negated atoms), formulas built up from atoms by using propositional connectives
- ▶ *quantifications* over (first-order variables):
    - ▶ existential quantifications $\exists x$ and universal quantifications $\forall x$
- ▶ *formulas*:

$$\varphi ::= \boldsymbol{x} = \boldsymbol{y} \mid R(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{ar(R)}) \qquad (\text{where } R \in \tau)$$
$$\mid \neg \varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \rightarrow \varphi_2 \mid \varphi_1 \leftrightarrow \varphi_2$$
$$\mid \exists x \varphi \mid \forall x \varphi$$

- ▶ *sentences*: formulas without *free* variables.

First-order logic: semantics (structures)

Definition

Let $\tau = \{R_1, \ldots, R_n\}$ be a vocabulary.
A τ-*structure* is a tuple $\mathcal{A} = \langle A; R_1^{\mathcal{A}}, \ldots R_n^{\mathcal{A}} \rangle$ consisting of:

- the *universe* A,
- *interpretations* $R_i^{\mathcal{A}} \subseteq A^{ar(R_i)} = \overbrace{A \times \ldots \times A}^{ar(R_i)}$ for each of the relation symbols R_i in τ, where $i \in \{1, \ldots, n\}$.

Examples

Let $\tau_\mathsf{G} = \{E/_2\}$ vocabulary with binary edge relation.
The *standard structure* for a graph $\mathcal{G} = \langle V, E \rangle$:
$$\mathcal{A}_{\tau_\mathsf{G}}(\mathcal{G}) := \langle V; E^{\mathsf{symm}} \rangle .$$

First-order logic: semantics (structures)

Definition

Let $\tau = \{R_1, \ldots, R_n\}$ be a vocabulary.
A τ-*structure* is a tuple $\mathcal{A} = \langle A; R_1^{\mathcal{A}}, \ldots R_n^{\mathcal{A}}\rangle$ consisting of:

- the *universe* A,
- *interpretations* $R_i{}^{\mathcal{A}} \subseteq A^{ar(R_i)} = \overbrace{A \times \ldots \times A}^{ar(R_i)}$ for each of the relation symbols R_i in τ, where $i \in \{1, \ldots, n\}$.

Examples

Let $\tau_{\mathsf{G}} = \{E/_2\}$ vocabulary with binary edge relation.
The *standard structure* for a graph $\mathcal{G} = \langle V, E\rangle$:
$$\mathcal{A}_{\tau_{\mathsf{G}}}(\mathcal{G}) := \langle V; E^{\mathsf{symm}}\rangle.$$

Example

Let $\tau_{\mathsf{HG}} = \{VERT/_1, EDGE/_1, INC/_2\}$ vocabulary (for hypergraphs).
The *hypergraph structure* for a graph $\mathcal{G} = \langle V, E\rangle$:
$$\mathcal{A}_{\tau_{\mathsf{HG}}}(\mathcal{G}) := \langle V \cup E; V, E, \{\langle v, e\rangle \mid v \in V, e \in E, v \in e\}\rangle.$$

Interpretation of first-order formulas in structures

Let $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$ be a τ-structure. For a τ-formula $\varphi(x_1, \ldots, x_k)$ its *interpretation* $\varphi(\mathcal{A}) \subseteq A^k$ in $\mathcal{A}$ is defined by:

Interpretation of first-order formulas in structures

Let $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$ be a τ-structure. For a τ-formula $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$ its *interpretation* $\varphi(\mathcal{A}) \subseteq A^k$ in $\mathcal{A}$ is defined by:

- If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \equiv R(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_r})$ with $i_1, \ldots, i_r \in [k]$, then:
$$\varphi(\mathcal{A}) := \{\langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{i_1}, \ldots, a_{i_k} \rangle \in R^{\mathcal{A}}\}$$

Interpretation of first-order formulas in structures

Let $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$ be a τ-structure. For a τ-formula $\varphi(x_1, \ldots, x_k)$ its *interpretation* $\varphi(\mathcal{A}) \subseteq A^k$ in $\mathcal{A}$ is defined by:

- If $\varphi(x_1, \ldots, x_k) \equiv R(x_{i_1}, \ldots, x_{i_r})$ with $i_1, \ldots, i_r \in [k]$, then:
$$\varphi(\mathcal{A}) := \left\{ \langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{i_1}, \ldots, a_{i_k} \rangle \in R^{\mathcal{A}} \right\}$$

- If $\varphi(x_1, \ldots, x_k) \equiv \varphi_1(x_{i_1}, \ldots, x_{i_l}) \wedge \varphi_2(x_{j_1}, \ldots, x_{j_m})$ with $i_1, \ldots, i_l, j_1, \ldots, j_m \in [k]$, then:
$$\varphi(\mathcal{A}) := \left\{ \langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{i_1}, \ldots, a_{i_l} \rangle \in \varphi_1(\mathcal{A}) \right\}$$
$$\cap \left\{ \langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{j_1}, \ldots, a_{j_m} \rangle \in \varphi_2(\mathcal{A}) \right\}$$

Interpretation of first-order formulas in structures

Let $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$ be a τ-structure. For a τ-formula $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$ its *interpretation* $\varphi(\mathcal{A}) \subseteq A^k$ in $\mathcal{A}$ is defined by:

▶ If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \equiv R(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_r})$ with $i_1, \ldots, i_r \in [k]$, then:
$$\varphi(\mathcal{A}) := \left\{ \langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{i_1}, \ldots, a_{i_k} \rangle \in R^{\mathcal{A}} \right\}$$

▶ If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \equiv \varphi_1(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_l}) \wedge \varphi_2(\boldsymbol{x}_{j_1}, \ldots, \boldsymbol{x}_{j_m})$ with $i_1, \ldots, i_l, j_1, \ldots, j_m \in [k]$, then:
$$\varphi(\mathcal{A}) := \left\{ \langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{i_1}, \ldots, a_{i_l} \rangle \in \varphi_1(\mathcal{A}) \right\}$$
$$\cap \left\{ \langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{j_1}, \ldots, a_{j_m} \rangle \in \varphi_2(\mathcal{A}) \right\}$$

▶ If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \equiv \exists \boldsymbol{x}_{k+1} \varphi_0(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_\ell})$ with $i_1, \ldots, i_\ell \in [k+1]$, then:
$$\varphi(\mathcal{A}) := \left\{ \langle a_1, \ldots, a_k \rangle \mid \text{there exists } a_{k+1} \in A \right.$$
$$\left. \text{such that } \langle a_{i_1}, \ldots, a_{i_\ell} \rangle \in \varphi_0(\mathcal{A}) \right\}$$

Interpretation of first-order formulas in structures

Let $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$ be a τ-structure. For a τ-formula $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$ its *interpretation* $\varphi(\mathcal{A}) \subseteq A^k$ in $\mathcal{A}$ is defined by:

▶ If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \equiv R(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_r})$ with $i_1, \ldots, i_r \in [k]$, then:
$$\varphi(\mathcal{A}) \coloneqq \{\langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{i_1}, \ldots, a_{i_k} \rangle \in R^{\mathcal{A}}\}$$

▶ If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \equiv \varphi_1(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_l}) \wedge \varphi_2(\boldsymbol{x}_{j_1}, \ldots, \boldsymbol{x}_{j_m})$ with $i_1, \ldots, i_l, j_1, \ldots, j_m \in [k]$, then:
$$\varphi(\mathcal{A}) \coloneqq \{\langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{i_1}, \ldots, a_{i_l} \rangle \in \varphi_1(\mathcal{A})\}$$
$$\cap \{\langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{j_1}, \ldots, a_{j_m} \rangle \in \varphi_2(\mathcal{A})\}$$

▶ If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \equiv \exists \boldsymbol{x}_{k+1} \varphi_0(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_\ell})$ with $i_1, \ldots, i_\ell \in [k+1]$, then:
$$\varphi(\mathcal{A}) \coloneqq \{\langle a_1, \ldots, a_k \rangle \in A^k \mid \text{there exists } a_{k+1} \in A$$
$$\text{such that } \langle a_{i_1}, \ldots, a_{i_\ell} \rangle \in \varphi_0(\mathcal{A})\}$$

▶ $\mathcal{A} \vDash \varphi(a_1, \ldots, a_k)$ will mean: $\langle a_1, \ldots, a_k \rangle \in \varphi(\mathcal{A})$.

Interpretation of first-order formulas in structures

Let $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$ be a τ-structure. For a τ-formula $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$ its *interpretation* $\varphi(\mathcal{A}) \subseteq A^k$ in $\mathcal{A}$ is defined by:

▶ If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \equiv R(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_r})$ with $i_1, \ldots, i_r \in [k]$, then:
$$\varphi(\mathcal{A}) := \left\{ \langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{i_1}, \ldots, a_{i_k} \rangle \in R^{\mathcal{A}} \right\}$$

▶ If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \equiv \varphi_1(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_l}) \wedge \varphi_2(\boldsymbol{x}_{j_1}, \ldots, \boldsymbol{x}_{j_m})$ with $i_1, \ldots, i_l, j_1, \ldots, j_m \in [k]$, then:
$$\varphi(\mathcal{A}) := \left\{ \langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{i_1}, \ldots, a_{i_l} \rangle \in \varphi_1(\mathcal{A}) \right\}$$
$$\cap \left\{ \langle a_1, \ldots, a_k \rangle \in A^k \mid \langle a_{j_1}, \ldots, a_{j_m} \rangle \in \varphi_2(\mathcal{A}) \right\}$$

▶ If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \equiv \exists \boldsymbol{x}_{k+1} \varphi_0(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_\ell})$ with $i_1, \ldots, i_\ell \in [k+1]$, then:
$$\varphi(\mathcal{A}) := \big\{ \langle a_1, \ldots, a_k \rangle \in A^k \big| \text{ there exists } a_{k+1} \in A$$
$$\text{such that } \langle a_{i_1}, \ldots, a_{i_\ell} \rangle \in \varphi_0(\mathcal{A}) \big\}$$

▶ $\mathcal{A} \vDash \varphi(a_1, \ldots, a_k)$ will mean: $\langle a_1, \ldots, a_k \rangle \in \varphi(\mathcal{A})$.

▶ For a sentence φ, $\mathcal{A} \vDash \varphi$ will mean $\varphi(\mathcal{A}) \neq \varnothing$ (then $\varphi(\mathcal{A}) = \{\langle \rangle\}$).

Expressing graph properties by first-order formulas

Exercise

For given formulas $\varphi(x)$ and for all $k \in \mathbb{N}$, $k \geq 1$ define formulas $\exists^{\geq k} x\, \varphi(x)$, $\exists^{<k} x\, \varphi(x)$, $\exists^{=k} x\, \varphi(x)$, such that in a given τ-structure $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$:

$$\mathcal{A} \models \exists^{\geq k} x\, \varphi(x) \iff |\{a \in A \mid \mathcal{A} \models \varphi(a)\}| \geq k$$

$$\mathcal{A} \models \exists^{<k} x\, \varphi(x) \iff |\{a \in A \mid \mathcal{A} \models \varphi(a)\}| < k$$

$$\mathcal{A} \models \exists^{=k} x\, \varphi(x) \iff |\{a \in A \mid \mathcal{A} \models \varphi(a)\}| = k$$

Expressing graph properties by first-order formulas

Exercise

For given formulas $\varphi(x)$ and for all $k \in \mathbb{N}$, $k \geq 1$ define formulas $\exists^{\geq k} x\, \varphi(x)$, $\exists^{<k} x\, \varphi(x)$, $\exists^{=k} x\, \varphi(x)$, such that in a given τ-structure $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau}\rangle$:

$$\mathcal{A} \models \exists^{\geq k} x\, \varphi(x) \quad \Longleftrightarrow \quad |\{a \in A \mid \mathcal{A} \models \varphi(a)\}| \geq k$$

$$\mathcal{A} \models \exists^{<k} x\, \varphi(x) \quad \Longleftrightarrow \quad |\{a \in A \mid \mathcal{A} \models \varphi(a)\}| < k$$

$$\mathcal{A} \models \exists^{=k} x\, \varphi(x) \quad \Longleftrightarrow \quad |\{a \in A \mid \mathcal{A} \models \varphi(a)\}| = k$$

Expressing graph properties by first-order formulas

Exercise

Express by a first-order formula with the vocabulary $\tau_{\mathsf{G}} = \{E/_2\}$ for graphs that:

(i) a graph $\mathcal{G}$ contains a clique with precisely k elements,

(ii) a graph $\mathcal{G}$ has a dominating set with less or equal to k elements,

