

Lecture 4: Fixed-Parameter Intractability

(A Short Introduction to Parameterized Complexity)

Clemens Grabmayer

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

June 14, 2024

Course overview

| Monday, June 10 10.30 – 12.30 | Tuesday, June 11 | Wednesday, June 12 10.30 – 12.30 | Thursday, June 13 | Friday, June 14 |
|--|--|--|-------------------|---|
| Introduction & basic FPT results motivation for FPT kernelization, Crown Lemma, Sunflower Lemma | | Algorithmic Meta-Theorems 1st-order logic, monadic 2nd-order logic, FPT-results by Courcelle's Theorems for tree and clique-width | | |
| | GDA | | GDA | GDA |
| <i>Algorithmic Techniques</i> | | <i>Formal-Method & Algorithmic Techniques</i> | | |
| | 14.30 – 16.30 | | | 14.30 – 16.30 |
| | Notions of bounded graph width path-, tree-, clique width, FPT-results by dynamic programming, transferring FPT results betw. widths | | | FPT-Intractability Classes & Hierarchies motivation for FP-intractability results, FPT-reductions, class XP (slicewise polynomial), W- and A-Hierarchies, placing problems on these hierarchies |
| | | GDA | GDA | |

Overview

- ▶ simple graphs (Correction)
- ▶ 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

Expressing graph properties by **MSO** formulas (5)

Exercise

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

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

Note:

- ▶ This property is not expressible by a (single) $\text{MSO}(\tau_G)$ formula.
- ▶ Other properties that are **not** $\text{MSO}(\tau_G)$ expressible:
 - ▶ balanced bipartite graphs
 - ▶ existence of a perfect matching
 - ▶ simple graphs (graphs with no parallel edges)
 - ▶ existence of spanning trees with maximum degree 3



the edge predicate $E(\cdot, \cdot)$ alone can obviously not express absence of parallel edges

MSO(\mathcal{T}_G) formula or
MSO₁ formula or

$$\begin{aligned}\forall x \neg E(x, x) \\ \forall_{(\text{vert})} x \neg E(x, x)\end{aligned}$$

expresses

Self-Loop-freeness

C^u C^v
Self-Loops

$$A_{\mathcal{T}_G}(G) \models \forall x \neg E(x, x)$$



G contains no self-loops

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 $\text{MSO}(\tau_G)$ formula that expresses that a graph has a dominating set of $\leq k$ elements.
3. Find an MSO_2 or $\text{MSO}(\tau_{HG})$ formula $\text{feedback}(S)$ that expresses that $S \subseteq V$ is a feedback vertex set.
4. (*) Find an MSO_1 or $\text{MSO}(\tau_G)$ formula that expresses that a graph is connected.
5. (*) Find an MSO_2 or $\text{MSO}(\tau_{HG})$ formula $\text{path}(x, y, Z)$ that expresses that Z is a set of edges that forms a path from x to y .

1. First-order logic formula "graph has cycle of length = k "
- $$\varphi_{k\text{-cycle}} := \exists x_1 \dots \exists x_k ((\bigwedge_{1 \leq i < j \leq k} \neg x_i = x_j) \wedge (\bigwedge_{1 \leq i \leq k-1} E(x_i, x_{i+1})) \wedge E(x_k, x_1))$$
- for $G = \langle V, E \rangle$ and $k \geq 3$: $\mathcal{A}_{\mathbb{J}_G}(G) \models \varphi_{k\text{-cycle}} \Leftrightarrow G \text{ has a } k\text{-cycle}$
- $|\varphi_{k\text{-cycle}}| \in O(k^2)$

