Lecture 4: Fixed-Parameter Intractability

(A Short Introduction to Parameterized Complexity)

Clemens Grabmayer

Ph.D. Program Advanced Course

Gran Sasso Science Institute

L'Aquila, Italy

June 12, 2026

Course overview

Monday, June 8 11.00 – 13.00	Tuesday, June 9 11.00 – 13.00	Wednesday, June 10 11.00 – 13.00	Thursday, June 11 11.00 – 13.00	Friday, June 13
Algorithmic Techniques			*Formal-Method & Algorithmic Techniques*	
Introduction & basic FPT results	**Notions of bounded graph width**	**Example: Good Edge Labelings**	**Algorithmic Meta-Theorems**	
motivation for FPT kernelization, Crown Lemma, Sunflower Lemma	path-, tree-, clique width, FPT-results by dynamic programming, transferring FPT results betw. width	*Guest Lecture* Davi de Andrade Iacono	1st-order logic, monadic 2nd-order logic, FPT-results by Courcelle's Theorems for tree and clique-width	
				14.30 – 16.30
				FPT-Intractability Classes & Hierarchies
				motivation for FP-intractability results, FPT-reductions, class XP (slicewise polynomial), W- and A-Hierarchies, placing problems on these hierarchies

Overview

- Motivation for fixed-parameter intractability
- Fixed parameter reductions
- The classes para-NP and XP
- The class W[P]
- Logic preliminaries (continued)
- W-hierarchy
  - definitions
    - with Boolean circuits
    - as parameterized weighted Fagin definability problems
- A-hierarchy
  - definition as parameterized model-checking problems
- picture overview of these classes

Two classical problems

QUERIES

Instance: a relational database D, a conjunctive query α.
Compute: answer to query α from database D.

Two classical problems

QUERIES
Instance: a relational database D, a conjunctive query α.
Compute: answer to query α from database D.

- QUERIES $\in$ NP-complete.

Two classical problems

QUERIES
Instance: a relational database D, a conjunctive query α.
Compute: answer to query α from database D.

- QUERIES ∈ NP-complete.

LTL-MODEL-CHECKING
Instance: a Kripke structure (state space) $\mathcal{K}$, an LTL formula φ
Question: Does $\mathcal{K} \vDash \varphi$ hold?

Two classical problems

QUERIES
Instance: a relational database D, a conjunctive query α.
Compute: answer to query α from database D.

- QUERIES ∈ NP-complete.

LTL-MODEL-CHECKING
Instance: a Kripke structure (state space) $\mathcal{K}$, an LTL formula φ
Question: Does $\mathcal{K} \vDash \varphi$ hold?

- LTL-MODEL-CHECKING ∈ PSPACE-complete.

Comparing their parameterizations

QUERIES
Instance: a relational database D, a conjunctive query α.
Parameter: size $k = |\alpha|$ of query α
Compute: answer to query α from database D.

- QUERIES $\in$ NP-complete.

LTL-MODEL-CHECKING
Instance: a Kripke structure (state space) $\mathcal{K}$, an LTL formula φ
Parameter: size $k = |\varphi|$ of formula φ
Question: Does $\mathcal{K} \vDash \varphi$ hold?

- LTL-MODEL-CHECKING $\in$ PSPACE-complete,

Comparing their parameterizations

QUERIES
Instance: a relational database D, a conjunctive query α.
Parameter: size $k = |\alpha|$ of query α
Compute: answer to query α from database D.

- QUERIES $\in$ NP-complete.
- QUERIES $\in O(n^k)$ for $n = \|D\|$, which does not give an FPT result.

LTL-MODEL-CHECKING
Instance: a Kripke structure (state space) $\mathcal{K}$, an LTL formula φ
Parameter: size $k = |\varphi|$ of formula φ
Question: Does $\mathcal{K} \models \varphi$ hold?

- LTL-MODEL-CHECKING $\in$ PSPACE-complete,

Comparing their parameterizations

QUERIES
Instance: a relational database D, a conjunctive query α.
Parameter: size $k = |\alpha|$ of query α
Compute: answer to query α from database D.

- QUERIES $\in$ NP-complete.
- QUERIES $\in O(n^k)$ for $n = \|D\|$, which does not give an FPT result.

LTL-MODEL-CHECKING
Instance: a Kripke structure (state space) $\mathcal{K}$, an LTL formula φ
Parameter: size $k = |\varphi|$ of formula φ
Question: Does $\mathcal{K} \models \varphi$ hold?

- LTL-MODEL-CHECKING $\in$ PSPACE-complete,
- LTL-MODEL-CHECKING $\in O(k \cdot 2^{2k} \cdot n) \in$ FPT for $n = \|\mathcal{K}\|$.

Fixed-parameter intractability

‘The purpose [...] is to give evidence that certain problems are not fixed-parameter tractable (just as the main purpose of the theory of NP*-completeness is to give evidence that certain problems are not polynomial time computable.)*

In classical theory, the notion of NP*-completeness is central to a nice, simple, and far-reaching theory for intractable problems.*

Unfortunately, the world of parameterized intractability is more complex: There is a big variety of seemingly different classes of intractable parameterized problems.’

(Flum, Grohe [2])

Fixed-Parameter tractable

Definition

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

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

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

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

Slices of parameterized problems

The *ℓ-th slice*, for $\ell \in \mathbb{N}$, of a parameterized problem $\langle Q, \kappa \rangle$ is:

$$\langle Q, \kappa \rangle_{\ell} := \{ x \in Q \mid \kappa(x) = \ell \} \ .$$

Proposition (slices of FPT problems are in PTIME)

Let $\langle Q, \kappa \rangle$ be a parameterized problem, and $\ell \in \mathbb{N}$.
If $\langle Q, \kappa \rangle \in$ FPT, then $\langle Q, \kappa \rangle_{\ell} \in$ PTIME.

Proof

Let ℓ be fixed. Then for all $x \in \Sigma^*$:
Decide $x \in Q$, $\kappa(x) = \ell$ in time $\le f(\kappa(x)) \cdot p(|x|) = f(\ell) \cdot p(|x|) \in$ PTIME.

A problem not in FPT

The *ℓ-th slice*, for $\ell \in \mathbb{N}$, of a parameterized problem $\langle Q, \kappa \rangle$ is:

$$\langle Q, \kappa \rangle_{\ell} := \{ x \in Q \mid \kappa(x) = \ell \} \ .$$

Slices of FPT problems are in PTIME

If $\langle Q, \kappa \rangle \in$ FPT, then $\langle Q, \kappa \rangle_{\ell} \in$ PTIME.

p-COLORABILITY
Instance: A graph $\mathcal{G}$, and $\ell \in \mathbb{N}$.
Parameter: ℓ.
Problem: Decide whether $\mathcal{G}$ is ℓ-colorable.

Consequence: p-COLORABILITY $\notin$ FPT (unless P = NP).

It is well-known: 3-COLORABILITY $\in$ NP-complete. Now since 3-COLORABILITY is the third slice of p-COLORABILITY, the proposition entails p-COLORABILITY $\notin$ FPT unless P = NP.

Polynomial reductions / hardness / completeness

Definition

Let $\langle Q_1, \Sigma_1 \rangle$, $\langle Q_2, \Sigma_2 \rangle$ be classical problems.
An *polynomial-time reduction* from $\langle Q_1, \Sigma_1 \rangle$ to $\langle Q_2, \Sigma_2 \rangle$ is a mapping $R : \Sigma_1^* \to \Sigma_2^*$:

R1. $(x \in Q_1 \iff R(x) \in Q_2)$ for all $x \in \Sigma_1^*$.

R2. R is computable by a polynomial-time algorithm: there is a polynomial $p(X)$ such that R is computable in time $p(|x|)$.

Polynomial reductions / hardness / completeness

Definition

Let $\langle Q_1, \Sigma_1 \rangle$, $\langle Q_2, \Sigma_2 \rangle$ be classical problems.
An *polynomial-time reduction* from $\langle Q_1, \Sigma_1 \rangle$ to $\langle Q_2, \Sigma_2 \rangle$ is a mapping $R : \Sigma_1^* \to \Sigma_2^*$:

R1. $(x \in Q_1 \iff R(x) \in Q_2)$ for all $x \in \Sigma_1^*$.

R2. R is computable by a polynomial-time algorithm: there is a polynomial $p(X)$ such that R is computable in time $p(|x|)$.

$\langle Q_1, \Sigma_1 \rangle \leq_{\text{pol}} \langle Q_2, \Sigma_2 \rangle :=$
there is a polynomial-time reduction from $\langle Q_1, \Sigma_1 \rangle$ to $\langle Q_2, \Sigma_2 \rangle$.

Polynomial reductions / hardness / completeness

Definition

Let $\langle Q_1, \Sigma_1 \rangle$, $\langle Q_2, \Sigma_2 \rangle$ be classical problems.
An *polynomial-time reduction* from $\langle Q_1, \Sigma_1 \rangle$ to $\langle Q_2, \Sigma_2 \rangle$ is a mapping $R : \Sigma_1^* \to \Sigma_2^*$:

R1. $(x \in Q_1 \iff R(x) \in Q_2)$ for all $x \in \Sigma_1^*$.