(iii) a graph $\mathcal{G}$ has a dominating set with precisely k elements,

Expressing graph properties by first-order formulas

Exercise

Express by a first-order formula with the vocabulary $\tau_{\mathsf{G}} = \{E/_2\}$ for graphs that:

(i) a graph $\mathcal{G}$ contains a clique with precisely k elements,

(ii) a graph $\mathcal{G}$ has a dominating set with less or equal to k elements,

(iii) a graph $\mathcal{G}$ has a dominating set with precisely k elements,

Recall:

$$\varphi_k := \exists x_1 \ldots \exists x_k \big(\bigwedge_{1 \le i < j \le k} \big(\neg(x_i = y_i) \wedge \neg E(x_i, x_j) \big) \big)$$

$$\mathcal{A}_{\tau_{\mathsf{G}}}(\mathcal{G}) \vDash \varphi_k \iff \mathcal{G} \text{ has a } k\text{-element independent set}.$$

Expressing graph properties by first-order formulas

Exercise

Express by a first-order formula with the vocabulary $\tau_\mathsf{G} = \{E/_2\}$ for graphs that:

(i) a graph $\mathcal{G}$ contains a clique with precisely k elements,

(ii) a graph $\mathcal{G}$ has a dominating set with less or equal to k elements,

(iii) a graph $\mathcal{G}$ has a dominating set with precisely k elements,

Expressing graph properties by first-order formulas

Exercise

Express by a first-order formula with the vocabulary $\tau_{\mathsf{G}} = \{E/_2\}$ for graphs that:

(i) a graph $\mathcal{G}$ contains a clique with precisely k elements,

(ii) a graph $\mathcal{G}$ has a dominating set with less or equal to k elements,

(iii) a graph $\mathcal{G}$ has a dominating set with precisely k elements,

Expressing graph properties by first-order formulas

> **Exercise**
>
> Express by a first-order formula with the vocabulary with vocabulary
> $\tau_{HG} = \{ VERT/_1, EDGE/_1, INC/_2 \}$ for hypergraphs that:
>
>   (i)   a graph $\mathcal{G}$ contains a clique with precisely k elements,
> (ii)   a graph $\mathcal{G}$ has a dominating set with precisely k elements,
> (iii)   a graph $\mathcal{G}$ has a dominating set with less or equal to k elements.

Expressing graph properties by first-order formulas

Exercise

Express by a first-order formula with the vocabulary with vocabulary
$\tau_{\mathsf{HG}} = \{\, VERT/_1,\ EDGE/_1,\ INC/_2 \,\}$ for hypergraphs that:

  (i)   a graph $\mathcal{G}$ contains a clique with precisely k elements,

 (ii) a graph $\mathcal{G}$ has a dominating set with precisely k elements,

(iii) a graph $\mathcal{G}$ has a dominating set with less or equal to k elements.

Evaluation and model checking (first-order logic)

Let Φ be a class of first-order formulas.
The *evaluation problem* for Φ:

> EVAL(Φ)
>   **Instance:** A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
>   **Problem:** Compute $\varphi(\mathcal{A})$.

Evaluation and model checking (first-order logic)

Let Φ be a class of *first-order* formulas.
The *evaluation problem* for Φ:

EVAL(Φ)
   Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
   Problem: Compute $\varphi(\mathcal{A})$.

The *model checking problem* for Φ:

MC(Φ)
   Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
   Problem: Decide whether $\mathcal{A} \vDash \varphi$ (that is, $\varphi(\mathcal{A}) \neq \varnothing$).

Evaluation and model checking (first-order logic)

Let Φ be a class of first-order formulas.
The *evaluation problem* for Φ:

EVAL(Φ)
    Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
    Problem: Compute $\varphi(\mathcal{A})$.

The *model checking problem* for Φ:

MC(Φ)
    Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
    Problem: Decide whether $\mathcal{A} \models \varphi$ (that is, $\varphi(\mathcal{A}) \neq \varnothing$).

Width of formula φ: maximal number of free variables
                     in a subformula of φ.

Theorem

EVAL(FO) *and* MC(FO) *can be solved in time* $O(|\varphi| \cdot |A|^w \cdot w)$, *where* w *is the width of the input formula* φ.

Monadic second-order logic (formula example)

$$\psi_3 \ := \ \exists C_1 \exists C_2 \exists C_3 \Big(\big(\forall x \big(\bigvee_{i=1}^{3} C_i(x) \big) \big) \wedge \forall x \big(\bigwedge_{1 \le i < j \le 3} \neg \big(C_i(x) \wedge C_j(x) \big) \big)$$
$$\wedge \ \forall x \forall y \big(E(x,y) \to \bigwedge_{i=1}^{3} \neg (C_i(x) \wedge C_i(y)) \big) \Big)$$

$$\mathcal{A}(\mathcal{G}) \models \psi_3 \quad \Longleftrightarrow \quad \mathcal{G} \text{ has is } 3\text{-colorable}.$$

Monadic second-order logic

- ▶ language based on:
  - ▶ a *vocabulary* $\tau = \{R_1, \ldots, R_n\}$ of *predicate symbols* R_i together with arity $ar(R_i) \in \mathbb{N}$
  - ▶ the binary equality predication $=$
  - ▶ first-order variable symbols: $x, y, z, w, x_1, y_1, z_1, w_1, x_2, \ldots$
  - ▶ monadic second-order variable symbols (symbolizing variables for unary predicate symbols): $X, Y, Z, W, X_1, Y_1, Z_1, W_1, X_1, \ldots,$
  - ▶ propositional connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$
  - ▶ existential quantifier $\exists$, universal quantifier $\forall$

Interpretation of MSO-formulas in first-order structures

Let $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$ be a τ-structure.

For a MSO(τ)-formula $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell)$ its *interpretation* $\varphi(\mathcal{A}) \subseteq A^k \times \mathcal{P}(A)^\ell$ in $\mathcal{A}$ is defined by:

Interpretation of MSO-formulas in first-order structures

Let $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$ be a τ-structure.

For a MSO(τ)-formula $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell)$ its *interpretation* $\varphi(\mathcal{A}) \subseteq A^k \times \mathcal{P}(A)^\ell$ in $\mathcal{A}$ is defined by:

- similar clauses as before, plus:

Interpretation of MSO-formulas in first-order structures

Let $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$ be a τ-structure.

For a $\mathrm{MSO}(\tau)$-formula $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell)$ its *interpretation* $\varphi(\mathcal{A}) \subseteq A^k \times \mathcal{P}(A)^\ell$ in $\mathcal{A}$ is defined by:

- similar clauses as before, plus:

- If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell) \equiv \boldsymbol{X}_i(\boldsymbol{x}_j)$ with $i \in [k]$ and $j \in [\ell]$, then:
$$\varphi(\mathcal{A}) := \{\langle a_1, \ldots, a_k, P_1, \ldots, P_\ell \rangle \in A^k \times \mathcal{P}(A)^\ell \mid a_j \in P_i \}$$

Interpretation of MSO-formulas in first-order structures

Let $\mathcal{A} = \langle A; \{ R^{\mathcal{A}} \}_{R \in \tau} \rangle$ be a τ-structure.
For a $\mathsf{MSO}(\tau)$-formula $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell)$ its *interpretation*
$\varphi(\mathcal{A}) \subseteq A^k \times \mathcal{P}(A)^\ell$ in $\mathcal{A}$ is defined by:

- similar clauses as before, plus:

- If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell) \equiv \boldsymbol{X}_i(\boldsymbol{x}_j)$ with $i \in [k]$ and $j \in [\ell]$, then:
$$\varphi(\mathcal{A}) := \big\{ \langle a_1, \ldots, a_k, P_1, \ldots, P_\ell \rangle \in A^k \times \mathcal{P}(A)^\ell \mid a_j \in P_i \big\}$$

- If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell) \equiv \exists \boldsymbol{X}_{k+1} \varphi_0(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_{k'}}, \boldsymbol{X}_{j_1}, \ldots, \boldsymbol{X}_{j_{\ell'}})$
with $i_1, \ldots, i_{k'} \in [k]$, and $j_1, \ldots, j_{\ell'} \in [\ell+1]$ then:
$$\varphi(\mathcal{A}) := \big\{ \langle a_1, \ldots, a_k, P_1, \ldots, P_\ell \rangle \in A^k \times \mathcal{P}(A)^\ell \big|$$
$$\text{there exists } P_{\ell+1} \in \mathcal{P}(A) \text{ such that}$$
$$\langle a_{i_1}, \ldots, a_{i_{k'}}, P_{j_1}, \ldots, P_{j_{\ell'}} \rangle \in \varphi_0(\mathcal{A}) \big\}$$

Interpretation of MSO-formulas in first-order structures

Let $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$ be a τ-structure.
For a MSO(τ)-formula $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell)$ its *interpretation*
$\varphi(\mathcal{A}) \subseteq A^k \times \mathcal{P}(A)^\ell$ in $\mathcal{A}$ is defined by:

- similar clauses as before, plus:

- If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell) \equiv \boldsymbol{X}_i(\boldsymbol{x}_j)$ with $i \in [k]$ and $j \in [\ell]$, then:
$$\varphi(\mathcal{A}) \coloneqq \left\{ \langle a_1, \ldots, a_k, P_1, \ldots, P_\ell \rangle \in A^k \times \mathcal{P}(A)^\ell \mid a_j \in P_i \right\}$$

- If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell) \equiv \exists \boldsymbol{X}_{k+1} \varphi_0(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_{k'}}, \boldsymbol{X}_{j_1}, \ldots, \boldsymbol{X}_{j_{\ell'}})$
with $i_1, \ldots, i_{k'} \in [k]$, and $j_1, \ldots, j_{\ell'} \in [\ell+1]$ then:
$$\varphi(\mathcal{A}) \coloneqq \big\{ \langle a_1, \ldots, a_k, P_1, \ldots, P_\ell \rangle \in A^k \times \mathcal{P}(A)^\ell \big|$$
$$\text{there exists } P_{\ell+1} \in \mathcal{P}(A) \text{ such that}$$
$$\langle a_{i_1}, \ldots, a_{i_{k'}}, P_{j_1}, \ldots, P_{j_{\ell'}} \rangle \in \varphi_0(\mathcal{A}) \big\}$$

  - $\mathcal{A} \vDash \varphi(a_1, \ldots, a_k, P_1, \ldots, P_\ell)$
will mean: $\langle a_1, \ldots, a_k, P_1, \ldots, P_\ell \rangle \in \varphi(\mathcal{A})$.

Interpretation of MSO-formulas in first-order structures

Let $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$ be a τ-structure.
For a MSO(τ)-formula $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell)$ its *interpretation*
$\varphi(\mathcal{A}) \subseteq A^k \times \mathcal{P}(A)^\ell$ in $\mathcal{A}$ is defined by:

- similar clauses as before, plus:

- If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell) \equiv \boldsymbol{X}_i(\boldsymbol{x}_j)$ with $i \in [k]$ and $j \in [\ell]$, then:
$$\varphi(\mathcal{A}) := \big\{ \langle a_1, \ldots, a_k, P_1, \ldots, P_\ell \rangle \in A^k \times \mathcal{P}(A)^\ell \mid a_j \in P_i \big\}$$

- If $\varphi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell) \equiv \exists \boldsymbol{X}_{k+1} \varphi_0(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_{k'}}, \boldsymbol{X}_{j_1}, \ldots, \boldsymbol{X}_{j_{\ell'}})$
with $i_1, \ldots, i_{k'} \in [k]$, and $j_1, \ldots, j_{\ell'} \in [\ell + 1]$ then:
$$\varphi(\mathcal{A}) := \big\{ \langle a_1, \ldots, a_k, P_1, \ldots, P_\ell \rangle \in A^k \times \mathcal{P}(A)^\ell \mid$$
there exists $P_{\ell+1} \in \mathcal{P}(A)$ such that
$$\langle a_{i_1}, \ldots, a_{i_{k'}}, P_{j_1}, \ldots, P_{j_{\ell'}} \rangle \in \varphi_0(\mathcal{A}) \big\}$$

  - $\mathcal{A} \models \varphi(a_1, \ldots, a_k, P_1, \ldots, P_\ell)$
will mean: $\langle a_1, \ldots, a_k, P_1, \ldots, P_\ell \rangle \in \varphi(\mathcal{A})$.

  - For a sentence φ, $\mathcal{A} \models \varphi$ will mean $\varphi(\mathcal{A}) \neq \varnothing$ (then $\varphi(\mathcal{A}) = \{\langle\rangle\}$).