2. MSO(\mathbb{J}_G) or MSO₁ formula: "graph G has a dominating set of $\leq k$ elements".
 $G = \langle V, E \rangle$. $S \subseteq V$ is dominating set: $\Leftrightarrow \forall v \in V [v \notin S \rightarrow \exists e = \{v, u\} \in E (u \in S)]$

$$\begin{aligned} \varphi_{\text{domset}}(x) &:= \forall x (\neg X(x) \rightarrow \exists y (E(x, y) \wedge X(y))) \in \text{For}(\text{MSO}(\mathbb{J}_G)) \\ &\quad \forall_{(\text{vert})} x (\neg X(x) \rightarrow \exists_{(\text{vert})} y (E(x, y) \wedge X(y))) \in \text{For}(\text{MSO}_1) \end{aligned}$$

$$\varphi_{\text{domset} \leq k}(x) := \varphi_{\text{domset}}(x) \wedge \exists_{(\text{vert})} x_1 \dots \exists_{(\text{vert})} x_k (\forall_{(\text{vert})} y (X(y) \leftrightarrow \bigvee_{i=1}^k y = x_i))$$

$$\varphi_{\exists \text{domset} \leq k} := \exists_{(\text{vert})} x \varphi_{\text{domset} \leq k}(x)$$

Then $\mathcal{A}(G) \models \varphi_{\exists \text{domset} \leq k} \Leftrightarrow G \text{ contains a dominating set of } \leq k \text{ elements}$



3. MSO(Σ_{HG})-formula for "S is a feedback-vertex set"



$\varphi_{fb}(X)$

\Leftrightarrow contains a vertex of every cycle
 \Leftrightarrow contains a vertex of every disjoint cycle collection

disjoint-cycle-collection (Υ) :=

$$\begin{aligned} &:= \forall e (\Upsilon(e) \rightarrow \text{EDGE}(e)) \\ &\wedge \forall v (\text{VERT}(v) \wedge \exists e (\Upsilon(e) \wedge \text{INC}(v, e))) \\ &\rightarrow \exists^{\leq 2}_e (\Upsilon(e) \wedge \text{INC}(v, e)) \end{aligned}$$

$$\begin{aligned} \varphi_{fb}(X) := & \forall x (X(x) \rightarrow \text{VERT}(x)) \wedge \\ & \forall C (\text{disjoint-cycle-collection } (C) \rightarrow \\ & \rightarrow \exists x (X(x) \wedge \exists e (C(e) \wedge \text{INC}(e)))) \end{aligned}$$

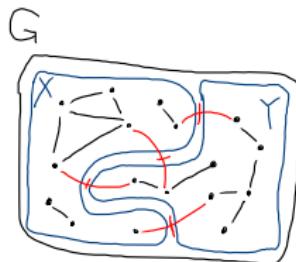
Then: for all finite graphs $G = \langle V, E \rangle$:

$$A_{\Sigma_{HG}}(G) \models \varphi(S) \iff S \text{ is a feedback vertex set of } G$$

4. Find an MSO₁ or MSO(Σ_{HG}) formula "G is connected"

$$\varphi_{\text{conn}} := \neg \exists_{(\text{vert})} X \exists_{(\text{vert})} Y \text{ Partition}(X, Y)$$

$$\begin{aligned} \text{Partition}(X, Y) := & \forall_{(\text{vert})} X \left((X(x) \vee Y(x)) \wedge \neg (X(x) \wedge Y(x)) \right) \\ & \wedge (\exists_{(\text{vert})} X(x) \wedge \exists_{(\text{vert})} Y(x)) \\ & \wedge \neg \exists_{(\text{vert})} X \exists_{(\text{vert})} Y (X(x) \wedge Y(y) \wedge E(x, y)) \end{aligned}$$



partition of G
into non-empty subsets X and Y of vertices
that are not connected by any edge