R2. R is computable by a polynomial-time algorithm: there is a polynomial $p(X)$ such that R is computable in time $p(|x|)$.

$\langle Q_1, \Sigma_1 \rangle \leq_{\text{pol}} \langle Q_2, \Sigma_2 \rangle$:=
there is a polynomial-time reduction from $\langle Q_1, \Sigma_1 \rangle$ to $\langle Q_2, \Sigma_2 \rangle$.

Proposition

If $\langle Q_1, \Sigma_1 \rangle \leq_{\text{pol}} \langle Q_2, \Sigma_2 \rangle$, then:
$\langle Q_1, \Sigma_1 \rangle \in \mathrm{P} \Longleftarrow \langle Q_2, \Sigma_2 \rangle \in \mathrm{P}$.
$\langle Q_1, \Sigma_1 \rangle \notin \mathrm{P} \Longrightarrow \langle Q_2, \Sigma_2 \rangle \notin \mathrm{P}$.

Polynomial reductions / hardness / completeness

Definition

Let $\langle Q_1, \Sigma_1\rangle$, $\langle Q_2, \Sigma_2\rangle$ be classical problems.
An *polynomial-time reduction* from $\langle Q_1, \Sigma_1\rangle$ to $\langle Q_2, \Sigma_2\rangle$ is a mapping $R : \Sigma_1^* \to \Sigma_2^*$:

R1. $(x \in Q_1 \iff R(x) \in Q_2)$ for all $x \in \Sigma_1^*$.

R2. R is computable by a polynomial-time algorithm: there is a polynomial $p(X)$ such that R is computable in time $p(|x|)$.

$\langle Q_1, \Sigma_1\rangle \leq_{\text{pol}} \langle Q_2, \Sigma_2\rangle :=$
there is a polynomial-time reduction from $\langle Q_1, \Sigma_1\rangle$ to $\langle Q_2, \Sigma_2\rangle$.

Proposition

If $\langle Q_1, \Sigma_1\rangle \leq_{\text{pol}} \langle Q_2, \Sigma_2\rangle$, then: $\langle Q_1, \Sigma_1\rangle \in \mathbf{P} \Longleftarrow \langle Q_2, \Sigma_2\rangle \in \mathbf{P}$.
$\langle Q_1, \Sigma_1\rangle \notin \mathbf{P} \Longrightarrow \langle Q_2, \Sigma_2\rangle \notin \mathbf{P}$.

Let C be class of classical problems.

- $\langle Q, \Sigma\rangle$ is C-hard: if, for all $\langle Q', \Sigma'\rangle \in$ C, $\langle Q', \Sigma'\rangle \leq_{\text{pol}} \langle Q, \Sigma\rangle$.

Polynomial reductions / hardness / completeness

Definition

Let $\langle Q_1, \Sigma_1 \rangle$, $\langle Q_2, \Sigma_2 \rangle$ be classical problems.
An *polynomial-time reduction* from $\langle Q_1, \Sigma_1 \rangle$ to $\langle Q_2, \Sigma_2 \rangle$ is a mapping $R : \Sigma_1^* \to \Sigma_2^*$:

R1. $(x \in Q_1 \iff R(x) \in Q_2)$ for all $x \in \Sigma_1^*$.

R2. R is computable by a polynomial-time algorithm: there is a polynomial $p(X)$ such that R is computable in time $p(|x|)$.

$\langle Q_1, \Sigma_1 \rangle \leq_{\text{pol}} \langle Q_2, \Sigma_2 \rangle$:=
there is a polynomial-time reduction from $\langle Q_1, \Sigma_1 \rangle$ to $\langle Q_2, \Sigma_2 \rangle$.

Proposition

If $\langle Q_1, \Sigma_1 \rangle \leq_{\text{pol}} \langle Q_2, \Sigma_2 \rangle$, then:
$\langle Q_1, \Sigma_1 \rangle \in$ P $\Longleftarrow$ $\langle Q_2, \Sigma_2 \rangle \in$ P.
$\langle Q_1, \Sigma_1 \rangle \notin$ P $\Longrightarrow$ $\langle Q_2, \Sigma_2 \rangle \notin$ P.

Let C be class of classical problems.

- $\langle Q, \Sigma \rangle$ is C-hard: if, for all $\langle Q', \Sigma' \rangle \in$ C, $\langle Q', \Sigma' \rangle \leq_{\text{pol}} \langle Q, \Sigma \rangle$.
- $\langle Q, \Sigma \rangle$ is C-complete: if $\langle Q, \Sigma \rangle$ is C-hard, and $\langle Q, \Sigma \rangle \in$ C.

Fixed-parameter tractable reductions

Definition

Let $\langle Q_1, \Sigma_1, \kappa\rangle$, $\langle Q_2, \Sigma_2, \kappa_2\rangle$ be parameterized problems.
An *fpt-reduction* from $\langle Q_1, \kappa_1\rangle$ to $\langle Q_2, \kappa_2\rangle$ is a mapping $R : \Sigma_1^* \to (\Sigma_2)^*$:

R1. $(x \in Q_1 \iff R(x) \in Q_2)$ for all $x \in \Sigma_1^*$.

R2. R is computable by a fpt-algorithm (with respect to κ): there are f computable and $p(X)$ polynomial such that R is computable in time $f(\kappa_1(x)) \cdot p(|x|)$.

Fixed-parameter tractable reductions

Definition

Let $\langle Q_1, \Sigma_1, \kappa \rangle$, $\langle Q_2, \Sigma_2, \kappa_2 \rangle$ be parameterized problems.
An *fpt-reduction* from $\langle Q_1, \kappa_1 \rangle$ to $\langle Q_2, \kappa_2 \rangle$ is a mapping $R : \Sigma_1^* \to (\Sigma_2)^*$:

R1. $(x \in Q_1 \iff R(x) \in Q_2)$ for all $x \in \Sigma_1^*$.

R2. R is computable by a fpt-algorithm (with respect to κ): there are f computable and $p(X)$ polynomial such that R is computable in time $f(\kappa_1(x)) \cdot p(|x|)$.

R3. $\kappa_2(R(x)) \le g(\kappa_1(x))$ for all $x \in \Sigma_1^*$, for some computable function $g : \mathbb{N} \to \mathbb{N}$.

Fixed-parameter tractable reductions

Definition

Let $\langle Q_1, \Sigma_1, \kappa \rangle$, $\langle Q_2, \Sigma_2, \kappa_2 \rangle$ be parameterized problems.
An *fpt-reduction* from $\langle Q_1, \kappa_1 \rangle$ to $\langle Q_2, \kappa_2 \rangle$ is a mapping $R : \Sigma_1^* \to (\Sigma_2)^*$:

R1. $(x \in Q_1 \iff R(x) \in Q_2)$ for all $x \in \Sigma_1^*$.

R2. R is computable by a fpt-algorithm (with respect to κ): there are f computable and $p(X)$ polynomial such that R is computable in time $f(\kappa_1(x)) \cdot p(|x|)$.

R3. $\kappa_2(R(x)) \leq g(\kappa_1(x))$ for all $x \in \Sigma_1^*$, for some computable function $g : \mathbb{N} \to \mathbb{N}$.

$\langle Q_1, \kappa_1 \rangle \leq_{\text{fpt}} \langle Q_2, \kappa_2 \rangle :=$ there is an fpt-red. from $\langle Q_1, \kappa_1 \rangle$ to $\langle Q_2, \kappa_2 \rangle$.

Fixed-parameter tractable reductions

Definition

Let $\langle Q_1, \Sigma_1, \kappa \rangle$, $\langle Q_2, \Sigma_2, \kappa_2 \rangle$ be parameterized problems.
An *fpt-reduction* from $\langle Q_1, \kappa_1 \rangle$ to $\langle Q_2, \kappa_2 \rangle$ is a mapping $R : \Sigma_1^* \to (\Sigma_2)^*$:

R1. $(x \in Q_1 \iff R(x) \in Q_2)$ for all $x \in \Sigma_1^*$.

R2. R is computable by a fpt-algorithm (with respect to κ): there are f computable and $p(X)$ polynomial such that R is computable in time $f(\kappa_1(x)) \cdot p(|x|)$.

R3. $\kappa_2(R(x)) \le g(\kappa_1(x))$ for all $x \in \Sigma_1^*$, for some computable function $g : \mathbb{N} \to \mathbb{N}$.

$\langle Q_1, \kappa_1 \rangle \le_{\text{fpt}} \langle Q_2, \kappa_2 \rangle :=$ there is an fpt-red. from $\langle Q_1, \kappa_1 \rangle$ to $\langle Q_2, \kappa_2 \rangle$.