Monadic second-order logic (formula example)

$$\psi_3 \;:=\; \exists C_1 \exists C_2 \exists C_3 \Big(\big(\forall x \big(\bigvee_{i=1}^{3} C_i(x) \big) \big) \wedge \forall x \big(\bigwedge_{1 \leq i < j \leq 3} \neg \big(C_i(x) \wedge C_j(x) \big) \big)$$

$$\wedge \; \forall x \forall y \big(E(x,y) \rightarrow \bigwedge_{i=1}^{3} \neg (C_i(x) \wedge C_i(y)) \big) \Big)$$

$$\mathcal{A}(\mathcal{G}) \vDash \psi_3 \quad \Longleftrightarrow \quad \mathcal{G} \text{ has is } 3\text{-colorable} \,.$$

Monadic second-order logic (formula example)

$$\psi_3 := \exists C_1 \exists C_2 \exists C_3 \Big(\big(\forall x (\bigvee_{i=1}^{3} C_i(x)) \big) \wedge \forall x \big(\bigwedge_{1 \le i < j \le 3} \neg \big(C_i(x) \wedge C_j(x) \big) \big)$$
$$\wedge \, \forall x \forall y \big(E(x,y) \to \bigwedge_{i=1}^{3} \neg (C_i(x) \wedge C_i(y)) \big) \Big)$$

$$\equiv \, \exists C_1 \exists C_2 \exists C_3 \Big(\forall x (C_1(x) \vee C_2(x) \vee C_3(x))$$
$$\wedge \, \forall x \big(\neg (C_1(x) \wedge C_2(x)) \wedge \neg (C_1(x) \wedge C_3(x))$$
$$\wedge \, \neg (C_2(x) \wedge C_3(x)) \big)$$
$$\wedge \, \forall x \forall y \big(E(x,y) \to \neg (C_1(x) \wedge C_1(y))$$
$$\wedge \, \neg (C_2(x) \wedge C_2(y))$$
$$\wedge \, \neg (C_3(x) \wedge C_3(y)) \big) \Big)$$

$$\mathcal{A}(\mathcal{G}) \vDash \psi_3 \iff \mathcal{G} \text{ has is } 3\text{-colorable} .$$

Expressing graph properties by MSO formulas (1)

Exercise

Express by a monadic second-order formula $\varphi(X)$ with one free unary predicate variable X over the vocabulary $\tau_{\mathsf{G}} = \{E/_2\}$ for graphs that for all graphs $\mathcal{G} = \langle V, E \rangle$:

$$\mathcal{A}_{\tau_{\mathsf{G}}}(\mathcal{G}) \vDash \varphi(S) \iff S \subseteq V \text{ is an independent set in } \mathcal{G}$$

Recall:

$$S \subseteq V \text{ is independent set in } \mathcal{G} :\iff \forall e = \{u,v\} \in E \, (\, \neg(u \in S \wedge v \in S) \,)$$
$$\iff \forall u,v \in S (\, u \neq v \Rightarrow \{u,v\} \notin E \,)$$

Exercise

Express the independent set property by a $\mathsf{MSO}(\tau_{\mathsf{HG}})$ formula ψ with vocabulary $\tau_{\mathsf{HG}} = \{ VERT/_1, EDGE/_1, INC/_2 \}$ for hypergraphs:

$$\mathcal{A}_{\tau_{\mathsf{HG}}}(\mathcal{G}) \vDash \psi(S) \iff S \subseteq V \text{ is an independent set in } \mathcal{G}$$

Expressing graph properties by MSO formulas (1)

Exercise

Express by a monadic second-order formula $\varphi(X)$ with one free unary predicate variable X over the vocabulary $\tau_\mathsf{G} = \{E/_2\}$ for graphs that for all graphs $\mathcal{G} = \langle V, E \rangle$:

$$\mathcal{A}_{\tau_\mathsf{G}}(\mathcal{G}) \vDash \varphi(S) \iff S \subseteq V \text{ is an independent set in } \mathcal{G}$$

Exercise

Express the independent set property by a $\mathsf{MSO}(\tau_\mathsf{HG})$ formula ψ with vocabulary $\tau_\mathsf{HG} = \{\textit{VERT}/_1, \textit{EDGE}/_1, \textit{INC}/_2\}$ for hypergraphs:

$$\mathcal{A}_{\tau_\mathsf{HG}}(\mathcal{G}) \vDash \psi(S) \iff S \subseteq V \text{ is an independent set in } \mathcal{G}$$

Expressing graph properties by MSO formulas (2)

Exercise

Express by a monadic second-order formula *feedback*(X) with one free unary predicate variable X over $\tau_{\mathsf{HG}} = \{$ *VERT*$/_1$, *EDGE*$/_1$, *INC*$/_2\}$, the vocabulary for graphs, that for all hypergraphs $\mathcal{G} = \langle V, E \rangle$:

$$\mathcal{A}_{\tau_{\mathsf{HG}}}(\mathcal{G}) \vDash \textit{feedback}(S) \iff S \subseteq V \text{ is a feedback vertex set}$$

(A set $S \subseteq V$ is a feedback vertex set in $\mathcal{G}$ if S contains a vertex of every cycle of $\mathcal{G}$.)

Steps:

- ▶ Construct a formula *cycle-family*(X) that expresses the property of a set being the union of cycles.
- ▶ Using *cycle-family*(X), construct *feedback*(X).

MSO for graphs and hypergraphs

- MSO(τ_G) : MSO with vocabulary $\tau_G = \{E/_2\}$

MSO for graphs and hypergraphs

- MSO(τ_G) : MSO with vocabulary $\tau_G = \{ E/_2 \}$
- MSO(τ_{HG}) : MSO with vocab. $\tau_{HG} = \{ VERT/_1, EDGE/_1, INC/_2 \}$

MSO for graphs and hypergraphs

- $MSO(\tau_G)$: MSO with vocabulary $\tau_G = \{E/_2\}$
- $MSO(\tau_{HG})$: MSO with vocab. $\tau_{HG} = \{VERT/_1, EDGE/_1, INC/_2\}$
- MSO_1:
  - vocabulary: $\{INC/_2\}$
  - quantifications: $\exists_{(vert)}x \ / \ \forall_{(vert)}x \ , \ \exists_{(edge)}x \ / \ \forall_{(edge)}x \ ,$
    $\exists_{(vert)}X \ / \ \forall_{(vert)}X$

MSO for graphs and hypergraphs

- $\mathrm{MSO}(\tau_{\mathrm{G}})$: MSO with vocabulary $\tau_{\mathrm{G}} = \{ E/_2 \}$
- $\mathrm{MSO}(\tau_{\mathrm{HG}})$: MSO with vocab. $\tau_{\mathrm{HG}} = \{ VERT/_1, EDGE/_1, INC/_2 \}$
- MSO_1 :
  - vocabulary: $\{ INC/_2 \}$
  - quantifications: $\exists_{(\mathrm{vert})} x \;/\; \forall_{(\mathrm{vert})} x \;,\;\; \exists_{(\mathrm{edge})} x \;/\; \forall_{(\mathrm{edge})} x \;,$
    $\exists_{(\mathrm{vert})} X \;/\; \forall_{(\mathrm{vert})} X$
- MSO_2 :
  - vocabulary: $\{ INC/_2 \}$
  - quantifications: $\exists_{(\mathrm{vert})} x \;/\; \forall_{(\mathrm{vert})} x \;,\;\; \exists_{(\mathrm{edge})} x \;/\; \forall_{(\mathrm{edge})} x \;,$
    $\exists_{(\mathrm{vert})} X \;/\; \forall_{(\mathrm{vert})} X, \; \exists_{(\mathrm{edge})} X \;/\; \forall_{(\mathrm{edge})} X$

MSO for graphs and hypergraphs

- $MSO(\tau_G)$: MSO with vocabulary $\tau_G = \{E/_2\}$
- $MSO(\tau_{HG})$: MSO with vocab. $\tau_{HG} = \{VERT/_1, EDGE/_1, INC/_2\}$
- MSO_1 :
  - vocabulary: $\{INC/_2\}$
  - quantifications: $\exists_{(vert)}x$ / $\forall_{(vert)}x$, $\exists_{(edge)}x$ / $\forall_{(edge)}x$,
    $\exists_{(vert)}X$ / $\forall_{(vert)}X$
- MSO_2 :
  - vocabulary: $\{INC/_2\}$
  - quantifications: $\exists_{(vert)}x$ / $\forall_{(vert)}x$, $\exists_{(edge)}x$ / $\forall_{(edge)}x$,
    $\exists_{(vert)}X$ / $\forall_{(vert)}X$, $\exists_{(edge)}X$ / $\forall_{(edge)}X$

Correspondences

$$MSO(\tau_G) \quad \text{corresponds to} \quad MSO_1$$

where 'corresponds to' means: 'is equally expressive as'.

MSO for graphs and hypergraphs

- $\mathrm{MSO}(\tau_{\mathrm{G}})$: MSO with vocabulary $\tau_{\mathrm{G}} = \{E/_2\}$
- $\mathrm{MSO}(\tau_{\mathrm{HG}})$: MSO with vocab. $\tau_{\mathrm{HG}} = \{VERT/_1, EDGE/_1, INC/_2\}$
- MSO_1 :
  - vocabulary: $\{INC/_2\}$
  - quantifications: $\exists_{(\mathrm{vert})}x$ / $\forall_{(\mathrm{vert})}x$, $\exists_{(\mathrm{edge})}x$ / $\forall_{(\mathrm{edge})}x$, $\exists_{(\mathrm{vert})}X$ / $\forall_{(\mathrm{vert})}X$
- MSO_2 :
  - vocabulary: $\{INC/_2\}$
  - quantifications: $\exists_{(\mathrm{vert})}x$ / $\forall_{(\mathrm{vert})}x$, $\exists_{(\mathrm{edge})}x$ / $\forall_{(\mathrm{edge})}x$, $\exists_{(\mathrm{vert})}X$ / $\forall_{(\mathrm{vert})}X$, $\exists_{(\mathrm{edge})}X$ / $\forall_{(\mathrm{edge})}X$

Correspondences

$\mathrm{MSO}(\tau_{\mathrm{G}})$	corresponds to	MSO_1
$\mathrm{MSO}(\tau_{\mathrm{HG}})$	corresponds to	MSO_2

where 'corresponds to' means: 'is equally expressive as'.

MSO for graphs and hypergraphs

- $\text{MSO}(\tau_G)$: MSO with vocabulary $\tau_G = \{E/_2\}$
- $\text{MSO}(\tau_{HG})$: MSO with vocab. $\tau_{HG} = \{VERT/_1, EDGE/_1, INC/_2\}$
- MSO_1:
  - vocabulary: $\{INC/_2\}$
  - quantifications: $\exists_{(\text{vert})} x$ / $\forall_{(\text{vert})} x$, $\exists_{(\text{edge})} x$ / $\forall_{(\text{edge})} x$,
    $\exists_{(\text{vert})} X$ / $\forall_{(\text{vert})} X$
- MSO_2:
  - vocabulary: $\{INC/_2\}$
  - quantifications: $\exists_{(\text{vert})} x$ / $\forall_{(\text{vert})} x$, $\exists_{(\text{edge})} x$ / $\forall_{(\text{edge})} x$,
    $\exists_{(\text{vert})} X$ / $\forall_{(\text{vert})} X$, $\exists_{(\text{edge})} X$ / $\forall_{(\text{edge})} X$

Correspondences

$\text{MSO}(\tau_G)$	corresponds to	MSO_1
$\text{MSO}(\tau_{HG})$	corresponds to	MSO_2

where 'corresponds to' means: 'is equally expressive as'.

Note:

$\text{MSO}(\tau_{HG})$ / MSO_2 are more expressive than $\text{MSO}(\tau_G)$ / MSO_1.

MSO for graphs and hypergraphs

- $\text{MSO}(\tau_G)$: MSO with vocabulary $\tau_G = \{E/_2\}$
- $\text{MSO}(\tau_{HG})$: MSO with vocab. $\tau_{HG} = \{VERT/_1, EDGE/_1, INC/_2\}$
- MSO_1:
  - vocabulary: $\{INC/_2\}$
  - quantifications: $\exists_{(vert)}x$ / $\forall_{(vert)}x$, $\exists_{(edge)}x$ / $\forall_{(edge)}x$,
    $\exists_{(vert)}X$ / $\forall_{(vert)}X$
- MSO_2:
  - vocabulary: $\{INC/_2\}$
  - quantifications: $\exists_{(vert)}x$ / $\forall_{(vert)}x$, $\exists_{(edge)}x$ / $\forall_{(edge)}x$,
    $\exists_{(vert)}X$ / $\forall_{(vert)}X$, $\exists_{(edge)}X$ / $\forall_{(edge)}X$

Correspondences

$\text{MSO}(\tau_G)$	corresponds to	MSO_1
$\text{MSO}(\tau_{HG})$	corresponds to	MSO_2

where 'corresponds to' means: 'is equally expressive as'.