5. Find an MSO_2 / $\text{MSO}(\text{J}_{\text{HG}})$ formula $\text{path}(x, y, \bar{Z})$ such that

for every finite graph $G = \langle V, E \rangle$

for every $\bar{Z} \subseteq E$ and $x, y \in V$:

$$\text{J}_{\text{HG}}(G) \models \text{path}(\bar{x}, \bar{y}, \bar{Z}) \iff$$

$\boxed{x \neq y}$ and
 $\boxed{\bar{Z} \text{ is a path (of edges) in } G \text{ from } x \text{ to } y}$

$$\boxed{\text{path}(x, y, \bar{Z})} :=$$

$$\neg x = y \wedge \text{VERT}(x) \wedge \text{VERT}(y) \wedge \forall e (\bar{Z} \mid e \rightarrow \text{EDGE}(e))$$

$$\wedge \exists_e^= (\bar{Z} \mid e \wedge \text{INC}(x, e))$$

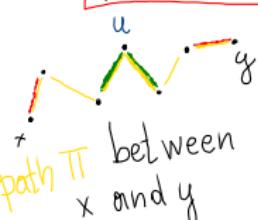
$$\wedge \exists_e^= (\bar{Z} \mid e \wedge \text{INC}(y, e))$$

$$\wedge \forall u (\text{VERT}(u) \wedge \exists e (\bar{Z} \mid e \wedge \text{INC}(u, e)))$$

$$\wedge \neg u = x \wedge u = y$$

$$\rightarrow \exists_e^= (\bar{Z} \mid e \wedge \text{INC}(u, e))$$

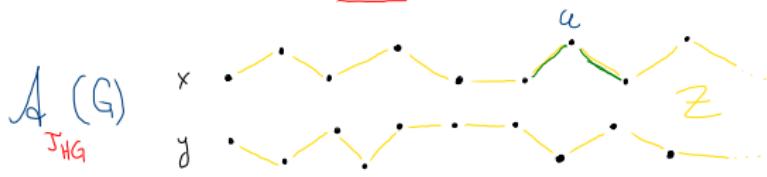
every vertex on an edge in \bar{Z} except x and y
is incident to precisely 2 edges of \bar{Z}



x and y are incident to precisely 1 edge of Π

every internal vertex u of the path Π is incident to precisely 2 edges of Π

Note: for infinite graphs G this formula
is not correct

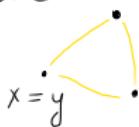


while $\mathcal{A}_{\text{JHG}}(G) \models \text{path}(x, y, z)$

holds (and all vertices a at internal edges have precisely 2 incident edges) it may be

impossible to reach y from x in finitely many steps ($\Rightarrow z$ is no path)

Further more: if $x = y$ were permitted, cyclic paths would have to be included



\Rightarrow slight change of the formula $\text{path}(x, y, z)$

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 \in NP-complete.

LTL-MODEL-CHECKING

Instance: a Kripke structure (state space) \mathcal{K} , an LTL formula φ

Parameter: size $|\varphi|$ of formula φ

Question: Does $\mathcal{K} \models \varphi$ hold?

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.

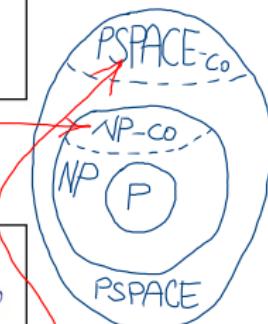
LTL-MODEL-CHECKING

Instance: a Kripke structure (state space) \mathcal{K} , an LTL formula φ

Parameter: size $|\varphi|$ 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.