Proposition

If $\langle Q_1, \kappa_1 \rangle \le_{\text{fpt}} \langle Q_2, \kappa_2 \rangle$, then:

$$\langle Q_1, \kappa_1 \rangle \in \text{FPT} \iff \langle Q_2, \kappa_2 \rangle \in \text{FPT}.$$

$$\langle Q_1, \kappa_1 \rangle \notin \text{FPT} \implies \langle Q_2, \kappa_2 \rangle \notin \text{FPT}.$$

Comparing parameterizations (revisited)

Definition (computably bounded below)

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

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

Proposition

For all parameterized problems $\langle Q, \kappa_1 \rangle$ and $\langle Q, \kappa_2 \rangle$ with $\kappa_1 \succeq \kappa_2$:

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

Comparing parameterizations (revisited)

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 $\forall x \in \Sigma^* \big[\, g(\kappa_1(x)) \geq \kappa_2(x) \,\big]$.
- $\kappa_1 \approx \kappa_2 \;:\Longleftrightarrow\; \kappa_1 \geq \kappa_2 \;\wedge\; \kappa_2 \geq \kappa_1$.
- $\kappa_1 \succ \kappa_2 \;:\Longleftrightarrow\; \kappa_1 \geq \kappa_2 \;\wedge\; \neg(\kappa_2 \geq \kappa_1)$.

Proposition

For all parameterized problems $\langle Q, \kappa_1 \rangle$ and $\langle Q, \kappa_2 \rangle$ with $\kappa_1 \geq \kappa_2$:

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

Proposition

For all parameterized problems $\langle Q, \kappa_1 \rangle$ and $\langle Q, \kappa_2 \rangle$ with $Q \subseteq \Sigma^*$:

$$\kappa_1 \geq \kappa_2 \quad \Longleftrightarrow \quad \langle Q, \kappa_1 \rangle \leq_{\text{fpt}} \langle Q, \kappa_2 \rangle \text{ via } R : \Sigma^* \to \Sigma^*,\ x \mapsto x.$$

Fixed-parameter tractable reductions

Examples

- p-CLIQUE $\equiv_{\text{fpt}}$ p-INDEPENDENT-SET.
- p-DOMINATING-SET $\equiv_{\text{fpt}}$ p-HITTING-SET.

Fixed-parameter tractable reductions

Examples

- p-CLIQUE $\equiv_{\text{fpt}}$ p-INDEPENDENT-SET.
- p-DOMINATING-SET $\equiv_{\text{fpt}}$ p-HITTING-SET.

Non-Example

- For graphs $\mathcal{G} = \langle V, E \rangle$, and sets $X \subseteq V$:

  $$X \text{ is independent set of } \mathcal{G} \iff V \setminus X \text{ is a vertex cover of } \mathcal{G}$$

  yields a polynomial reduction between p-INDEPENDENT-SET and p-VERTEX-COVER, but does not yield an fpt-reduction.

Fpt-reduction closure / hardness / reducibility

Let C be a class of parameterized problems.

We define for all parameterized problems $\langle Q, \kappa \rangle$:

- $\langle Q, \kappa \rangle$ is C*-hard under fpt-reductions*
  if every problem in C is fpt-reducible to $\langle Q, \kappa \rangle$

Fpt-reduction closure / hardness / reducibility

Let C be a class of parameterized problems.

We define for all parameterized problems $\langle Q, \kappa \rangle$:

- $\langle Q, \kappa \rangle$ is C*-hard under fpt-reductions*
  if every problem in C is fpt-reducible to $\langle Q, \kappa \rangle$
- $\langle Q, \kappa \rangle$ is C*-complete under fpt-reductions*
  if $\langle Q, \kappa \rangle \in$ C and $\langle Q, \kappa \rangle$ is C-hard under fpt-reductions,

Fpt-reduction closure / hardness / reducibility

Let C be a class of parameterized problems.

We define for all parameterized problems $\langle Q, \kappa \rangle$:

- $\big[\langle Q, \kappa \rangle\big]^{\text{fpt}} := \big\{\langle Q', \kappa' \rangle \mid \langle Q', \kappa' \rangle \leq_{\text{fpt}} \langle Q, \kappa \rangle\big\}$.
- $\big[\mathrm{C}\big]^{\text{fpt}} := \bigcup_{\langle Q, \kappa \rangle \in \mathrm{C}} \big[\langle Q, \kappa \rangle\big]^{\text{fpt}}$
  is the *closure of* C *under fpt-reductions.*
- $\langle Q, \kappa \rangle$ is C*-hard under fpt-reductions*
  if every problem in C is fpt-reducible to $\langle Q, \kappa \rangle$
- $\langle Q, \kappa \rangle$ is C*-complete under fpt-reductions*
  if $\langle Q, \kappa \rangle \in \mathrm{C}$ and $\langle Q, \kappa \rangle$ is C-hard under fpt-reductions,

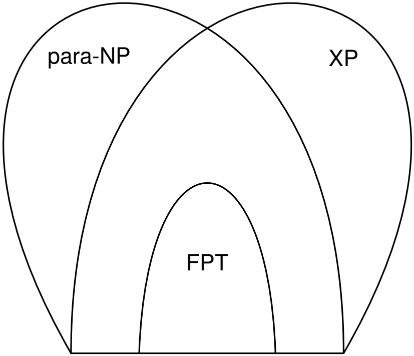
Fpt-reduction closure / hardness / reducibility

Let C be a class of parameterized problems.

We define for all parameterized problems $\langle Q, \kappa \rangle$:

- $\big[\langle Q, \kappa \rangle\big]^{\text{fpt}} := \big\{ \langle Q', \kappa' \rangle \mid \langle Q', \kappa' \rangle \leq_{\text{fpt}} \langle Q, \kappa \rangle \big\}$.
- $\big[\mathrm{C}\big]^{\text{fpt}} := \bigcup_{\langle Q, \kappa \rangle \in \mathrm{C}} \big[\langle Q, \kappa \rangle\big]^{\text{fpt}}$
  is the *closure of* C *under fpt-reductions.*
- $\langle Q, \kappa \rangle$ is C*-hard under fpt-reductions*
  if every problem in C is fpt-reducible to $\langle Q, \kappa \rangle$
  that is: $\mathrm{C} \subseteq \big[\langle Q, \kappa \rangle\big]^{\text{fpt}}$, and hence $\big[\mathrm{C}\big]^{\text{fpt}} \subseteq \big[\langle Q, \kappa \rangle\big]^{\text{fpt}}$.
- $\langle Q, \kappa \rangle$ is C*-complete under fpt-reductions*
  if $\langle Q, \kappa \rangle \in \mathrm{C}$ and $\langle Q, \kappa \rangle$ is C-hard under fpt-reductions,
  and then: $\big[\mathrm{C}\big]^{\text{fpt}} = \big[\langle Q, \kappa \rangle\big]^{\text{fpt}}$.

para-NP and XP



para-NP

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in para-NP if there is a computable function $f : \mathbb{N} \to \mathbb{N}$, and a polynomial $p \in \mathbb{N}[X]$ such that there is a non-deterministic algorithm $\mathbb{A}$ such that:

- $\mathbb{A}$ decides, for all $x \in \Sigma^*$, whether $x \in Q$ in $\leq f(\kappa(x)) \cdot p(|x|)$ steps.

para-NP

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in para-NP if there is a computable function $f : \mathbb{N} \to \mathbb{N}$, and a polynomial $p \in \mathbb{N}[X]$ such that there is a non-deterministic algorithm $\mathbb{A}$ such that:

- $\mathbb{A}$ decides, for all $x \in \Sigma^*$, whether $x \in Q$ in $\leq f(\kappa(x)) \cdot p(|x|)$ steps.

- para-NP is closed under fpt-reductions.

para-NP

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in para-NP if there is a computable function $f : \mathbb{N} \to \mathbb{N}$, and a polynomial $p \in \mathbb{N}[X]$ such that there is a non-deterministic algorithm $\mathbb{A}$ such that:

- $\mathbb{A}$ decides, for all $x \in \Sigma^*$, whether $x \in Q$ in $\leq f(\kappa(x)) \cdot p(|x|)$ steps.

- para-NP is closed under fpt-reductions.
- NP $\subseteq$ para-NP.

para-NP

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in para-NP if there is a computable function $f : \mathbb{N} \to \mathbb{N}$, and a polynomial $p \in \mathbb{N}[X]$ such that there is a non-deterministic algorithm $\mathbb{A}$ such that:

- $\mathbb{A}$ decides, for all $x \in \Sigma^*$, whether $x \in Q$ in $\leq f(\kappa(x)) \cdot p(|x|)$ steps.

- para-NP is closed under fpt-reductions.
- NP $\subseteq$ para-NP.

Example

- p-CLIQUE, p-INDEPENDENT-SET, p-DOMINATING-SET, p-HITTING-SET, p-COLORABILITY $\in$ para-NP.

para-NP

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in para-NP if there is a computable function $f : \mathbb{N} \to \mathbb{N}$, and a polynomial $p \in \mathbb{N}[X]$ such that there is a non-deterministic algorithm $\mathbb{A}$ such that:

- $\mathbb{A}$ decides, for all $x \in \Sigma^*$, whether $x \in Q$ in $\leq f(\kappa(x)) \cdot p(|x|)$ steps.

- para-NP is closed under fpt-reductions.
- NP $\subseteq$ para-NP.

Example

- p-CLIQUE, p-INDEPENDENT-SET, p-DOMINATING-SET, p-HITTING-SET, p-COLORABILITY $\in$ para-NP.