Note:

We use MSO for either logic, restrict to $\text{MSO}(\tau_G)$ / MSO_1 if needed.

MSO for graphs and hypergraphs

- $MSO(\tau_G)$: MSO with vocabulary $\tau_G = \{ E/_2 \}$
- $MSO(\tau_{HG})$: MSO with vocab. $\tau_{HG} = \{ VERT/_1, EDGE/_1, INC/_2 \}$
- MSO_1:
  - vocabulary: $\{ INC/_2 \}$
  - quantifications: $\exists_{(vert)} x \,/\, \forall_{(vert)} x\,,\; \exists_{(edge)} x \,/\, \forall_{(edge)} x\,,$
    $\exists_{(vert)} X \,/\, \forall_{(vert)} X$
- MSO_2:
  - vocabulary: $\{ INC/_2 \}$
  - quantifications: $\exists_{(vert)} x \,/\, \forall_{(vert)} x\,,\; \exists_{(edge)} x \,/\, \forall_{(edge)} x\,,$
    $\exists_{(vert)} X \,/\, \forall_{(vert)} X\,,\; \exists_{(edge)} X \,/\, \forall_{(edge)} X$

Correspondences

$MSO(\tau_G)$	corresponds to	MSO_1
$MSO(\tau_{HG})$	corresponds to	MSO_2

where 'corresponds to' means: 'is equally expressive as'.

Note:

$MSO(\tau_{HG})$ / MSO_2 are more expressive than $MSO(\tau_G)$ / MSO_1.

Expressing graph properties by MSO formulas (5)

Exercise

Express by a MSO(τ_{HG}) formula *conn*(X) with one free unary predicate variable X over $\tau_{\text{HG}} = \{$ *VERT*/₁, *EDGE*/₁, *INC*/₂ $\}$, the vocabulary for graphs, that for all hypergraphs $\mathcal{G} = \langle V, E \rangle$:

$\quad \mathcal{A}_{\tau_{\text{HG}}}(\mathcal{G}) \vDash$ *hamiltonian* $\iff$ there is a Hamiltonian path in $\mathcal{G}$.

Expressing graph properties by MSO formulas (5)

Exercise

Express by a MSO(τ_{HG}) formula *conn*(X) with one free unary predicate variable X over $\tau_{\text{HG}} = \{\, VERT/_1, EDGE/_1, INC/_2 \,\}$, the vocabulary for graphs, that for all hypergraphs $\mathcal{G} = \langle V, E \rangle$:

$$\mathcal{A}_{\tau_{\text{HG}}}(\mathcal{G}) \vDash hamiltonian \iff \text{there is a Hamiltonian path in } \mathcal{G}.$$

Note:

▶ This property is not expressible by a (single) MSO(τ_{G}) formula.

Expressing graph properties by MSO formulas (5)

Exercise

Express by a MSO(τ_{HG}) formula *conn*(X) with one free unary predicate variable X over $\tau_{HG} = \{$ *VERT*/₁, *EDGE*/₁, *INC*/₂ $\}$, the vocabulary for graphs, that for all hypergraphs $\mathcal{G} = \langle V, E \rangle$:

$$\mathcal{A}_{\tau_{HG}}(\mathcal{G}) \vDash \textit{hamiltonian} \iff \text{ there is a Hamiltonian path in } \mathcal{G}.$$

Note:

► This property is not expressible by a (single) MSO(τ_G) formula.

► Other property that are not MSO(τ_G) expressible:
  ► balanced bipartite graphs
  ► existence of a perfect matching
  ► simple graphs
  ► existence of spanning trees with maximum degree 3

Expressing graph properties by MSO formulas (5)

Exercise

$\mathcal{A}_{\tau_{\mathsf{HG}}}(\mathcal{G}) \vDash$ *hamiltonian* $\iff$ there is a Hamiltonian path in $\mathcal{G}$.

Expressing graph properties by MSO formulas (5)

Exercise

$\mathcal{A}_{\tau_{\mathsf{HG}}}(\mathcal{G}) \vDash$ *hamiltonian* $\iff$ there is a Hamiltonian path in $\mathcal{G}$.

Evaluation and model checking (MSO)

The *model checking problem* for MSO-formulas on labeled, ordered unranked trees:

MC(MSO, TREE$_{lo}$)

   Instance: A labeled, ordered, unranked Σ-tree $\mathcal{T}$,
                   and a MSO(τ_Σ^u)-formula φ

   Problem: Decide whether $\mathcal{T} \vDash \varphi$.

---

where for given alphabet Σ, $\tau_\Sigma^u := \{E/_2, N/_2\} \cup \{P_a/_1 \mid a \in \Sigma\}$.

Evaluation and model checking (MSO)

The *model checking problem* for MSO-formulas on labeled, ordered
unranked trees:

MC(MSO, TREE$_{lo}$)
    Instance: A labeled, ordered, unranked Σ-tree $\mathcal{T}$,
                 and a MSO(τ_Σ^u)-formula φ
    Problem: Decide whether $\mathcal{T} \vDash \varphi$.

---

where for given alphabet Σ, $\tau_\Sigma^u := \{E/_2, N/_2\} \cup \{P_a/_1 \mid a \in \Sigma\}$.

Theorem

MC(MSO, TREE$_{lo}$) $\in$ FPT.
More precisely, there is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that
MC(MSO, TREE$_{lo}$) *can be decided in time* $\leq O(f(|\varphi|) + \|\mathcal{T}\|)$.

Evaluation and model checking (MSO)

The *model checking problem* for MSO-formulas on labeled, ordered unranked trees:

$\mathrm{MC}(\mathrm{MSO}, \mathrm{TREE}_{lo})$

    Instance: A labeled, ordered, unranked Σ-tree $\mathcal{T}$,
                 and a $\mathrm{MSO}(\tau_\Sigma^u)$-formula φ

    Problem: Decide whether $\mathcal{T} \vDash \varphi$.

---

where for given alphabet Σ, $\tau_\Sigma^u := \{E/_2, N/_2\} \cup \{P_a/_1 \mid a \in \Sigma\}$.

Theorem

$\mathrm{MC}(\mathrm{MSO}, \mathrm{TREE}_{lo}) \in \mathrm{FPT}$.
More precisely, there is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that
$\mathrm{MC}(\mathrm{MSO}, \mathrm{TREE}_{lo})$ *can be decided in time* $\leq O(f(|\varphi|) + \|\mathcal{T}\|)$.

Note that here: $f(k) \geq 2^{\cdot^{\cdot^{\cdot^{2}}}} \Big\} k$ (a non-elementary function).

Courcelle's Theorem

Courcelle's Theorem for graphs

---

p^*-*tw*-MC(MSO)

    Instance: A graph $\mathcal{G}$ and an MSO(τ_{HG})-sentence φ.
    Parameter: $tw(\mathcal{G}) + |\varphi|$ (where $tw(\mathcal{G})$ the tree-width of $\mathcal{G}$)
    Problem: Decide whether $\mathcal{A}(\mathcal{G}) \vDash \varphi$.

Courcelle's Theorem for graphs

p^*-tw-MC(MSO)
   Instance: A graph $\mathcal{G}$ and an MSO(τ_{HG})-sentence φ.
   Parameter: $tw(\mathcal{G}) + |\varphi|$ (where $tw(\mathcal{G})$ the tree-width of $\mathcal{G}$)
   Problem: Decide whether $\mathcal{A}(\mathcal{G}) \vDash \varphi$.

Theorem (special case of Courcelle's Theorem)

p^*-tw-MC(MSO) $\in$ FPT. *More precisely, the problem is decidable, for some computable and non-decreasing function $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ by an algorithm in time:*

$$f(k_1, k_2) \cdot n, \quad \text{where } k_1 := tw(\mathcal{A}),\, k_2 := |\varphi|,\, n := |V(\mathcal{G})|$$

Courcelle's Theorem: applications (1)

> p^*-*tw*-COLORABILITY $\in$ FPT
> **Instance:** A graph $\mathcal{G}$ and $\ell \in \mathbb{N}$.
> **Parameter:** $tw(\mathcal{C})$
> **Problem:** Decide whether is $\mathcal{G}$ ℓ-colorable.

Example

▶ p^*-*tw*-3-COLORABILITY $\in$ FPT.

Courcelle's Theorem: applications (1)

p^*-*tw*-COLORABILITY $\in$ FPT
 Instance: A graph $\mathcal{G}$ and $\ell \in \mathbb{N}$.
 Parameter: $tw(\mathcal{C})$
 Problem: Decide whether is $\mathcal{G}$ ℓ-colorable.

Example

▶ p^*-*tw*-3-COLORABILITY $\in$ FPT.
▶ p^*-*tw*-COLORABILITY $\in$ FPT.

Courcelle's Theorem: applications (1)

> p^*-*tw*-COLORABILITY $\in$ FPT
>   **Instance:** A graph $\mathcal{G}$ and $\ell \in \mathbb{N}$.
>   **Parameter:** *tw*($\mathcal{C}$)
>   **Problem:** Decide whether is $\mathcal{G}$ ℓ-colorable.

Example

- p^*-*tw*-3-COLORABILITY $\in$ FPT.
- p^*-*tw*-COLORABILITY $\in$ FPT.

Courcelle's Theorem: applications (2)

p^*-*tw*-HAMILTONICITY
  Instance: A graph $\mathcal{G}$
  Parameter: $tw(\mathcal{C})$
  Problem: Decide whether $\mathcal{G}$ is a hamiltonian (that is, contains
              a cyclic path that visits every vertex precisely once).

Example

p^*-*tw*-HAMILTONICITY $\in$ FPT.

Courcelle's Theorem: applications (2)

---

p^*-*tw*-HAMILTONICITY

Instance: A graph $\mathcal{G}$
Parameter: $tw(\mathcal{C})$
Problem: Decide whether $\mathcal{G}$ is a hamiltonian (that is, contains a cyclic path that visits every vertex precisely once).

Example

p^*-*tw*-HAMILTONICITY $\in$ FPT.

Tree decompositions, and tree-width for graphs

Definition (recalling tree-width for graphs)

A *tree decomposition* of a graph $\mathcal{G} = \langle V, E \rangle$
is a pair $\langle \mathcal{T}, \{B_t\}_{t \in T} \rangle$ where $\mathcal{T} = \langle T, F \rangle$ a (undirected, unrooted) tree, and $B_t \subseteq V$ for all $t \in T$ such that:

(T1) $A = \bigcup_{t \in T} B_t$ (every vertex of $\mathcal{G}$ is in some bag).

(T2) $(\forall a \in A)\big[\{t \in T \mid a \in B_t\}$ is connected subtree of $\mathcal{T} \big]$
        (the vertices whose bags contain some vertex of $\mathcal{G}$
          form a connected subtree of $\mathcal{T}$).

(T3) $(\forall \{u, v\} \in E)(\exists t \in T)\big[\{u, v\} \subseteq B_t \big]$
        (the vertices of every edge of $\mathcal{G}$ are realized in some bag).

The *width* of a tree dec. $\langle \mathcal{T}, \{B_t\}_{t \in T} \rangle$ is $\max\{|B_t| - 1 \mid t \in T\}$.

The *tree-width* $tw(\mathcal{A})$ of a τ-structure $\mathcal{A}$ is defined by:

        $tw(\mathcal{A}) :=$ minimal width of a tree decomposition of $\mathcal{A}$.

Tree decompositions, and tree-width for structures

Definition (extension of tree-width to structures)

A *tree decomposition* of a τ-structure $\mathcal{A} = \langle A; \{R^{\mathcal{A}}\}_{R \in \tau} \rangle$
is a pair $\langle \mathcal{T}, \{B_t\}_{t \in T} \rangle$ where $\mathcal{T} = \langle T, F \rangle$ a (undirected, unrooted) tree,
and $B_t \subseteq V$ for all $t \in T$ such that:

(T1) $A = \bigcup_{t \in T} B_t$ (every element of the universe of $\mathcal{A}$ is in some bag).

(T2) $(\forall a \in A) \big[\{t \in T \mid a \in B_t\}$ is connected subtree of $\mathcal{T} \big]$
      (the universe elements whose bags contain some vertex of $\mathcal{G}$
       form a connected subtree of $\mathcal{T}$).

(T3) $(\forall R \in \tau)\,(\forall \langle a_1, \ldots, a_r \rangle \in R^{\mathcal{A}})(\exists t \in T) \big[\{a_1, \ldots, a_r\} \subseteq B_t \big]$
      (the vertices of every 'hyperedge' in $R^{\mathcal{A}}$ are realized in some bag).

The *width* of a tree dec. $\langle \mathcal{T}, \{B_t\}_{t \in T} \rangle$ is $\max \{|B_t| - 1 \mid t \in T\}$.