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,

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

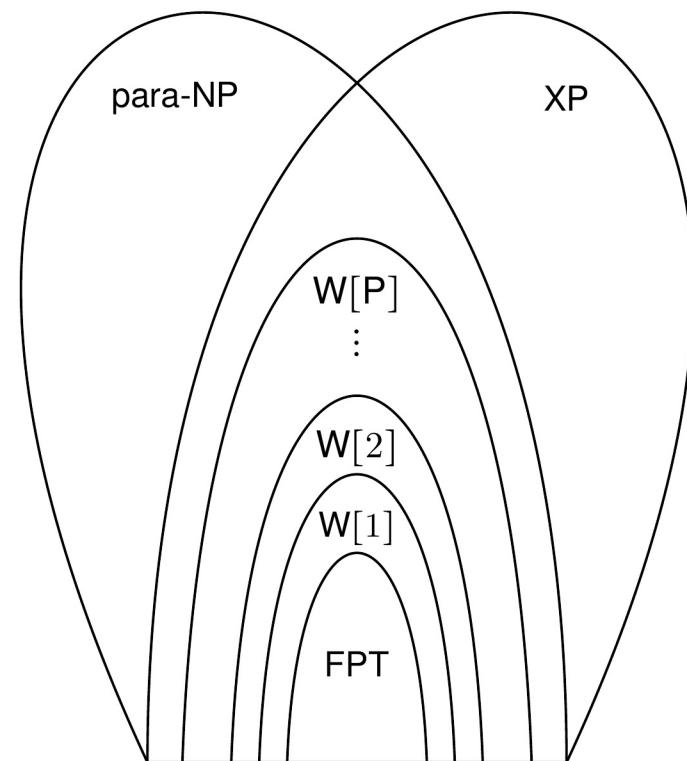
'The purpose [...] is to give evidence that certain problems are not fixed-parameter tractable (just as the main purpose of the theory of NP-completeness is to give evidence that certain problems are not polynomial time computable.)'

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

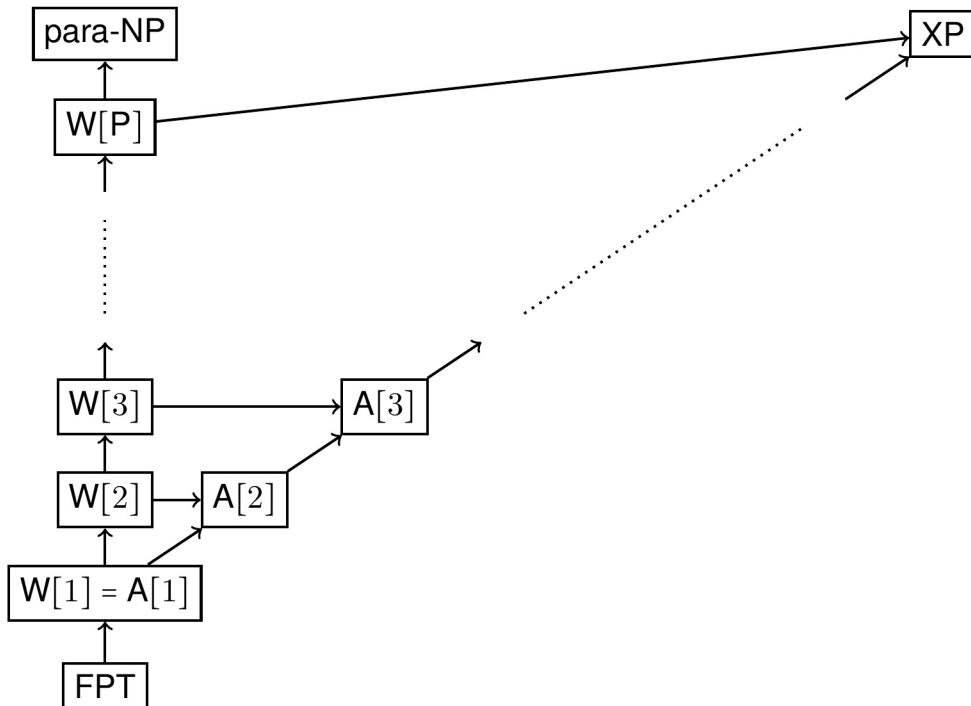
Unfortunately, the world of parameterized intractability is more complex: There is a big variety of seemingly different classes of intractable parameterized problems.'