- FPT = para-NP if and only if PTIME = NP.

para-NP

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in para-NP if there is a computable function $f : \mathbb{N} \to \mathbb{N}$, and a polynomial $p \in \mathbb{N}[X]$ such that there is a non-deterministic algorithm $\mathbb{A}$ such that:

- $\mathbb{A}$ decides, for all $x \in \Sigma^*$, whether $x \in Q$ in $\leq f(\kappa(x)) \cdot p(|x|)$ steps.

- para-NP is closed under fpt-reductions.
- NP $\subseteq$ para-NP.

Example

- p-CLIQUE, p-INDEPENDENT-SET, p-DOMINATING-SET, p-HITTING-SET, p-COLORABILITY $\in$ para-NP.

- FPT = para-NP if and only if PTIME = NP.
- A non-trivial problem $\langle Q, \kappa \rangle$ is para-NP-complete for fpt-reductions if and only if the union of finitely many slices of $\langle Q, \kappa \rangle$ is NP-complete.

para-NP

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in para-NP if there is a computable function $f : \mathbb{N} \to \mathbb{N}$, and a polynomial $p \in \mathbb{N}[X]$ such that there is a non-deterministic algorithm $\mathbb{A}$ such that:

- $\mathbb{A}$ decides, for all $x \in \Sigma^*$, whether $x \in Q$ in $\leq f(\kappa(x)) \cdot p(|x|)$ steps.

- para-NP is closed under fpt-reductions.
- NP $\subseteq$ para-NP.

Example

- p-CLIQUE, p-INDEPENDENT-SET, p-DOMINATING-SET, p-HITTING-SET, p-COLORABILITY $\in$ para-NP.

- FPT = para-NP if and only if PTIME = NP.
- A non-trivial problem $\langle Q, \kappa \rangle$ is para-NP-complete for fpt-reductions if and only if the union of finitely many slices of $\langle Q, \kappa \rangle$ is NP-complete. Hence a non-trivial problem with at least one NP-complete slice is para-NP-complete.

para-NP

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in para-NP if there is a computable function $f : \mathbb{N} \to \mathbb{N}$, and a polynomial $p \in \mathbb{N}[X]$ such that there is a non-deterministic algorithm $\mathbb{A}$ such that:

- $\mathbb{A}$ decides, for all $x \in \Sigma^*$, whether $x \in Q$ in $\leq f(\kappa(x)) \cdot p(|x|)$ steps.

- para-NP is closed under fpt-reductions.
- NP $\subseteq$ para-NP.

Example

- p-CLIQUE, p-INDEPENDENT-SET, p-DOMINATING-SET, p-HITTING-SET, p-COLORABILITY $\in$ para-NP.

- FPT = para-NP if and only if PTIME = NP.
- A non-trivial problem $\langle Q, \kappa \rangle$ is para-NP-complete for fpt-reductions if and only if the union of finitely many slices of $\langle Q, \kappa \rangle$ is NP-complete. Hence a non-trivial problem with at least one NP-complete slice is para-NP-complete.
  - p-COLORABILITY $\in$ para-NP-complete.

XP (slicewise polynomial problems)

Recall: slices of FPT-problems are in PTIME. This suggests a class:

$\mathrm{XP}_{\mathrm{nu}}$, *non-uniform* XP : the class of parameterized problems $\langle Q, \kappa \rangle$, whose slices $\langle Q, \kappa \rangle_k$ are all in PTIME.

XP (slicewise polynomial problems)

Recall: slices of FPT-problems are in PTIME. This suggests a class:

XP (slicewise polynomial problems)

Recall: slices of FPT-problems are in PTIME. This suggests a class:

$\mathrm{XP}_{\mathrm{nu}}$, *non-uniform* XP : the class of parameterized problems $\langle Q, \kappa \rangle$, whose slices $\langle Q, \kappa \rangle_k$ are all in PTIME.

XP (slicewise polynomial problems)

Recall: slices of FPT-problems are in PTIME. This suggests a class:

$\mathrm{XP}_{\mathrm{nu}}$, *non-uniform* XP : the class of parameterized problems $\langle Q, \kappa \rangle$, whose slices $\langle Q, \kappa \rangle_k$ are all in PTIME.

- But: $\mathrm{XP}_{\mathrm{nu}}$ contains undecidable problems:
  - Let $Q \subseteq \{1\}^*$ be an undecidable set. Let $\kappa : \{1\}^* \to \mathbb{N}$, $x \mapsto \kappa(x) := \max\{1, |x|\}$. Then $\langle Q, \kappa \rangle \in \mathrm{XP}_{\mathrm{nu}}$.

XP (slicewise polynomial problems)

Recall: slices of FPT-problems are in PTIME. This suggests a class:

$\mathrm{XP}_{\mathrm{nu}}$, *non-uniform* XP : the class of parameterized problems $\langle Q, \kappa \rangle$, whose slices $\langle Q, \kappa \rangle_k$ are all in PTIME.

- But: $\mathrm{XP}_{\mathrm{nu}}$ contains undecidable problems:
  - Let $Q \subseteq \{1\}^*$ be an undecidable set. Let $\kappa : \{1\}^* \to \mathbb{N}$, $x \mapsto \kappa(x) := \max\{1, |x|\}$. Then $\langle Q, \kappa \rangle \in \mathrm{XP}_{\mathrm{nu}}$.

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in XP if there is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that there is an algorithm $\mathbb{A}$ such that:

- $\mathbb{A}$ decides $x \in Q$, for all $x \in \Sigma^*$, in $\leq f(\kappa(x)) + |x|^{f(\kappa(x))}$ steps;

XP (slicewise polynomial problems)

Recall: slices of FPT-problems are in PTIME. This suggests a class:

$\mathrm{XP}_{\mathrm{nu}}$, *non-uniform* XP : the class of parameterized problems $\langle Q, \kappa \rangle$, whose slices $\langle Q, \kappa \rangle_k$ are all in PTIME.

- But: $\mathrm{XP}_{\mathrm{nu}}$ contains undecidable problems:
  - Let $Q \subseteq \{1\}^*$ be an undecidable set. Let $\kappa : \{1\}^* \to \mathbb{N}$, $x \mapsto \kappa(x) := \max\{1, |x|\}$. Then $\langle Q, \kappa \rangle \in \mathrm{XP}_{\mathrm{nu}}$.

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in XP if there is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that there is an algorithm $\mathbb{A}$ such that:

- $\mathbb{A}$ decides $x \in Q$, for all $x \in \Sigma^*$, in $\leq f(\kappa(x)) + |x|^{f(\kappa(x))}$ steps;

equivalently, if in addition to computable $f : \mathbb{N} \to \mathbb{N}$ there are polynomials $p_k \in \mathbb{N}[X]$ for all $k \in \mathbb{N}$ such that:

- $\mathbb{A}$ decides $x \in Q$, for all $x \in \Sigma^*$, in $\leq f(\kappa(x)) \cdot p_{\kappa(x)}(|x|)$ steps.

XP (slicewise polynomial problems)

Example

- p-CLIQUE, p-INDEPENDENT-SET, p-DOMINATING-SET, p-HITTING-SET $\in$ XP.
- p-COLORABILITY $\notin$ XP, because 3-COLORABILITY $\in$ NP-complete.

Proposition

If PTIME $\neq$ NP, then para-NP $\nsubseteq$ XP.

XP (slicewise polynomial problems)

Example

- p-CLIQUE, p-INDEPENDENT-SET, p-DOMINATING-SET, p-HITTING-SET $\in$ XP.
- p-COLORABILITY $\notin$ XP, because 3-COLORABILITY $\in$ NP-complete.

Proposition

If PTIME $\neq$ NP, then para-NP $\nsubseteq$ XP.

Proof.

If para-NP $\subseteq$ XP, then p-COLORABILITY $\in$ XP. But then it follows that 3-COLORABILITY $\in$ PTIME, and as 3-COLORABILITY is NP-complete, that PTIME = NP. □

XP (slicewise polynomial problems)

Example

- p-CLIQUE, p-INDEPENDENT-SET, p-DOMINATING-SET, p-HITTING-SET $\in$ XP.
- p-COLORABILITY $\notin$ XP, because 3-COLORABILITY $\in$ NP-complete.

Proposition

If PTIME $\neq$ NP, then para-NP $\nsubseteq$ XP.

Proof.

If para-NP $\subseteq$ XP, then p-COLORABILITY $\in$ XP. But then it follows that 3-COLORABILITY $\in$ PTIME, and as 3-COLORABILITY is NP-complete, that PTIME $=$ NP. □

Proposition

FPT $\subsetneqq$ XP .

Model checking

The *model checking problem* for a class Φ of first-order formulas:

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

Theorem

MC(FO) *can be solved in time* $O(|\varphi| \cdot |A|^w \cdot w)$, *where* w *is the width of the input formula* φ *(max. no. of free variables in a subformula of* φ*).*

Model checking

The *model checking problem* for a class Φ of first-order formulas:

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

Theorem

MC(FO) *can be solved in time* $O(|\varphi| \cdot |A|^w \cdot w)$, *where* w *is the width of the input formula* φ *(max. no. of free variables in a subformula of* φ*).*

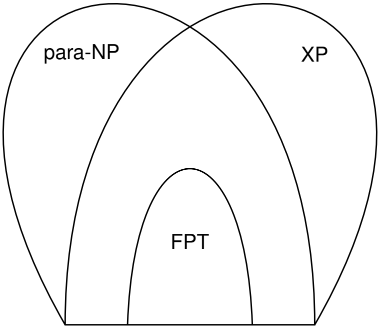
The *parameterized model checking problem* for a class Φ of formulas:

p-MC(Φ).
Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
Parameter: $|\varphi|$.
Problem: Decide whether $\mathcal{A} \models \varphi$.