The *tree-width tw($\mathcal{A}$)* of a τ-structure $\mathcal{A}$ is defined by:

$tw(\mathcal{A}) :=$ minimal width of a tree decomposition of $\mathcal{A}$.

Courcelle's Theorem

p^*-tw-MC(MSO)

Instance: A structure $\mathcal{A}$ and an MSO-sentence φ.
Parameter: $tw(\mathcal{A}) + |\varphi|$.
Problem: Decide whether $\mathcal{A} \vDash \varphi$.

Courcelle's Theorem

p^*-tw-MC(MSO)

Instance: A structure $\mathcal{A}$ and an MSO-sentence φ.

Parameter: $tw(\mathcal{A}) + |\varphi|$.

Problem: Decide whether $\mathcal{A} \vDash \varphi$.

Theorem ([Courcelle, 1990])

p^*-tw-MC(MSO) $\in$ FPT.

Courcelle's Theorem

p^*-*tw*-MC(MSO)

Instance: A structure $\mathcal{A}$ and an MSO-sentence φ.
Parameter: *tw*$(\mathcal{A}) + |\varphi|$.
Problem: Decide whether $\mathcal{A} \vDash \varphi$.

Theorem ([Courcelle, 1990])

p^*-*tw*-MC(MSO) $\in$ FPT. *More precisely, the problem is decidable by an algorithm in time:*

$$f(k_1, k_2) \cdot |A| + O(\|\mathcal{A}\|), \quad \text{where } k_1 := \text{tw}(\mathcal{A}), \text{ and } k_2 := |\varphi|,$$
$$f \text{ computable and non-decreasing}$$

Courcelle's Theorem

---

p^*-*tw*-MC(MSO)

    Instance: A structure $\mathcal{A}$ and an MSO-sentence φ.

    Parameter: $tw(\mathcal{A}) + |\varphi|$.

    Problem: Decide whether $\mathcal{A} \vDash \varphi$.

Theorem ([Courcelle, 1990])

p^*-*tw*-MC(MSO) $\in$ FPT. *More precisely, the problem is decidable by an algorithm in time:*

$$f(k_1, k_2) \cdot |A| + O(\|\mathcal{A}\|), \quad \text{where } k_1 := tw(\mathcal{A}), \text{ and } k_2 := |\varphi|,$$
$$f \text{ computable and non-decreasing}$$

$f(k_1, k_2) \cdot |A| + O(\|\mathcal{A}\|) \le f(k_1, k_2) \cdot |A| + c \cdot \|\mathcal{A}\|$ with some $c > 0$

Courcelle's Theorem

p^*-tw-MC(MSO)
   Instance: A structure $\mathcal{A}$ and an MSO-sentence φ.
   Parameter: $tw(\mathcal{A}) + |\varphi|$.
   Problem: Decide whether $\mathcal{A} \vDash \varphi$.

Theorem ([Courcelle, 1990])

p^*-tw-MC(MSO) $\in$ FPT. *More precisely, the problem is decidable by an algorithm in time:*

$$f(k_1, k_2) \cdot |A| + O(\|\mathcal{A}\|), \quad \textit{where } k_1 := tw(\mathcal{A}), \textit{ and } k_2 := |\varphi|,$$
$$f \textit{ computable and non-decreasing}$$

$$f(k_1, k_2) \cdot |A| + O(\|\mathcal{A}\|) \le f(k_1, k_2) \cdot |A| + c \cdot \|\mathcal{A}\| \quad \text{with some } c > 0$$
$$\le (f(k_1, k_2) + c) \cdot \|\mathcal{A}\|$$

Courcelle's Theorem

p^*-tw-MC(MSO)

Instance: A structure $\mathcal{A}$ and an MSO-sentence φ.
Parameter: $tw(\mathcal{A}) + |\varphi|$.
Problem: Decide whether $\mathcal{A} \vDash \varphi$.

Theorem ([Courcelle, 1990])

p^*-tw-MC(MSO) $\in$ FPT. *More precisely, the problem is decidable by an algorithm in time:*

$$f(k_1, k_2) \cdot |A| + O(\|\mathcal{A}\|), \quad \text{where } k_1 := tw(\mathcal{A}), \text{ and } k_2 := |\varphi|,$$
$$f \text{ computable and non-decreasing}$$

$$f(k_1, k_2) \cdot |A| + O(\|\mathcal{A}\|) \leq f(k_1, k_2) \cdot |A| + c \cdot \|\mathcal{A}\| \quad \text{with some } c > 0$$
$$\leq (f(k_1, k_2) + c) \cdot \|\mathcal{A}\|$$
$$\leq g(k) \cdot (\|\mathcal{A}\| + |\varphi|) \quad \text{for } g(x) := f(x, x) + c$$
$$k := k_1 + k_2$$

Courcelle's Theorem

p^*-*tw*-MC(MSO)

   Instance: A structure $\mathcal{A}$ and an MSO-sentence φ.
   Parameter: $tw(\mathcal{A}) + |\varphi|$.
   Problem: Decide whether $\mathcal{A} \vDash \varphi$.

Theorem ([Courcelle, 1990])

p^*-*tw*-MC(MSO) $\in$ FPT. *More precisely, the problem is decidable by an algorithm in time:*

$$f(k_1, k_2) \cdot |A| + O(\|\mathcal{A}\|), \quad \text{where } k_1 := tw(\mathcal{A}), \text{ and } k_2 := |\varphi|,$$
$$f \text{ computable and non-decreasing}$$

$$
\begin{aligned}
f(k_1, k_2) \cdot |A| + O(\|\mathcal{A}\|) &\leq f(k_1, k_2) \cdot |A| + c \cdot \|\mathcal{A}\| && \text{with some } c > 0 \\
&\leq (f(k_1, k_2) + c) \cdot \|\mathcal{A}\| \\
&\leq g(k) \cdot (\|\mathcal{A}\| + |\varphi|) && \text{for } g(x) := f(x, x) + c \\
& && k := k_1 + k_2 \\
&\leq g(k) \cdot n && \text{where } n := \|\mathcal{A}\| + |\varphi|
\end{aligned}
$$

Vertex Cover (first attempt)

Let $\mathcal{G} = \langle V, E \rangle$ a graph. For all $S \subseteq V$:

S is a vertex cover of $\mathcal{G}$ $:\Longleftrightarrow \forall e = \{u, v\} \in E \, (\, u \in S \lor v \in S \,))$

$p^*\text{-}tw\text{-}\text{VERTEX-COVER}$

Instance: A graph $\mathcal{G} = \langle V, E \rangle$, and $\ell \in \mathbb{N}$.

Instance: $tw(\mathcal{G})$.

Problem: Does $\mathcal{G}$ have a vertex cover of size at most ℓ?

Courcelle's Theorem: Refinement 1

p^*-*tw*-MC$^\leq$(MSO)
   Instance: A structure $\mathcal{A}$, an $\varphi(X)$, and $m \in \mathbb{N}$.
   Parameter: *tw*$(\mathcal{A}) + |\varphi(X)|$.
   Problem: Decide whether $\mathcal{A} \vDash \exists X (\, card^{\leq m}(X) \wedge \varphi(X)\,)$.

Refinement 1 of Courcelle's Theorem

p^*-*tw*-MC$^\leq$(MSO) $\in$ FPT. More precisely, the problem is decidable by an algorithm in time:

$$f(k_1, k_2) \cdot |A| + O(\|\mathcal{A}\|)\,, \quad \text{where } k_1 := \textit{tw}(\mathcal{A}), \text{ and } k_2 := |\varphi|,$$
$$f \text{ computable and non-decreasing}$$

Vertex Cover

Let $\mathcal{G} = \langle V, E \rangle$ a graph. For all $S \subseteq V$:

S is a vertex cover of $\mathcal{G}$ $:\Longleftrightarrow \forall e = \{u, v\} \in E\, (\, u \in S \vee v \in S\,)$

---

p^*-*tw*-VERTEX-COVER

Instance: A graph $\mathcal{G} = \langle V, E \rangle$, and $\ell \in \mathbb{N}$.

Instance: *tw*($\mathcal{G}$).

Problem: Does $\mathcal{G}$ have a vertex cover of size at most ℓ?

Example

p^*-*tw*-VERTEX-COVER $\in$ FPT.

Vertex Cover

Let $\mathcal{G} = \langle V, E \rangle$ a graph. For all $S \subseteq V$:

S is a vertex cover of $\mathcal{G}$:$\iff \forall e = \{u, v\} \in E \ (u \in S \lor v \in S))$

$p^*\text{-}tw\text{-}$VERTEX-COVER

Instance: A graph $\mathcal{G} = \langle V, E \rangle$, and $\ell \in \mathbb{N}$.
Instance: $tw(\mathcal{G})$.
Problem: Does $\mathcal{G}$ have a vertex cover of size at most ℓ?

Example

$p^*\text{-}tw\text{-}$VERTEX-COVER $\in$ FPT.

Courcelle's Theorem: Refinement 2

p^*-*tw*-MC$^=$(MSO)
  Instance: A structure $\mathcal{A}$, an MSO-sentence $\varphi(X)$, and $m \in \mathbb{N}$.
  Parameter: $tw(\mathcal{A}) + |\varphi(X)|$.
  Problem: Decide whether $\mathcal{A} \vDash \exists X (\, card^{=m}(X) \wedge \varphi(X) \,)$.

Refinement 2 of Courcelle's Theorem

p^*-*tw*-MC$^=$(MSO) $\in$ FPT. More precisely, the problem is decidable by an algorithm in time:

$$f(k_1, k_2) \cdot |A|^2 + O(\|\mathcal{A}\|), \quad \text{where } k_1 := tw(\mathcal{A}), \text{ and } k_2 := |\varphi|,$$
$$f \text{ computable and non-decreasing}$$

Courcelle's Theorem Ref. 3: Optimization version

p^*-*tw*-opt-MC(MSO)

    Instance: A graph $\mathcal{G} = \langle V, E \rangle$, an MSO-sentence $\varphi(X_1, \ldots, X_p)$.

    Parameter: $tw(\mathcal{G}) + |\varphi(X_1, \ldots, X_p)|$.

    Compute: $\genfrac{}{}{0pt}{}{\max}{\min} \left\{ \alpha(|X_1|, \ldots, |X_p|) \mid \begin{array}{l} X_1, \ldots, X_p \subseteq V \cup E \\ \mathcal{A}(\mathcal{G}) \vDash \varphi(X_1, \ldots, X_p). \end{array} \right\}.$

        where α is an affine function $\alpha(x_1, \ldots, x_p) = a_0 + \sum_{i=1}^{p} a_i \cdot x_i$.

Optimization version of Courcelle's Theorem

p^*-*tw*-opt-MC(MSO) $\in$ FPT, and it is decidable by an algorithm in time:

    $f(tw(\mathcal{G}), |\varphi|) \cdot |V|$, where f computable and non-decreasing.

Maximum 2-edge colorable subgraphs

p^*-*tw*-max-2-edge-colorable-subgraph
    Instance: A graph $\mathcal{G} = \langle V, E \rangle$.
    Parameter: $tw(\mathcal{G})$.
    Compute: Maximum number of edges
                in a 2-edge colored subgraph of G.

Example [AA & Vahan Mkrtchyan]

p^*-*tw*-max-2-edge-colorable-subgraph $\in$ FPT.

Maximum 2-edge colorable subgraphs

p^*-*tw*-max-2-edge-colorable-subgraph
  Instance: A graph $\mathcal{G} = \langle V, E \rangle$.
  Parameter: $tw(\mathcal{G})$.
  Compute: Maximum number of edges
       in a 2-edge colored subgraph of G.

Example [AA & Vahan Mkrtchyan]

p^*-*tw*-max-2-edge-colorable-subgraph $\in$ FPT.

Courcelle's Theorem: applications (3)

p^*-*tw*-INDEPENDENT-SET
  Instance: A graph $\mathcal{G}$, a number $\ell \in \mathbb{N}$.
  Parameter: $tw(\mathcal{G})$
  Problem: Decide whether $\mathcal{G}$ has an independent set of ℓ elements.

Example

p^*-*tw*-INDEPENDENT-SET $\in$ FPT.

Courcelle's Theorem: applications (3)

---

p^*-*tw*-INDEPENDENT-SET

Instance: A graph $\mathcal{G}$, a number $\ell \in \mathbb{N}$.

Parameter: $tw(\mathcal{G})$

Problem: Decide whether $\mathcal{G}$ has an independent set of ℓ elements.

Example

p^*-*tw*-INDEPENDENT-SET $\in$ FPT.