(Flum, Grohe [2])

W-Hierarchy versus para-NP and XP



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



Fixed-Parameter tractable

Definition

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

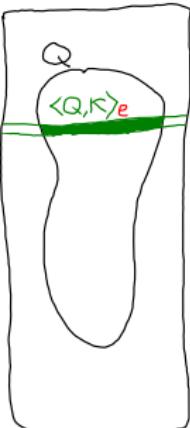
$\exists f : \mathbb{N} \rightarrow \mathbb{N}$ computable $\exists p \in \mathbb{N}[X]$ polynomial

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

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

Σ^*

Slices of parameterized problems



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

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

Proposition (slices of FPT problems are in PTIME)

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

Proof

Let ℓ be fixed. Then for all $x \in \Sigma^*$:

Decide $x \in Q, \kappa(x) = \ell$ in time $\leq f(\kappa(x)) \cdot p(|x|) = f(\ell) \cdot p(|x|) \in \text{PTIME}$.

A problem not in FPT / XP

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

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

XP

Proposition:

Slices of FPT problems are in PTIME

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

∈ XP

p -COLORABILITY

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

Parameter: ℓ .

Problem: Decide whether \mathcal{G} is ℓ -colorable.

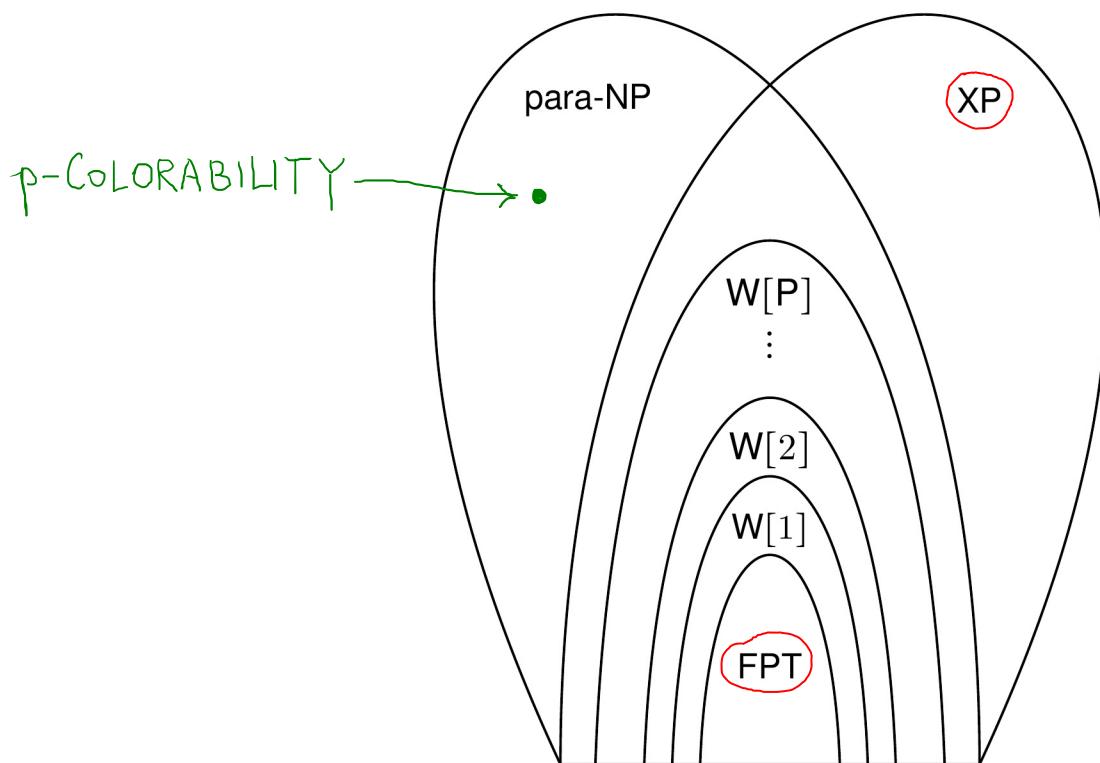
notin XP

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.