Theorem

p-MC(Φ) $\in$ XP.

FPT versus para-NP and XP



Proposition

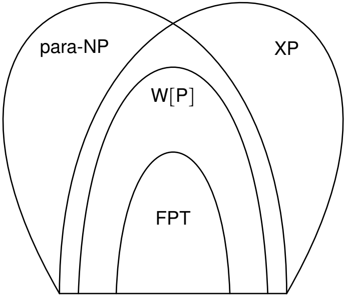
- FPT $\subseteq$ para-NP, and:
  FPT $=$ para-NP if and only if PTIME $=$ NP.
- para-NP $\nsubseteq$ XP if PTIME $\neq$ NP.
- FPT $\subsetneq$ XP.

W[P]

'There is no definite single class that can be viewed as "the parameterized NP". Rather, there is a whole hierarchy of classes playing this role.

The class W[P] can be placed on top of this hierarchy. It is one of the most important parameterized complexity classes.'

(Flum, Grohe [2])



W[P] and limited non-determinism

$\langle Q, \Sigma \rangle \in$ NP$[f]$ means:

$\iff \exists p(X)$ polynomial $\exists \mathbb{M}$ non-deterministic Turingmachine

$\big(\forall x \in \Sigma^* \big((x \in Q \iff \mathbb{M}$ accepts $x)$

$\wedge$ on input x, $\mathbb{M}$ halts in $\leq p(|x|)$ steps, of which at most $\leq f(|x|)$ are non-deterministic$\big)$

W[P] and limited non-determinism

$\langle Q, \Sigma \rangle \in$ NP$[f]$ means:

$\iff \exists p(X)$ polynomial $\exists \mathbb{M}$ non-deterministic Turingmachine
$\big(\forall x \in \Sigma^* \big(\, (x \in Q \iff \mathbb{M}$ accepts $x)$
$\wedge$ on input x, $\mathbb{M}$ halts in $\le p(|x|)$ steps, of which
at most $\le f(|x|)$ are non-deterministic$\big)$

NP$[\mathcal{F}] := \bigcup_{f \in \mathcal{F}}$ NP$[f]$ for class of functions $\mathcal{F}$.

W[P] and limited non-determinism

$\langle Q, \Sigma \rangle \in$ NP$[f]$ means:

$\iff \exists p(X)$ polynomial $\exists \mathbb{M}$ non-deterministic Turingmachine
$\big(\forall x \in \Sigma^* \big((x \in Q \iff \mathbb{M}$ accepts $x)$
$\land$ on input x, $\mathbb{M}$ halts in $\le p(|x|)$ steps, of which
at most $\le f(|x|)$ are non-deterministic$\big)$

NP$[\mathcal{F}] := \bigcup_{f \in \mathcal{F}}$ NP$[f]$ for class of functions $\mathcal{F}$.

Fact

$$\text{NP}[\log n] = \text{P}\,, \qquad \text{NP}[n^{O(1)}] = \text{NP}\,.$$

W[P]

Definition

- Let Σ be an alphabet, and $\kappa : \Sigma^* \to \mathbb{N}$ a parameterization.
  A nondeterministic Turing machine $\mathbb{M}$ with input alphabet Σ is *κ-restricted*

W[P]

Definition

- Let Σ be an alphabet, and $\kappa : \Sigma^* \to \mathbb{N}$ a parameterization.

  A nondeterministic Turing machine $\mathbb{M}$ with input alphabet Σ is *κ-restricted* if there are computable functions $f, h : \mathbb{N} \to \mathbb{N}$ and a polynomial $p \in \mathbb{N}_0[x]$ such that on every run with input $x \in \Sigma^*$ the machine $\mathbb{M}$ performs

  - at most $f(\kappa(x)) \cdot p(|x|)$ steps,

W[P]

Definition

- Let Σ be an alphabet, and $\kappa : \Sigma^* \to \mathbb{N}$ a parameterization.

  A nondeterministic Turing machine $\mathbb{M}$ with input alphabet Σ is *κ-restricted* if there are computable functions $f, h : \mathbb{N} \to \mathbb{N}$ and a polynomial $p \in \mathbb{N}_0[x]$ such that on every run with input $x \in \Sigma^*$ the machine $\mathbb{M}$ performs
  - at most $f(\kappa(x)) \cdot p(|x|)$ steps,
  - at most $h(\kappa(x)) \cdot \log |x|$ of them being nondeterministic,

W[P]

Definition

- Let Σ be an alphabet, and $\kappa : \Sigma^* \to \mathbb{N}$ a parameterization.

  A nondeterministic Turing machine $\mathbb{M}$ with input alphabet Σ is *κ-restricted* if there are computable functions $f, h : \mathbb{N} \to \mathbb{N}$ and a polynomial $p \in \mathbb{N}_0[x]$ such that on every run with input $x \in \Sigma^*$ the machine $\mathbb{M}$ performs
  - at most $f(\kappa(x)) \cdot p(|x|)$ steps,
  - at most $h(\kappa(x)) \cdot \log |x|$ of them being nondeterministic,
- W[P] contains all problems $\langle Q, \kappa \rangle$ that can be decided by a κ-restricted nondeterministic Turing machine.

W[P] (properties)

Theorems

T1. FPT ⊆ W[P] ⊆ XP ∩ para-NP

T2. W[P] is closed under fpt-reductions.

T3. p-CLIQUE, p-INDEPENDENT-SET, p-DOMINATING-SET, and p-HITTING-SET are in W[P].

The W-hierarchy – Boolean circuits

A *(Boolean) circuit* is a DAG in which nodes are labeled:

- nodes of in-degree > 1 as *and-node* or as *or-node*,
- nodes of in-degree $= 1$ as *negation nodes*,

The W-hierarchy – Boolean circuits

A *(Boolean) circuit* is a DAG in which nodes are labeled:

- nodes of in-degree > 1 as *and-node* or as *or-node*,
- nodes of in-degree $= 1$ as *negation nodes*,
- nodes of in-degree $= 0$ as *Boolean constants* 0 or 1, or *input node* (we assume input nodes to be numbered 1,. . . ,n),

The W-hierarchy – Boolean circuits

A *(Boolean) circuit* is a DAG in which nodes are labeled:

- nodes of in-degree > 1 as *and-node* or as *or-node*,
- nodes of in-degree $= 1$ as *negation nodes*,
- nodes of in-degree $= 0$ as *Boolean constants* 0 or 1, or *input node* (we assume input nodes to be numbered 1,. . . ,n),
- one node of out-degree 0 is labeled as *output node*.

The W-hierarchy – Boolean circuits

A *(Boolean) circuit* is a DAG in which nodes are labeled:

- nodes of in-degree > 1 as *and-node* or as *or-node*,
- nodes of in-degree $= 1$ as *negation nodes*,
- nodes of in-degree $= 0$ as *Boolean constants* 0 or 1, or *input node* (we assume input nodes to be numbered 1,. . . ,n),
- one node of out-degree 0 is labeled as *output node*.

A circuit $\mathcal{C}$ with n input nodes defines a function $\mathcal{C}(\cdot) : \{0, 1\}^n \to \{0, 1\}$ (a Boolean function) in the natural way.

Definition

We say that $\mathcal{C}$ is *k-satisfiable* if $\mathcal{C}$ is satisfied by a tuple of weight k.

The W-hierarchy – Boolean circuits

A *(Boolean) circuit* is a DAG in which nodes are labeled:

- nodes of in-degree > 1 as *and-node* or as *or-node*,
- nodes of in-degree $= 1$ as *negation nodes*,
- nodes of in-degree $= 0$ as *Boolean constants* 0 or 1, or *input node* (we assume input nodes to be numbered 1,. . . ,n),
- one node of out-degree 0 is labeled as *output node*.

A circuit $\mathcal{C}$ with n input nodes defines a function $\mathcal{C}(\cdot) : \{0,1\}^n \to \{0,1\}$ (a Boolean function) in the natural way.

- If $\mathcal{C}(x) = 1$, for $x \in \{0,1\}^n$, we say that x *satisfies* $\mathcal{C}$.

Definition

We say that $\mathcal{C}$ is *k-satisfiable* if $\mathcal{C}$ is satisfied by a tuple of weight k.

The W-hierarchy – Boolean circuits

A *(Boolean) circuit* is a DAG in which nodes are labeled:

- nodes of in-degree > 1 as *and-node* or as *or-node*,
- nodes of in-degree $= 1$ as *negation nodes*,
- nodes of in-degree $= 0$ as *Boolean constants* 0 or 1, or *input node* (we assume input nodes to be numbered 1,. . . ,n),
- one node of out-degree 0 is labeled as *output node*.

A circuit $\mathcal{C}$ with n input nodes defines a function $\mathcal{C}(\cdot) : \{0,1\}^n \to \{0,1\}$ (a Boolean function) in the natural way.