Courcelle's Theorem: applications (4)

---

p^*-*tw*-FEEDBACK-VERTEX-SET
    Instance: A graph $\mathcal{G}$ and $\ell \in \mathbb{N}$.
    Parameter: $tw(\mathcal{C})$
    Problem: Decide whether $\mathcal{G}$ has a feedback vertex set of ℓ
               elements.

Example

p^*-*tw*-FEEDBACK-VERTEX-SET $\in$ FPT.

Courcelle's Theorem: applications (4)

---

p^*-*tw*-FEEDBACK-VERTEX-SET

Instance: A graph $\mathcal{G}$ and $\ell \in \mathbb{N}$.

Parameter: $tw(\mathcal{C})$

Problem: Decide whether $\mathcal{G}$ has a feedback vertex set of ℓ elements.

Example

p^*-*tw*-FEEDBACK-VERTEX-SET $\in$ FPT.

Courcelle's Theorem: applications (5)

p^*-*tw*-CROSSING-NUMBER
   Instance: A graph $\mathcal{G}$, and $k \in \mathbb{N}$
   Parameter: $tw(\mathcal{G}) + k$
   Problem: Decide whether the crossing number of $\mathcal{G}$ is k.

Example

p^*-*tw*-CROSSING-NUMBER $\in$ FPT.

The *crossing number* is the least number of edge crossings required to draw the graph in the plane such that at each point at most two edges cross.

Courcelle's Theorem: applications (5)

Definition

Let $\mathcal{G}_1 = \langle V_1, E_1 \rangle$ and $\mathcal{G}_2 = \langle V_2, E_2 \rangle$ be graphs.
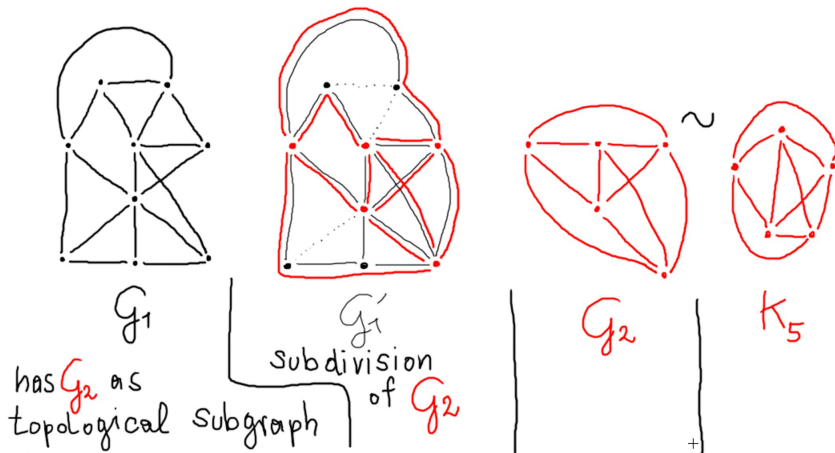$\mathcal{G}_1$ is a *subdivision* of $\mathcal{G}_2$ if:

- $\mathcal{G}_1$ arises by splitting the edges of $\mathcal{G}_2$
  into paths with intermediate vertices.

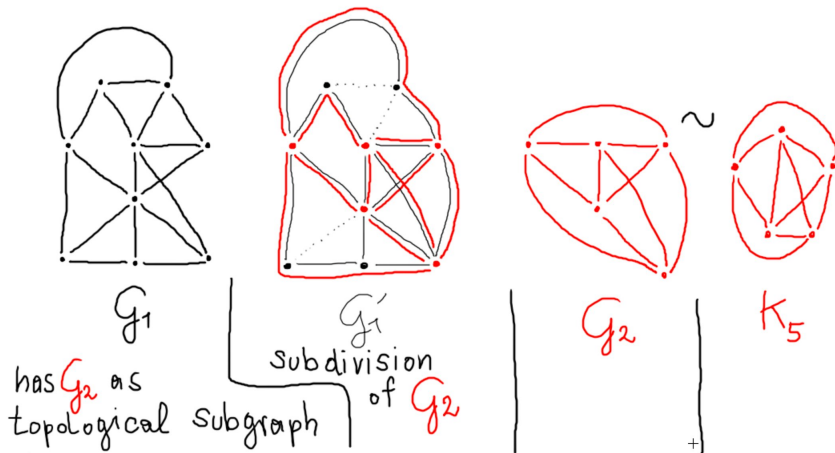$\mathcal{H}$ is a *topological subgraph* of $\mathcal{G}$
  if $\mathcal{G}$ has a subgraph that is a subdivision of $\mathcal{H}$.

Courcelle's Theorem: applications (5)

Definition

Let $\mathcal{G}_1 = \langle V_1, E_1 \rangle$ and $\mathcal{G}_2 = \langle V_2, E_2 \rangle$ be graphs.
$\mathcal{G}_1$ is a *subdivision* of $\mathcal{G}_2$ if:

- $\mathcal{G}_1$ arises by splitting the edges of $\mathcal{G}_2$
  into paths with intermediate vertices.

$\mathcal{H}$ is a *topological subgraph* of $\mathcal{G}$
  if $\mathcal{G}$ has a subgraph that is a subdivision of $\mathcal{H}$.

Courcelle's Theorem: applications (5)

Courcelle's Theorem: applications (5)

Theorem (Kuratowski)

A graph is planar if and only if it contains neither $\mathcal{K}_5$ nor $\mathcal{K}_{3,3}$ as topological subgraph.

Courcelle's Theorem: applications (5)

Theorem (Kuratowski)

A graph is planar if and only if it contains neither $\mathcal{K}_5$ nor $\mathcal{K}_{3,3}$ as topological subgraph.

Courcelle's Theorem: applications (5)

Theorem (Kuratowski)

A graph is planar if and only if it contains neither $\mathcal{K}_5$ nor $\mathcal{K}_{3,3}$ as topological subgraph.

Lemma

There is a MSO(τ_{HG}) formula *top-sub*$_{\mathcal{H}}$ such that for every graph $\mathcal{G}$:

$$\mathcal{A}_{\tau_{\mathsf{HG}}}(\mathcal{G}) \vDash \textit{top-sub}_{\mathcal{H}} \iff \mathcal{H} \text{ is a topological subgraph of } \mathcal{G}.$$

Courcelle's Theorem: applications (5)

Theorem (Kuratowski)

A graph is planar if and only if it contains neither $\mathcal{K}_5$ nor $\mathcal{K}_{3,3}$ as topological subgraph.

Lemma

There is a MSO(τ_{HG}) formula *top-sub$_\mathcal{H}$* such that for every graph $\mathcal{G}$:

$$\mathcal{A}_{\tau_{\mathsf{HG}}}(\mathcal{G}) \vDash \textit{top-sub}_\mathcal{H} \iff \mathcal{H} \text{ is a topological subgraph of } \mathcal{G}.$$

Using MSO(τ_{HG}) formula *path*(x, y, Z) that Z is a path from x to y.

Courcelle's Theorem: applications (5)

Theorem (Kuratowski)

A graph is planar if and only if it contains neither $\mathcal{K}_5$ nor $\mathcal{K}_{3,3}$ as topological subgraph.

Lemma

There is a MSO(τ_{HG}) formula *top-sub$_{\mathcal{H}}$* such that for every graph $\mathcal{G}$:

$$\mathcal{A}_{\tau_{\mathsf{HG}}}(\mathcal{G}) \vDash \textit{top-sub}_{\mathcal{H}} \iff \mathcal{H} \text{ is a topological subgraph of } \mathcal{G}.$$

Using MSO(τ_{HG}) formula *path(x, y, Z)* that Z is a path from x to y.

Lemma

There is a MSO(τ_{HG}) formula *cross$_k$* such that for every graph $\mathcal{G}$:

$$\mathcal{A}_{\tau_{\mathsf{HG}}}(\mathcal{G}) \vDash \textit{cross}_k \iff \text{the crossing number of } \mathcal{G} \text{ is at most } k.$$

Proof: By induction, where *cross$_0$* := $\neg$*top-sub$_{\mathcal{K}_5}$* $\wedge$ $\neg$*top-sub$_{\mathcal{K}_{3,3}}$*.

Computably boundedness between notions of width

(from Sasák, [Sásak, 2010])

$$g(wd_1) \geq wd_2 \ :\Leftrightarrow wd_1 \xrightarrow{g} wd_2$$
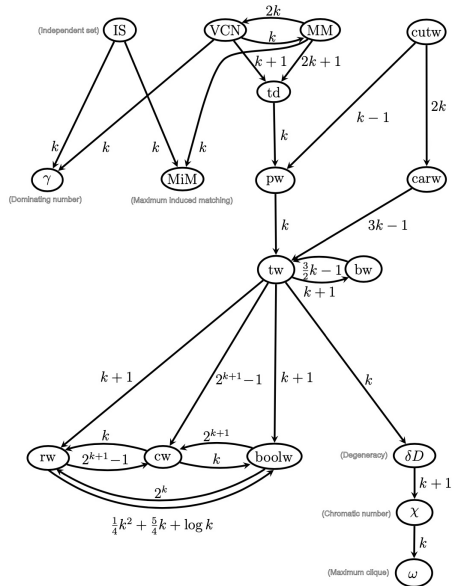
- FPT-results
  transfer upwards
  (and conversely to $\xrightarrow{g}$)

- ($\notin$ FPT)-results
  transfer downwards
  (and along $\xrightarrow{g}$)

Comparing parameterizations

Definition (computably bounded below)

Let $\kappa_1, \kappa_2 : \Sigma^* \to \mathbb{N}$ parameterizations.

- $\kappa_1 \geq \kappa_2 : \Longleftrightarrow \exists g : \mathbb{N} \to \mathbb{N}$ computable $\forall x \in \Sigma^* \big[\, g(\kappa_1(x)) \geq \kappa_2(x) \,\big]$.

- $\kappa_1 \approx \kappa_2 : \Longleftrightarrow \kappa_1 \geq \kappa_2 \,\wedge\, \kappa_2 \geq \kappa_1$.

- $\kappa_1 > \kappa_2 : \Longleftrightarrow \kappa_1 \geq \kappa_2 \,\wedge\, \neg(\kappa_2 \geq \kappa_1)$.

Proposition

For all parameterized problems $\langle Q, \kappa_1 \rangle$ and $\langle Q, \kappa_2 \rangle$ with parameterizations $\kappa_1, \kappa_2 : \Sigma^* \to \mathbb{N}$ with $\kappa_1 \geq \kappa_2$:

$$\langle Q, \kappa_1 \rangle \in \mathsf{FPT} \quad \Longleftarrow \quad \langle Q, \kappa_2 \rangle \in \mathsf{FPT}$$
$$\langle Q, \kappa_1 \rangle \notin \mathsf{FPT} \quad \Longrightarrow \quad \langle Q, \kappa_2 \rangle \notin \mathsf{FPT}$$

Courcelle's Theorem for clique-width

Recall that $\mathsf{MSO}(\tau_\mathsf{G}) \sim \mathsf{MSO}_1$ (quantification over sets of vertices, but not sets of edges).

---

p^*-*clw*-MC($\mathsf{MSO}(\tau_\mathsf{G})/\mathsf{MSO}_1$)
    Instance: A graph $\mathcal{G}$ and an $\mathsf{MSO}(\tau_\mathsf{G})$-sentence φ.
    Parameter: *clw*$(\mathcal{G}) + |\varphi|$.
    Problem: Decide whether $\mathcal{A}(\mathcal{G}) \vDash \varphi$.

Theorem ([Courcelle et al., 2000])

p^*-*clw*-MC($\mathsf{MSO}(\tau_\mathsf{G})/\mathsf{MSO}_1$) $\in$ FPT.

Also, there is a maximization version of this theorem.

Courcelle's Theorem for clique-width (example)

Let $\mathcal{G} = \langle V, E \rangle$ a graph. For all $S \subseteq V$:

S is a vertex cover of $\mathcal{G}$ $:\Longleftrightarrow \forall e = \{u, v\} \in E \, (\, u \in S \lor v \in S \,))$

$p^*\text{-}clw\text{-}\text{VERTEX-COVER}$

Instance: A graph $\mathcal{G} = \langle V, E \rangle$, and $\ell \in \mathbb{N}$.

Instance: $clw(\mathcal{G})$.

Problem: Does $\mathcal{G}$ have a vertex cover of size at most ℓ?

Example

$p^*\text{-}clw\text{-}\text{VERTEX-COVER} \in \text{FPT}$.