notin XP

W-Hierarchy versus para-NP and XP



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

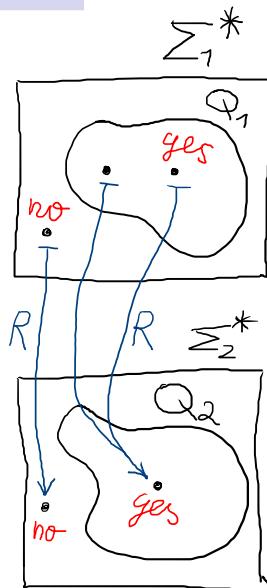
$$R : \Sigma_1^* \rightarrow \Sigma_2^*$$

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 (Richard Karp)

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

$R : \Sigma_1^* \rightarrow \Sigma_2^*$:

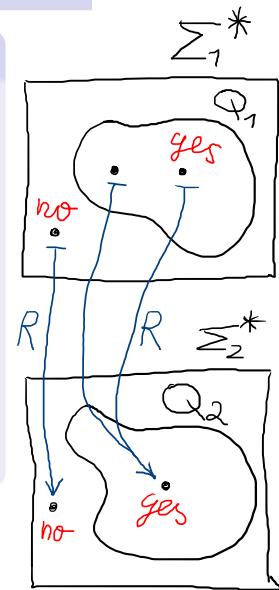
R1. $(x \in Q_1 \iff R(x) \in Q_2)$ for all $x \in \Sigma_1^*$.

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 many-one reduction)



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 $R : \Sigma_1^* \rightarrow \Sigma_2^*$:

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:

| |
|--|
| $\langle Q_1, \Sigma_1 \rangle \in \mathbf{P} \iff \langle Q_2, \Sigma_2 \rangle \in \mathbf{P}$. |
| $\langle Q_1, \Sigma_1 \rangle \notin \mathbf{P} \implies \langle Q_2, \Sigma_2 \rangle \notin \mathbf{P}$. |

Proposition.

if $\langle Q_1, \Sigma_1 \rangle \leq_p \langle Q_2, \Sigma_2 \rangle$

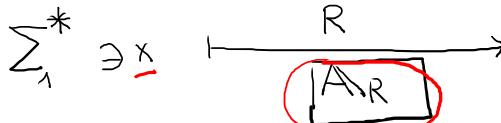
then

$$Q_1 \in P$$

$$Q_2 \in P$$

Proof Idea.

Construct alg. A_R



$R(x) \in \Sigma_2^*$

polynomial decision alg.
for Q_2

$$\text{time}_{A_2}(x) \leq p_2(|x|)$$

$$\text{time } A_R(x) =$$

$$\text{time } A_R(x) + \text{time } A(R(x))$$

$$\leq p_R(|x|) + p_2(|R(x)|)$$

$$\leq |x| + p_R(|x|)$$

$$\leq p_R(|x|) + p_2(|x| + p_R(|x|))$$

$$\leq \tilde{p}(|x|) \quad \text{for polynomial } \tilde{p}(n) := p_R(n) + p_2(n + p_R(n))$$

$$R(x) \notin Q_2$$

$$R(x) \in Q_2$$

$$x \notin Q_1$$

$$R(x) \in Q_2$$

$$x \in Q_1$$

$$(R1) \Updownarrow$$

$$x \notin Q_1$$

$$(R2) \Updownarrow$$

$$x \in Q_1$$

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

$$R : \Sigma_1^* \rightarrow \Sigma_2^*$$

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 $p(|x|)$.

$$\langle Q_1, \Sigma_1 \rangle \leq_{\text{pol}} \langle Q_2, \Sigma_2 \rangle :=$$

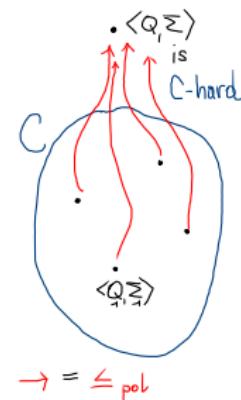
there is a polynomial-time reduction from $\langle Q_1, \Sigma_1 \rangle$ to $\langle Q_2, \Sigma_2 \rangle$.