- If $\mathcal{C}(x) = 1$, for $x \in \{0,1\}^n$, we say that x *satisfies* $\mathcal{C}$.
- The *weight* of a tuple $x = \langle x_1, \ldots, x_n \rangle \in \{0,1\}^*$ is $\sum_{i=1}^{n} x_i$.

Definition

We say that $\mathcal{C}$ is *k-satisfiable* if $\mathcal{C}$ is satisfied by a tuple of weight k.

W[P] complete problems

p-WSAT(CIRC)
Instance: A circuit $\mathcal{C}$ and $k \in \mathbb{N}$
Parameter: k.
Problem: Decide whether $\mathcal{C}$ is k-satisfiable.

Theorem

p-WSAT(CIRC) *is* W[P]*-complete under fpt-reductions.*

Definition

The depth of the circuit is the max. length of a path from an input node to the output node. Small nodes have indegree at most 2 while large nodes have indegree > 2. The weft of a circuit is the max. number of large nodes on a path from an input node to the output node. We denote by $\text{CIRC}_{t,d}$ the class of circuits with weft $\leq t$ and depth $\leq d$.

Application

p-DOMINATING-SET $\in$ W[P], since it reduces to p-WSAT($\text{CIRC}_{2,3}$).

Limited non-determinism (classically)

$\langle Q, \Sigma \rangle \in \mathrm{NP}[f]$ means:

$\iff \exists p(X)$ polynomial $\exists \mathbb{M}$ non-deterministic Turingmachine

$\big(\forall x \in \Sigma^* \big(\, (x \in Q \iff \mathbb{M}$ accepts $x)$

$\wedge$ on input x, $\mathbb{M}$ halts in $\leq p(|x|)$ steps, of which

at most $\leq f(|x|)$ are non-deterministic$\big)$

Limited non-determinism (classically)

$\langle Q, \Sigma \rangle \in$ NP$[f]$ means:

$\iff \exists p(X)$ polynomial $\exists \mathbb{M}$ non-deterministic Turingmachine

$\big(\forall x \in \Sigma^* \big((x \in Q \iff \mathbb{M}$ accepts $x)$

$\wedge$ on input x, $\mathbb{M}$ halts in $\leq p(|x|)$ steps, of which

at most $\leq f(|x|)$ are non-deterministic$\big)$

NP$[\mathcal{F}] := \bigcup_{f \in \mathcal{F}}$ NP$[f]$ for class of functions $\mathcal{F}$.

Limited non-determinism (classically)

$\langle Q, \Sigma \rangle \in$ NP$[f]$ means:

$\iff \exists p(X)$ polynomial $\exists \mathbb{M}$ non-deterministic Turingmachine
$\big(\forall x \in \Sigma^* \big(\, (x \in Q \iff \mathbb{M}$ accepts $x)$
$\wedge$ on input x, $\mathbb{M}$ halts in $\leq p(|x|)$ steps, of which
at most $\leq f(|x|)$ are non-deterministic$\big)$

NP$[\mathcal{F}] := \bigcup_{f \in \mathcal{F}}$ NP$[f]$ for class of functions $\mathcal{F}$.

Fact

$$\text{NP}[\log n] = \text{P}\,, \qquad \text{NP}[n^{O(1)}] = \text{NP}\,.$$

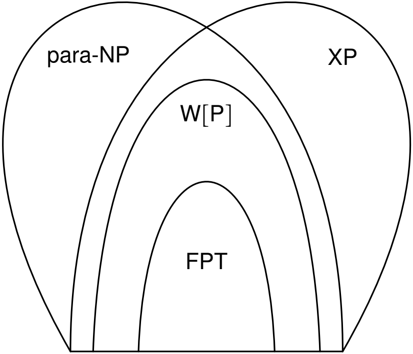
Theorem (Cai, Chen, 1997)

The following are equivalent:

(i) FPT = W[P].

(ii) *There is a computable, nondecreasing, unbounded function* $\iota : \mathbb{N} \to \mathbb{N}$ *such that* P = NP$[\iota(n) \cdot \log n]$.

FPT and W[P] versus para-NP and XP



Proposition

$\mathrm{FPT} \subseteq \mathrm{W}[\mathrm{P}] \subseteq \mathrm{XP} \cap \text{para-NP}$.

Why is the theory of W[P]/W/A-hardness important?

- Prevents from wasting hours tackling a problem which is fundamentally difficult;
- Finding results on a problem is always a ping-pong game between trying to design a hardness/FPT result;
- There is a hierarchy on parameters and it is worth knowing which is the smallest one such that the problem remains FPT;
- There is a hierarchy on complexity classes and it is worth noting to which extent a problem is hard.

Logic preliminaries (continued)

- *atomic formulas/atoms*: a formula $x = y$ or $Rx_1 \ldots x_n$
- *literal*: an atom or a negated atom
- *quantifier-free formula*: a formula without quantifiers
- formula in *negation-normal form*:
  negations only occur in front of atoms
- formula in *prenex normal form*: formula of the form
  $Q_1 x_1 \ldots Q_k x_k \, \psi$, where ψ is quantifier-free
  and $Q_1, \ldots, Q_k \in \{\exists, \forall\}$

Logic preliminaries (continued)

- *atomic formulas/atoms*: a formula $x = y$ or $Rx_1 \dots x_n$
- *literal*: an atom or a negated atom
- *quantifier-free formula*: a formula without quantifiers
- formula in *negation-normal form*:
  negations only occur in front of atoms
- formula in *prenex normal form*: formula of the form
  $Q_1 x_1 \dots Q_k x_k \, \psi$, where ψ is quantifier-free
  and $Q_1, \dots, Q_k \in \{\exists, \forall\}$
- Σ_0 and Π_0 : the class of quantifier-free formulas
- Σ_{t+1} : class of all formulas $\exists x_1 \dots \exists x_k \varphi$ where $\varphi \in \Pi_t$
- Π_{t+1} : class of all formulas $\forall x_1 \dots \forall x_k \varphi$ where $\varphi \in \Sigma_t$

Weighted Fagin definability

Let $\varphi(X)$ be a f-o formula with a free relation variable X with arity s.
Let τ be a vocabulary for φ, plus a relation symbol R of arity s.

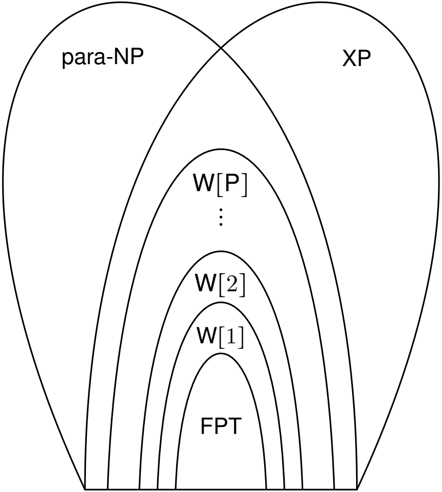
A *solution for φ in a τ-structure $\mathcal{A}$* is a relation $S \subseteq A^s$ such that $\mathcal{A} \vDash \varphi(\overline{S})$.

The *weighted Fagin definability problem* for $\varphi(X)$ is:

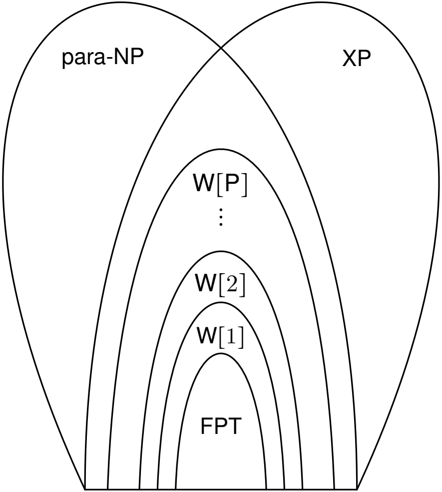
WD$_\varphi$
Instance: A structure $\mathcal{A}$ and $k \in \mathbb{N}$.
Problem: Decide whether there is a solution $S \subseteq A^s$ for φ of cardinality $|S| = k$.

WD$_\Phi$: the class of all problems WD$_\varphi$ with $\varphi \in \Phi$, where Φ is a class of first-order formulas with free relation variable X.

W-Hierarchy



W-Hierarchy



W-Hierarchy

p-WD$_\varphi$ (φ a fo-formula with free relation variable X of arity s)
Instance: A structure $\mathcal{A}$ and $k \in \mathbb{N}$.
Parameter: k.
Problem: Is there a relation $S \subseteq A^s$ of cardinality $|S| = k$ with $\mathcal{A} \models \varphi(S)$.

p-WD-Φ: the class of all problems p-WD-φ with $\varphi \in \Phi$, Φ is a class of first-order formulas.

Definition (Downey–Fellows, 1995)

$W[t] := \left[p\text{-WD-}\Pi_t\right]^{\text{fpt}}$, for $t \geq 1$, form the *W-hierarchy*.