Courcelle's Theorem for clique-width (example)

Let $\mathcal{G} = \langle V, E \rangle$ a graph. For all $S \subseteq V$:

S is a vertex cover of $\mathcal{G}$ $:\Longleftrightarrow \forall e = \{u, v\} \in E\,(\,u \in S \lor v \in S\,)\,)$

$p^*\text{-}\textit{clw}\text{-}\textsc{Vertex-Cover}$
   Instance: A graph $\mathcal{G} = \langle V, E \rangle$, and $\ell \in \mathbb{N}$.
   Instance: $\textit{clw}(\mathcal{G})$.
   Problem: Does $\mathcal{G}$ have a vertex cover of size at most ℓ?

Example

$p^*\text{-}\textit{clw}\text{-}\textsc{Vertex-Cover} \in$ FPT.

Application to maximum 2-edge colorable subgraphs?

---

p^*-**clw**-max-2-edge-colorable-subgraph
    Instance: A graph $\mathcal{G} = \langle V, E \rangle$.
    Parameter: **clw**$(\mathcal{G})$.
    Compute: Maximum number of edges
                in a 2-edge colored subgraph of G.

Open problem [AA, Vahan Mkrtchyan]

p^*-**clw**-max-2-edge-colorable-subgraph $\in$ FPT ?

Application to maximum 2-edge colorable subgraphs?

p^*-*clw*-max-2-edge-colorable-subgraph
   Instance: A graph $\mathcal{G} = \langle V, E \rangle$.
   Parameter: *clw*$(\mathcal{G})$.
   Compute: Maximum number of edges
          in a 2-edge colored subgraph of G.

Open problem [AA, Vahan Mkrtchyan]

p^*-*clw*-max-2-edge-colorable-subgraph $\in$ FPT ?

We saw that there is a MSO$_2$ formula $\varphi(X)$ such that:

$$\mathcal{A}_{\tau_{\mathsf{HG}}}(\mathcal{G}) \vDash \varphi(S) \iff S \subseteq E \text{ is an 2-colorable set of edges in } \mathcal{G}$$

But there seems not to be such an MSO$_1$ formula.

Courcelle's Theorem for clique-width (non-example)

p^*-**clw**-HAMILTONICITY
  Instance: A graph $\mathcal{G}$
  Parameter: **clw**$(\mathcal{C})$
  Problem: Decide whether $\mathcal{G}$ is a hamiltonian (that is, contains
              a cyclic path that visits every vertex precisely once).

Courcelle's Theorem for clique-width (non-example)

p^*-*clw*-HAMILTONICITY

Instance: A graph $\mathcal{G}$

Parameter: *clw*$(\mathcal{C})$

Problem: Decide whether $\mathcal{G}$ is a hamiltonian (that is, contains a cyclic path that visits every vertex precisely once).

Recall

There is no MSO_1 formula that expresses Hamiltonicity.

Courcelle's Theorem for clique-width (non-example)

p^*-*clw*-HAMILTONICITY
  Instance: A graph $\mathcal{G}$
  Parameter: *clw*$(\mathcal{C})$
  Problem: Decide whether $\mathcal{G}$ is a hamiltonian (that is, contains
          a cyclic path that visits every vertex precisely once).

Recall

There is no MSO_1 formula that expresses Hamiltonicity.

Hence we cannot apply Courcelle's Theorem for clique-width.

Courcelle's Theorem for clique-width (non-example)

p^*-**clw**-HAMILTONICITY
Instance: A graph $\mathcal{G}$
Parameter: **clw**$(\mathcal{C})$
Problem: Decide whether $\mathcal{G}$ is a hamiltonian (that is, contains a cyclic path that visits every vertex precisely once).

Recall

There is no MSO$_1$ formula that expresses Hamiltonicity.

Hence we cannot apply Courcelle's Theorem for clique-width. Indeed:

Theorems

(T1) p^*-**clw**-HAMILTONICITY $\notin$ FPT,
since it is not decidable in time $\notin n^{o(\textbf{clw}(\mathcal{C}))}$ (Fomin et al, 2014).

(T2) p^*-**clw**-HAMILTONICITY $\in O(n^{o(\textbf{clw}(\mathcal{C}))})$
(Bergougnoux, Kanté, Kwon, 2020).

Computably boundedness between notions of width

(from Sasák, [Sásak, 2010])

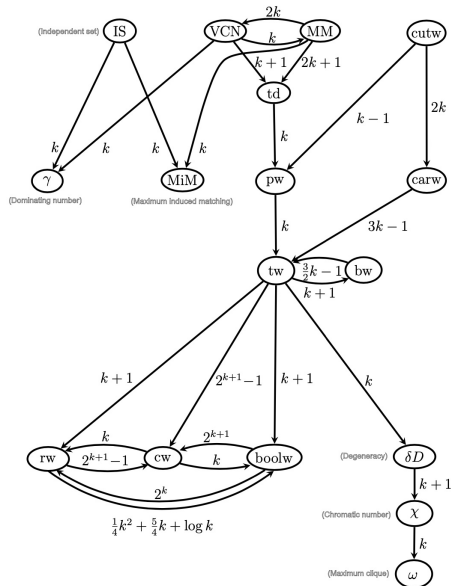$$g(wd_1) \geq wd_2 \; : \Leftrightarrow \; wd_1 \xrightarrow{g} wd_2$$

- ▶ FPT-results
  transfer upwards
  (and conversely to $\xrightarrow{g}$)

- ▶ ($\notin$ FPT)-results
  transfer downwards
  (and along $\xrightarrow{g}$)

Graph Minors

Graph minors

Definition

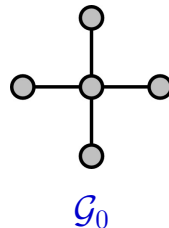A graph $\mathcal{G}_0$ is a *minor* of a graph $\mathcal{G}$ if $\mathcal{G}_0$ is obtained by:
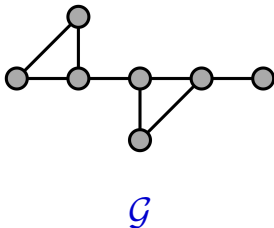
- deleting some edges,
- deleting arising isolated vertices,
- contracting edges in $\mathcal{G}$.

Graph minors

Definition

A graph $\mathcal{G}_0$ is a *minor* of a graph $\mathcal{G}$ if $\mathcal{G}_0$ is obtained by:

- deleting some edges,
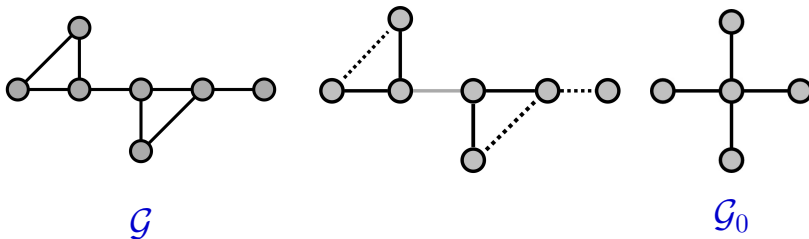- deleting arising isolated vertices,
- contracting edges in $\mathcal{G}$.

$$\mathcal{G} \qquad\qquad\qquad\qquad \mathcal{G}_0$$

Graph minors

Definition

A graph $\mathcal{G}_0$ is a *minor* of a graph $\mathcal{G}$ if $\mathcal{G}_0$ is obtained by:

- deleting some edges,
- deleting arising isolated vertices,
- contracting edges in $\mathcal{G}$.

$\mathcal{G}$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $\mathcal{G}_0$

Excluded minors

Definition (minor closed)

A class $\mathcal{G}$ is *minor closed* if for every $\mathcal{G} \in \boldsymbol{\mathcal{G}}$ all minors of $\mathcal{G}$ are in $\boldsymbol{\mathcal{G}}$.

Excluded minors

Definition (minor closed)

A class $\mathcal{G}$ is *minor closed* if for every $\mathcal{G} \in \mathcal{G}$ all minors of $\mathcal{G}$ are in $\mathcal{G}$.

We say that a class $\mathcal{G}$ is characterized by excluded minors in $\mathcal{H}$ if:

$$\mathcal{G} := \text{Excl}(\mathcal{H}) := \{\mathcal{G} \mid \mathcal{G} \text{ does not have a minor in } \mathcal{H}\}$$

Excluded minors

Definition (minor closed)

A class $\mathcal{G}$ is *minor closed* if for every $\mathcal{G} \in \mathcal{G}$ all minors of $\mathcal{G}$ are in $\mathcal{G}$.

We say that a class $\mathcal{G}$ is characterized by excluded minors in $\mathcal{H}$ if:

$$\mathcal{G} := \text{Excl}(\mathcal{H}) := \{\mathcal{G} \mid \mathcal{G} \text{ does not have a minor in } \mathcal{H}\}$$

Theorem (Graph Minor Theorem (Robertson–Seymour, 1983–2004))

Every class of graphs that is minor closed can be characterized by finitely many excluded minors. That is, for every class $\mathcal{G}$ of minor closed graphs there are graphs $\mathcal{H}_1, \ldots, \mathcal{H}_m$ such that:

$$\mathcal{G} = \text{Excl}(\{\mathcal{H}_1, \ldots, \mathcal{H}_m\}).$$

Deciding minor closed classes

p-MINOR
  Instance: Graphs $\mathcal{G}$ and $\mathcal{H}$.
  Parameter: $\|\mathcal{G}\|$
  Problem: Decide whether $\mathcal{G}$ is a minor of $\mathcal{H}$.

Deciding minor closed classes

p-MINOR
Instance: Graphs $\mathcal{G}$ and $\mathcal{H}$.
Parameter: $\|\mathcal{G}\|$
Problem: Decide whether $\mathcal{G}$ is a minor of $\mathcal{H}$.

Theorem

p-MINOR $\in$ FPT, *decidable in time* $f(k) \cdot n^3$ *where* $k = \|\mathcal{G}\|$, *and* n *is the number of vertices of* $\mathcal{H}$.

Deciding minor closed classes

p-MINOR
> **Instance:** Graphs $\mathcal{G}$ and $\mathcal{H}$.
> **Parameter:** $\|\mathcal{G}\|$
> **Problem:** Decide whether $\mathcal{G}$ is a minor of $\mathcal{H}$.

Theorem

p-MINOR $\in$ FPT, decidable in time $f(k) \cdot n^3$ where $k = \|\mathcal{G}\|$, and n is the number of vertices of $\mathcal{H}$.

Corollary

Every minor-closed class of graphs is decidable in cubic time.

Deciding minor closed classes

p-MINOR
Instance: Graphs $\mathcal{G}$ and $\mathcal{H}$.
Parameter: $\|\mathcal{G}\|$
Problem: Decide whether $\mathcal{G}$ is a minor of $\mathcal{H}$.

Theorem

p-MINOR $\in$ FPT, *decidable in time* $f(k) \cdot n^3$ *where* $k = \|\mathcal{G}\|$, *and* n *is the number of vertices of* $\mathcal{H}$.

Corollary

Every minor-closed class of graphs is decidable in cubic time.

Corollary

Let $\langle Q, \kappa \rangle$ *be a parameterized problem on graphs such that for every* $k \in \mathbb{N}$, *either* $\{\mathcal{G} \in Q \mid \kappa(\mathcal{G}) = k\}$ *or* $\{\mathcal{G} \notin Q \mid \kappa(\mathcal{G}) = k\}$ *is minor closed.*

Then every slice $\langle Q, \kappa \rangle_k$ *is decidable in cubic time. In this case we can say that* $\langle Q, \kappa \rangle$ *is nonuniformly fixed-parameter tractable.*

Non-uniformly fixed-parameter tractable

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is *fixed-parameter tractable* if:

$\exists f : \mathbb{N} \to \mathbb{N}$ computable $\exists p \in \mathbb{N}[X]$ polynomial

$\exists \mathbb{A}$ algorithm, takes inputs in Σ^*

$\forall x \in \Sigma^* \big[\mathbb{A}$ decides whether $x \in Q$ holds

in time $\leq f(\kappa(x)) \cdot p(|x|) \big]$

Non-uniformly fixed-parameter tractable

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is *fixed-parameter tractable* if:

$$\exists f : \mathbb{N} \to \mathbb{N} \text{ computable } \exists p \in \mathbb{N}[X] \text{ polynomial}$$

$$\exists \mathbb{A} \text{ algorithm, takes inputs in } \Sigma^*$$

$$\forall x \in \Sigma^* \big[\mathbb{A} \text{ decides whether } x \in Q \text{ holds}$$
$$\text{in time } \leq f(\kappa(x)) \cdot p(|x|) \big]$$

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is *non-uniformly fixed-parameter tractable* (in nu-FPT) if:

$$\exists f : \mathbb{N} \to \mathbb{N} \text{ computable } \exists p \in \mathbb{N}[X] \text{ polynomial}$$

$$\exists \{\mathbb{A}_k\}_{k \in \mathbb{N}} \text{ algorithms, takes inputs in } \Sigma^*$$

$$\forall x \in \Sigma^* \big[\mathbb{A}_{\kappa(x)} \text{ decides whether } x \in Q \text{ holds}$$
$$\text{in time } \leq f(\kappa(x)) \cdot p(|x|) \big]$$

Using minor-closed classes for FPT results

Corollary

Let $\langle Q, \kappa \rangle$ be a parameterized problem on graphs such that for every $k \in \mathbb{N}$, either $\{\mathcal{G} \in Q \mid \kappa(\mathcal{G}) = k\}$ or $\{\mathcal{G} \notin Q \mid \kappa(\mathcal{G}) = k\}$ is minor closed. Then $\langle Q, \kappa \rangle$ is non-uniformly fixed-parameter tractable (in nu-FPT).