W-Hierarchy

p-WD$_{\varphi}$ (φ a fo-formula with free relation variable X of arity s)
Instance: A structure $\mathcal{A}$ and $k \in \mathbb{N}$.
Parameter: k.
Problem: Is there a relation $S \subseteq A^s$ of cardinality $|S| = k$ with $\mathcal{A} \models \varphi(S)$.

p-WD-Φ: the class of all problems p-WD-φ with $\varphi \in \Phi$, Φ is a class of first-order formulas.

Definition (Downey–Fellows, 1995)

$W[t] := [p\text{-WD-}\Pi_t]^{\text{fpt}}$, for $t \geq 1$, form the *W-hierarchy*.

Examples

- p-CLIQUE $\in$ W$[1]$.
- p-DOMINATING-SET $\in$ W$[2]$.
- p-HITTING-SET $\in$ W$[2]$.

W-Hierarchy

p-WD$_\varphi$ (φ a fo-formula with free relation variable X of arity s)
Instance: A structure $\mathcal{A}$ and $k \in \mathbb{N}$.
Parameter: k.
Problem: Is there a relation $S \subseteq A^s$ of cardinality $|S| = k$ with $\mathcal{A} \models \varphi(S)$.

p-WD-Φ: the class of all problems p-WD-φ with $\varphi \in \Phi$, Φ is a class of first-order formulas.

Definition (Downey–Fellows, 1995)

$W[t] := \left[p\text{-WD-}\Pi_t\right]^{\text{fpt}}$, for $t \geq 1$, form the *W-hierarchy*.

Examples

- p-CLIQUE $\in$ W$[1]$.

W-Hierarchy

p-WD$_\varphi$ (φ a fo-formula with free relation variable X of arity s)
Instance: A structure $\mathcal{A}$ and $k \in \mathbb{N}$.
Parameter: k.
Problem: Is there a relation $S \subseteq A^s$ of cardinality $|S| = k$ with $\mathcal{A} \vDash \varphi(S)$.

p-WD-Φ: the class of all problems p-WD-φ with $\varphi \in \Phi$, Φ is a class of first-order formulas.

Definition (Downey–Fellows, 1995)

$W[t] := \left[p\text{-WD-}\Pi_t\right]^{\text{fpt}}$, for $t \geq 1$, form the *W-hierarchy*.

Examples

- p-DOMINATING-SET $\in$ W$[2]$.

W-Hierarchy

p-WD$_\varphi$ (φ a fo-formula with free relation variable X of arity s)
Instance: A structure $\mathcal{A}$ and $k \in \mathbb{N}$.
Parameter: k.
Problem: Is there a relation $S \subseteq A^s$ of cardinality $|S| = k$ with $\mathcal{A} \models \varphi(S)$.

p-WD-Φ: the class of all problems p-WD-φ with $\varphi \in \Phi$, Φ is a class of first-order formulas.

Definition (Downey–Fellows, 1995)

$W[t] := \left[p\text{-WD-}\Pi_t\right]^{\text{fpt}}$, for $t \geq 1$, form the *W-hierarchy*.

Examples

- p-HITTING-SET $\in$ W$[2]$.

W-hierarchy

Definition

(W-hierarchy) For $t \geq 1$, a parameterized problem $\langle Q, \kappa \rangle$ belongs to the class W$[t]$ if there is a parameterized reduction from $\langle Q, \kappa \rangle$ to p-WSAT(CIRC$_{t,d}$) (with parameter t) for some $d \geq 1$.

FPT $\subseteq$ W$[1]$ $\subseteq$ W$[2]$ $\ldots$

- p-Clique, p-Independent-Set are W$[1]$-Complete.
- p-Dominating-Set, p-Hitting-Set are W$[2]$-Complete.

Hypothesis: W$[1] \neq$ FPT

W-hierarchy

Definition

(W-hierarchy) For $t \geq 1$, a parameterized problem $\langle Q, \kappa \rangle$ belongs to the class W$[t]$ if there is a parameterized reduction from $\langle Q, \kappa \rangle$ to p-WSAT(CIRC$_{t,d}$) (with parameter t) for some $d \geq 1$.

FPT $\subseteq$ W$[1]$ $\subseteq$ W$[2]$ $\ldots$

- p-CLIQUE, p-INDEPENDENT-SET are W$[1]$-Complete.
- p-DOMINATING-SET, p-HITTING-SET are W$[2]$-Complete.

Hypothesis: W$[1] \neq$ FPT

Proposition

This definition of the W-hierarchy is equivalent to the one here before. That is, it holds, for all $t \geq 1$:

$$\text{W}[t] = \left[\{ p\text{-WSAT}(\text{CIRC}_{t,d}) \mid d \geq 1 \} \right]^{\text{fpt}} .$$

W-Hierarchy (properties)

Immediate from definition follows: $\left[p\text{-WD-FO}\right]^{\text{fpt}} = \bigcup_{i=1}^{\infty} \text{W}[i]$.

Theorems

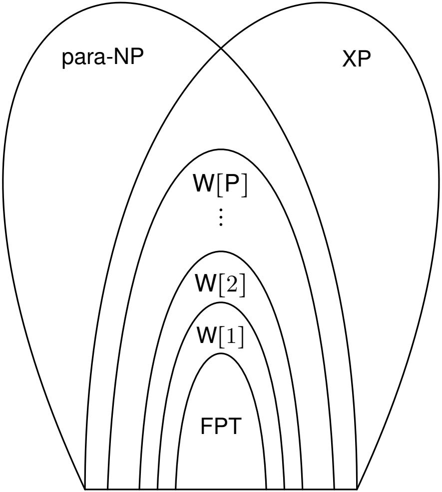
T1. p-WD-FO ⊆ W[P], and hence $\text{W}[t] \subseteq \text{W[P]}$ for all $t \geq 1$.

T2. $p\text{-WD-}\Sigma_1 \subseteq \text{FPT}$.

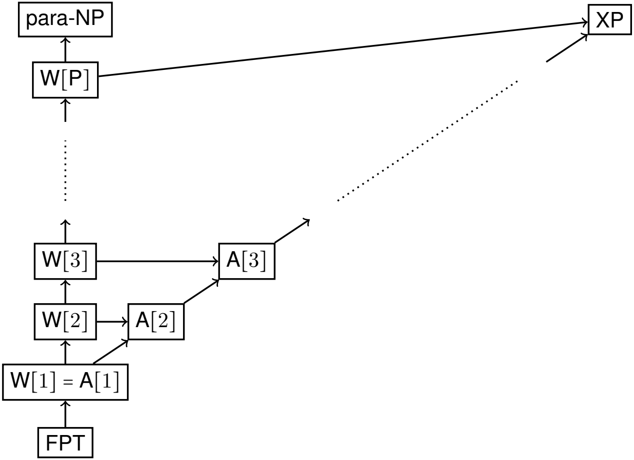
T3. $p\text{-WD-}\Sigma_{t+1} \subseteq p\text{-WD-}\Pi_t$, for all $t \geq 1$.

T4. $\text{W}[t] = \left[p\text{-WD-}\Sigma_{t+1}\right]^{\text{fpt}}$ for all $t \geq 1$.

W-Hierarchy versus para-NP and XP



A-Hierarchy



A-Hierarchy (definition and examples 1,2)

The parameterized model checking problem for a class Φ of formulas:

p-MC(Φ)
Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
Parameter: $|\varphi|$.
Problem: Decide whether $\varphi(\mathcal{A}) \neq \varnothing$.

Definition (Flum, Grohe, 2001)

$A[t] := \left[p\text{-MC}(\Sigma_t)\right]^{\text{fpt}}$, for $t \geq 1$, form the *A-hierarchy*.

A-Hierarchy (definition and examples 1,2)

The parameterized model checking problem for a class Φ of formulas:

p-MC(Φ)
Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
Parameter: $|\varphi|$.
Problem: Decide whether $\varphi(\mathcal{A}) \neq \varnothing$.

Definition (Flum, Grohe, 2001)

$A[t] := \left[p\text{-MC}(\Sigma_t)\right]^{\text{fpt}}$, for $t \geq 1$, form the *A-hierarchy*.

Examples

- p-CLIQUE $\in$ A$[1]$.
- p-DOMINATING-SET $\in$ A$[2]$.

A-Hierarchy (definition and examples 3,4)

The parameterized model checking problem for a class Φ of formulas:

p-MC(Φ)
Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
Parameter: $|\varphi|$.
Problem: Decide whether $\varphi(\mathcal{A}) \neq \varnothing$.

Definition (Flum, Grohe, 2001)

$A[t] := \left[p\text{-MC}(\Sigma_t)\right]^{\text{fpt}}$, for $t \geq 1$, form the *A-hierarchy*.

A-Hierarchy (definition and examples 3,4)

The parameterized model checking problem for a class Φ of formulas:

p-MC(Φ)
Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
Parameter: $|\varphi|$.
Problem: Decide whether $\varphi(\mathcal{A}) \neq \varnothing$.

Definition (Flum, Grohe, 2001)

$A[t] := [p\text{-MC}(\Sigma_t)]^{\text{fpt}}$, for $t \geq 1$, form the *A-hierarchy*.

Examples

- p-HITTING-SET $\in$ A$[2]$.
- p-SUBGRAPH-ISOMORPHISM $\in$ A$[1]$.