Applications:

- p-VERTEX-COVER $\in$ nu-FPT (p-VERTEX-COVER is minor closed).

Using minor-closed classes for FPT results

Corollary

Let $\langle Q, \kappa \rangle$ be a parameterized problem on graphs such that for every $k \in \mathbb{N}$, either $\{\mathcal{G} \in Q \mid \kappa(\mathcal{G}) = k\}$ or $\{\mathcal{G} \notin Q \mid \kappa(\mathcal{G}) = k\}$ is minor closed. Then $\langle Q, \kappa \rangle$ is non-uniformly fixed-parameter tractable (in nu-FPT).

Applications:

▶ p-VERTEX-COVER $\in$ nu-FPT (p-VERTEX-COVER is minor closed).

▶ p-FEEDBACK-VERTEX-SET $\in$ nu-FPT (problem is minor closed).

Using minor-closed classes for FPT results

Corollary

Let $\langle Q, \kappa \rangle$ be a parameterized problem on graphs such that for every $k \in \mathbb{N}$, either $\{ \mathcal{G} \in Q \mid \kappa(\mathcal{G}) = k \}$ or $\{ \mathcal{G} \notin Q \mid \kappa(\mathcal{G}) = k \}$ is minor closed. Then $\langle Q, \kappa \rangle$ is non-uniformly fixed-parameter tractable (in nu-FPT).

Applications:

▶ p-VERTEX-COVER ∈ nu-FPT (p-VERTEX-COVER is minor closed).

▶ p-FEEDBACK-VERTEX-SET ∈ nu-FPT (problem is minor closed).

▶
p-DISJOINT-CYCLES
  Instance: A graph $\mathcal{G}$, and $k \in \mathbb{N}$.
  Parameter: k.
  Problem: Decide whether $\mathcal{G}$ has k disjoint cycles.

p-DISJOINT-CYCLES ∈ nu-FPT , since the class of graphs that do not have k disjoint cycles is minor closed.

First-Order Meta-Theorem (example)

Seese's theorem

A class $\mathcal{G}$ of graphs has *bounded degree* if there is $d \in \mathbb{N}$ such that $\Delta(\mathcal{G}) \leq d$ for all $\mathcal{G} \in \mathcal{G}$ (where $\Delta(\mathcal{G})$ = max. degree of vertex in $\mathcal{G}$).

p-MC(FO, $\mathcal{G}$)
   Instance: A graph $\mathcal{G} \in \mathcal{G}$, and a f-o formula φ over τ_{HG}
   Parameter: $|\varphi|$.
   Problem: Decide whether $\mathcal{A}(\mathcal{G}) \vDash \varphi$.

Seese's theorem

A class $\mathcal{G}$ of graphs has *bounded degree* if there is $d \in \mathbb{N}$ such that $\Delta(\mathcal{G}) \leq d$ for all $\mathcal{G} \in \mathcal{G}$ (where $\Delta(\mathcal{G})$ = max. degree of vertex in $\mathcal{G}$).

$p\text{-MC}(\text{FO}, \mathcal{G})$

Instance: A graph $\mathcal{G} \in \mathcal{G}$, and a f-o formula φ over τ_{HG}
Parameter: $|\varphi|$.
Problem: Decide whether $\mathcal{A}(\mathcal{G}) \vDash \varphi$.

Theorem ([Seese, 1995])

$p\text{-MC}(\text{FO}, \mathcal{G}) \in \text{FPT}$ *for every class $\mathcal{G}$ of bounded degree. This model checking problem can be solved in time $f(|\varphi|) \cdot |\mathcal{G}|$, (linear in $|\mathcal{G}|$).*

Seese's theorem

A class $\mathcal{G}$ of graphs has *bounded degree* if there is $d \in \mathbb{N}$ such that $\Delta(\mathcal{G}) \leq d$ for all $\mathcal{G} \in \mathcal{G}$ (where $\Delta(\mathcal{G})$ = max. degree of vertex in $\mathcal{G}$).

---

p-MC(FO, $\mathcal{G}$)
   Instance: A graph $\mathcal{G} \in \mathcal{G}$, and a f-o formula φ over τ_{HG}
   Parameter: $|\varphi|$.
   Problem: Decide whether $\mathcal{A}(\mathcal{G}) \vDash \varphi$.

Theorem ([Seese, 1995])

p-MC(FO, $\mathcal{G}$) $\in$ FPT *for every class $\mathcal{G}$ of bounded degree. This model checking problem can be solved in time $f(|\varphi|) \cdot |\mathcal{G}|$, (linear in $|\mathcal{G}|$).*

Theorem (for comparison, we saw it earlier)

EVAL(FO) *and* MC(FO) *can be solved in time $O(|\varphi| \cdot |A|^w \cdot w)$, where w is the width of the input formula φ.*

Seese's theorem

A class $\mathcal{G}$ of graphs has *bounded degree* if there is $d \in \mathbb{N}$ such that $\Delta(\mathcal{G}) \leq d$ for all $\mathcal{G} \in \mathcal{G}$ (where $\Delta(\mathcal{G})$ = max. degree of vertex in $\mathcal{G}$).

---

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

Instance: A graph $\mathcal{G} \in \mathcal{G}$, and a f-o formula φ over τ_{HG}

Parameter: $|\varphi|$.

Problem: Decide whether $\mathcal{A}(\mathcal{G}) \vDash \varphi$.

Theorem ([Seese, 1995])

p-MC(FO, $\mathcal{G}$) $\in$ FPT *for every class $\mathcal{G}$ of bounded degree. This model checking problem can be solved in time $f(|\varphi|) \cdot |\mathcal{G}|$, (linear in $|\mathcal{G}|$).*

Example

On the class of graphs of bounded degree:

▶ p-CLIQUE $\in$ FPT ('is there a clique of size k')?

Seese's theorem

A class $\mathcal{G}$ of graphs has *bounded degree* if there is $d \in \mathbb{N}$ such that $\Delta(\mathcal{G}) \leq d$ for all $\mathcal{G} \in \mathcal{G}$ (where $\Delta(\mathcal{G}) =$ max. degree of vertex in $\mathcal{G}$).

$p\text{-MC}(\text{FO}, \mathcal{G})$

Instance: A graph $\mathcal{G} \in \mathcal{G}$, and a f-o formula φ over τ_{HG}
Parameter: $|\varphi|$.
Problem: Decide whether $\mathcal{A}(\mathcal{G}) \vDash \varphi$.

Theorem ([Seese, 1995])

$p\text{-MC}(\text{FO}, \mathcal{G}) \in \text{FPT}$ *for every class $\mathcal{G}$ of bounded degree. This model checking problem can be solved in time $f(|\varphi|) \cdot |\mathcal{G}|$, (linear in $|\mathcal{G}|$).*

Example

On the class of graphs of bounded degree:

▶ $p\text{-CLIQUE} \in \text{FPT}$ ('is there a clique of size k')?
▶ 'Does $\mathcal{G}$ contain a cycle of length k?' (parameter k) is in FPT.

Seese's theorem

A class $\mathcal{G}$ of graphs has *bounded degree* if there is $d \in \mathbb{N}$ such that $\Delta(\mathcal{G}) \leq d$ for all $\mathcal{G} \in \mathcal{G}$ (where $\Delta(\mathcal{G}) =$ max. degree of vertex in $\mathcal{G}$).

---

p-MC(FO, $\mathcal{G}$)
    Instance: A graph $\mathcal{G} \in \mathcal{G}$, and a f-o formula φ over τ_{HG}
    Parameter: $|\varphi|$.
    Problem: Decide whether $\mathcal{A}(\mathcal{G}) \models \varphi$.

Theorem ([Seese, 1995])

p-MC(FO, $\mathcal{G}$) $\in$ FPT *for every class $\mathcal{G}$ of bounded degree. This model checking problem can be solved in time* $f(|\varphi|) \cdot |\mathcal{G}|$, *(linear in* $|\mathcal{G}|$*).*

Example

On the class of graphs of bounded degree:

- p-CLIQUE $\in$ FPT ('is there a clique of size k')?
- 'Does $\mathcal{G}$ contain a cycle of length k?' (parameter k) is in FPT.
- p-VERTEX-COVER $\in$ FPT ('vertex cover of size at most k?')

First-order metatheorems: reference

A good reference for other meta-theorems for first-order logic is:

[Kreutzer, 2009]: Stephan Kreutzer: *Algorithmic Meta-Theorems*.

Summary

- ▶ Logic preliminaries
  - ▶ first-order logic
    - ▶ expressing graph problems by f-o formulas
  - ▶ monadic second-order logic (MSO)
    - ▶ expressing graph problems by MSO formulas
  - ▶ complexity of evaluation and model checking problems
- ▶ Courcelle's theorem
  - ▶ FPT-results by model-checking MSO-formulas
    - ▶ for graphs with bounded tree-width
    - ▶ for structures with bounded tree-width
    - ▶ for graphs of bounded clique-width
  - ▶ applications to concrete problems
- ▶ graph minors
- ▶ meta-theorems for first-order model-checking: an example

Tomorrow

Monday, June 19 10.00 – 12.00	Tuesday, June 20 10.00 – 12.00	Wednesday, June 21 10.00 – 12.00	Thursday, June 22 10.00 – 12.00	Friday, June 23 10.00 – 12.00
Introduction & basic FPT results	**Notions of bounded graph width**	**Guest Lecture Alessandro Aloisio**	**Algorithmic Meta-Theorems**	**FPT-Intractability Classes & Hierarchies**
motivation for FPT kernelization, Crown Lemma, Sunflower Lemma	path-, tree-, clique width, FPT-results by dynamic programming, transferring FPT results betw. widths	examples FPT-results: *firefighting problem*, *coverage in multi-interface networks*,	1st-order logic, monadic 2nd-order logic, FPT-results by Courcelle's Theorems for tree and clique-width	motivation for FP-intractability results, FPT-reductions, class XP (slicewise polynomial), W- and A-Hierarchies, placing problems on these hierarchies
Algorithmic Techniques			*Formal-Method & Algorithmic Techniques*	
				15.00 – 16.00
				Guest Exercise Class Alessandro Aloisio
				Intractability results on the firefighting problem

Example suggestions

Examples

1. Find a first-order logic formula over τ_G that expresses that a graph has a cycle of length precisely k.

2. Find an MSO_1 or $MSO(\tau_G)$ formula that expresses that a graph has a dominating set of $\leq k$ elements.

3. Find an MSO_2 or $MSO(\tau_{HG})$ formula *feedback*(S) that expresses that $S \subseteq V$ is a feedback vertex set.

4. (*) Find an MSO_1 or $MSO(\tau_G)$ formula that expresses that a graph is connected.

5. (*) Find an MSO_2 or $MSO(\tau_{HG})$ formula *path*(x, y, Z) that expresses that Z is a set of edges that forms a path from x to y.

References I

Courcelle, B. (1990).
The monadic second-order logic of graphs. i. recognizable sets of finite graphs.
Information and Computation, 85(1):12 – 75.

Courcelle, B., Makowsky, J. A., and Rotics, U. (2000).
Linear time solvable optimization problems on graphs of bounded clique-width.
Theory of Computing Systems, 33(2):125–150.

Cygan, M., Fomin, F. V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., and Saurabh, S. (2015).
Parameterized Algorithms.
Springer, 1st edition.

Flum, J. and Grohe, M. (2006).
Parameterized Complexity Theory.
Springer.

References II

📄 Kreutzer, S. (2009).
Algorithmic Meta-Theorems.
Technical report, arXiv.org.
https://arxiv.org/abs/0902.3616.

📄 Sásak, R. (2010).
Comparing 17 graph parameters.
Master's thesis, University of Bergen, Norway.

📄 Seese, D. (1995).
Linear time computable problems and logical descriptions.
Electronic Notes in Theoretical Computer Science, 2:246 – 259.
SEGRAGRA 1995, Joint COMPUGRAPH/SEMAGRAPH
Workshop on Graph Rewriting and Computation.