A-Hierarchy (example 5)

The parameterized model checking problem for a class Φ of formulas:

p-MC(Φ)
Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
Parameter: $|\varphi|$.
Problem: Decide whether $\varphi(\mathcal{A}) \neq \emptyset$.

Definition (Flum, Grohe, 2001)

$A[t] := \left[p\text{-MC}(\Sigma_t)\right]^{\text{fpt}}$, for $t \geq 1$, form the *A-hierarchy*.

Examples

- p-SUBGRAPH-ISOMORPHISM $\in$ A$[1]$.

p-SUBGRAPH-ISOMORPHISM
Instance: Graphs $\mathcal{G}$ and $\mathcal{H}$.
Parameter: The number of vertices of $\mathcal{H}$.
Problem: Does $\mathcal{G}$ have a subgraph isomorphic to $\mathcal{H}$.

A-Hierarchy (example 6)

The parameterized model checking problem for a class Φ of formulas:

p-MC(Φ)
Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
Parameter: $|\varphi|$.
Problem: Decide whether $\varphi(\mathcal{A}) \neq \varnothing$.

Definition (Flum, Grohe, 2001)

$A[t] := \left[p\text{-MC}(\Sigma_t)\right]^{\text{fpt}}$, for $t \geq 1$, form the *A-hierarchy*.

Examples

- p-Vertex-Deletion $\in$ A$[2]$.

p-Vertex-Deletion
Instance: Graphs $\mathcal{G}$ and $\mathcal{H}$, and $k \in \mathbb{N}$.
Parameter: $k + \ell$, where ℓ the number of vertices of $\mathcal{H}$.
Problem: Is it possible to delete at most k vertices from $\mathcal{G}$ such that the resulting graph has no subgraph isomorphic to $\mathcal{H}$?

A-Hierarchy (example 7)

The parameterized model checking problem for a class Φ of formulas:

p-MC(Φ)
Instance: A structure $\mathcal{A}$ and a formula $\varphi \in \Phi$.
Parameter: $|\varphi|$.
Problem: Decide whether $\varphi(\mathcal{A}) \neq \varnothing$.

Definition (Flum, Grohe, 2001)

$A[t] := \left[p\text{-MC}(\Sigma_t)\right]^{\text{fpt}}$, for $t \geq 1$, form the *A-hierarchy*.

Examples

- p-Clique-Dominating-Set $\in$ A$[2]$.

p-Clique-Dominating-Set
Instance: Graphs $\mathcal{G}$, and $k, \ell \in \mathbb{N}$.
Parameter: $k + \ell$, where ℓ the number of vertices of $\mathcal{H}$.
Problem: Decide whether $\mathcal{G}$ contains a set of k vertices from $\mathcal{G}$ that dominates every clique of ℓ elements.

A-Hierarchy (properties)

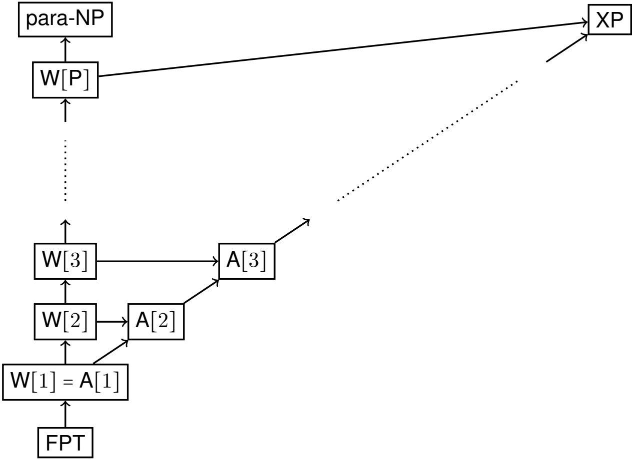
Theorems

T1. A$[1] \subseteq$ W[P].

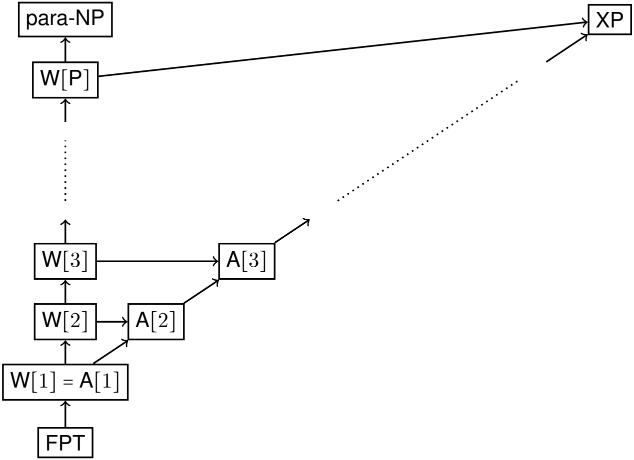
T2. W$[t] \subseteq$ A$[t]$, for all $t \in \mathbb{N}$.

- Unlikely: A$[t] \subseteq$ W$[t]$, for $t > 1$.
  Reason:
  - the A-hierarchy are parameterizations of problems that are complete for the levels of the polynomial hierarchy
  - the W-hierarchy is a refinement of NP in parameterized complexity
- Unlikely: $\left[p\text{-MC(FO)}\right]^{\text{fpt}} = \bigcup_{i=1}^{\infty} \text{A}[i]$,
  contrasting with: $\left[p\text{-WD-FO}\right]^{\text{fpt}} = \bigcup_{i=1}^{\infty} \text{W}[i]$.

W-Hierarchy and A-Hierarchy versus para-NP and XP



W-Hierarchy and A-Hierarchy versus para-NP and XP



Revisiting the two problems at start today

QUERIES
Instance: a relational database D, a conjunctive query α.
Parameter: size $k = |\alpha|$ of query α
Compute: answer to query α from database D.

- QUERIES ∈ NP-complete.
- QUERIES ∈ $O(n^k)$ for $n = \|D\|$, which does not give an FPT result.

LTL-MODEL-CHECKING
Instance: a Kripke structure (state space) $\mathcal{K}$, an LTL formula φ
Parameter: size $k = |\varphi|$ of formula φ
Question: Does $\mathcal{K} \vDash \varphi$ hold?

- LTL-MODEL-CHECKING ∈ PSPACE-complete,
- LTL-MODEL-CHECKING ∈ $O(k \cdot 2^{2^k} \cdot n)$ ∈ FPT for $n = \|\mathcal{K}\|$.

Revisiting the two problems at start today

QUERIES
Instance: a relational database D, a conjunctive query α.
Parameter: size $k = |\alpha|$ of query α
Compute: answer to query α from database D.

- QUERIES $\in$ NP-complete.
- QUERIES $\in O(n^k)$ for $n = \|D\|$, which does not give an FPT result.
- QUERIES $\in$ W[1] (= strong evidence for it likely not to be in FPT).

LTL-MODEL-CHECKING
Instance: a Kripke structure (state space) $\mathcal{K}$, an LTL formula φ
Parameter: size $k = |\varphi|$ of formula φ
Question: Does $\mathcal{K} \models \varphi$ hold?

- LTL-MODEL-CHECKING $\in$ PSPACE-complete,
- LTL-MODEL-CHECKING $\in O(k \cdot 2^{2^k} \cdot n) \in$ FPT for $n = \|\mathcal{K}\|$.

Summary

- Motivation for fixed-parameter intractability
- Fixed parameter reductions
- The classes para-NP and XP
- The class W[P]
- Logic preliminaries (continued)
- W-hierarchy
  - definitions
    - with Boolean circuits
    - as parameterized weighted Fagin definability problems
- A-hierarchy
  - definition as parameterized model-checking problems
- picture overview of these classes

Course overview

Monday, June 8 11.00 – 13.00	Tuesday, June 9 11.00 – 13.00	Wednesday, June 10 11.00 – 13.00	Thursday, June 11 11.00 – 13.00	Friday, June 13
Algorithmic Techniques			*Formal-Method & Algorithmic Techniques*	
Introduction & basic FPT results	**Notions of bounded graph width**	**Example: Good Edge Labelings**	**Algorithmic Meta-Theorems**	
motivation for FPT kernelization, Crown Lemma, Sunflower Lemma	path-, tree-, clique width, FPT-results by dynamic programming, transferring FPT results betw. width	*Guest Lecture* Davi de Andrade Iacono	1st-order logic, monadic 2nd-order logic, FPT-results by Courcelle's Theorems for tree and clique-width	
				14.30 – 16.30
				FPT-Intractability Classes & Hierarchies
				motivation for FP-intractability results, FPT-reductions, class XP (slicewise polynomial), W- and A-Hierarchies, placing problems on these hierarchies

Example suggestions

Examples

1. FPT results transfer backwards over fpt-reductions:
   If $\langle Q_1, \kappa_1 \rangle \leq_{\text{fpt}} \langle Q_2, \kappa_2 \rangle$, then $Q_2 \in$ FPT implies $Q_1 \in$ FPT.
2. Find the idea for:
   p-DOMINATING-SET $\equiv_{\text{fpt}}$ p-HITTING-SET.
3.

References

Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Daniel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh.
Parameterized Algorithms.
Springer, 1st edition, 2015.

Jörg Flum and Martin Grohe.
Parameterized Complexity Theory.
Springer, 2006.