Proposition

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

Let \mathbf{C} be class of classical problems.

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



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

$$R : \Sigma_1^* \rightarrow \Sigma_2^*$$

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 $p(|x|)$.

$$\langle Q_1, \Sigma_1 \rangle \leq_{\text{pol}} \langle Q_2, \Sigma_2 \rangle :=$$

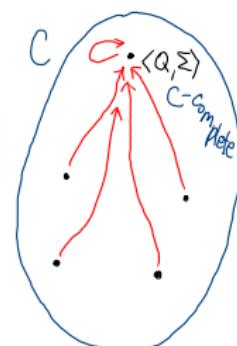
there is a polynomial-time reduction from $\langle Q_1, \Sigma_1 \rangle$ to $\langle Q_2, \Sigma_2 \rangle$.

Proposition

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

Let **C** be class of classical problems.

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



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

$$R : \Sigma_1^* \rightarrow \Sigma_2^*$$

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

R2. R is computable by an 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 an polynomial-time reduction from $\langle Q, \kappa \rangle$ to $\langle Q', \kappa' \rangle$.

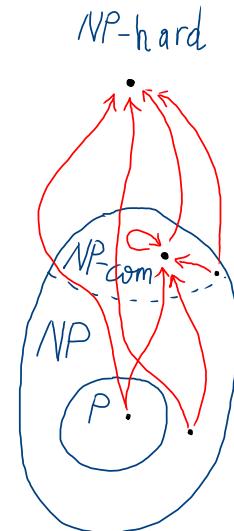
Proposition

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

$$\langle Q_1, \Sigma_1 \rangle \notin \mathbf{P} \implies \langle Q_2, \Sigma_2 \rangle \notin \mathbf{P}.$$

Let \mathbf{C} be class of classical problems.

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



Fixed-parameter tractable reductions

Definition

Let $\langle Q_1, \Sigma_1, \kappa_1 \rangle$, $\langle Q_2, \Sigma_2, \kappa_2 \rangle$ be parameterized problems.

An **fpt-reduction** from $\langle Q_1, \kappa_1 \rangle$ to $\langle Q_2, \kappa_2 \rangle$ is a mapping

$R : \Sigma_1^* \rightarrow (\Sigma_2)^*$:

- 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 $\langle Q_1, \Sigma_1, \kappa_1 \rangle, \langle Q_2, \Sigma_2, \kappa_2 \rangle$ be parameterized problems.

An **fpt-reduction** from $\langle Q_1, \kappa_1 \rangle$ to $\langle Q_2, \kappa_2 \rangle$ is a mapping

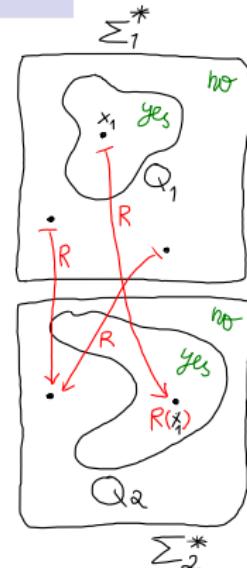
$$R : \Sigma_1^* \rightarrow (\Sigma_2)^*$$

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 $g : \mathbb{N} \rightarrow \mathbb{N}$.

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



$$\kappa_2(R(x)) \leq g(\kappa_1(x))$$

Fixed-parameter tractable reductions

Definition

Let $\langle Q_1, \Sigma_1, \kappa_1 \rangle, \langle Q_2, \Sigma_2, \kappa_2 \rangle$ be parameterized problems.

An **fpt-reduction** from $\langle Q_1, \kappa_1 \rangle$ to $\langle Q_2, \kappa_2 \rangle$ is a mapping

$$R : \Sigma_1^* \rightarrow (\Sigma_2)^*$$

- 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 $g : \mathbb{N} \rightarrow \mathbb{N}$.

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

Proposition

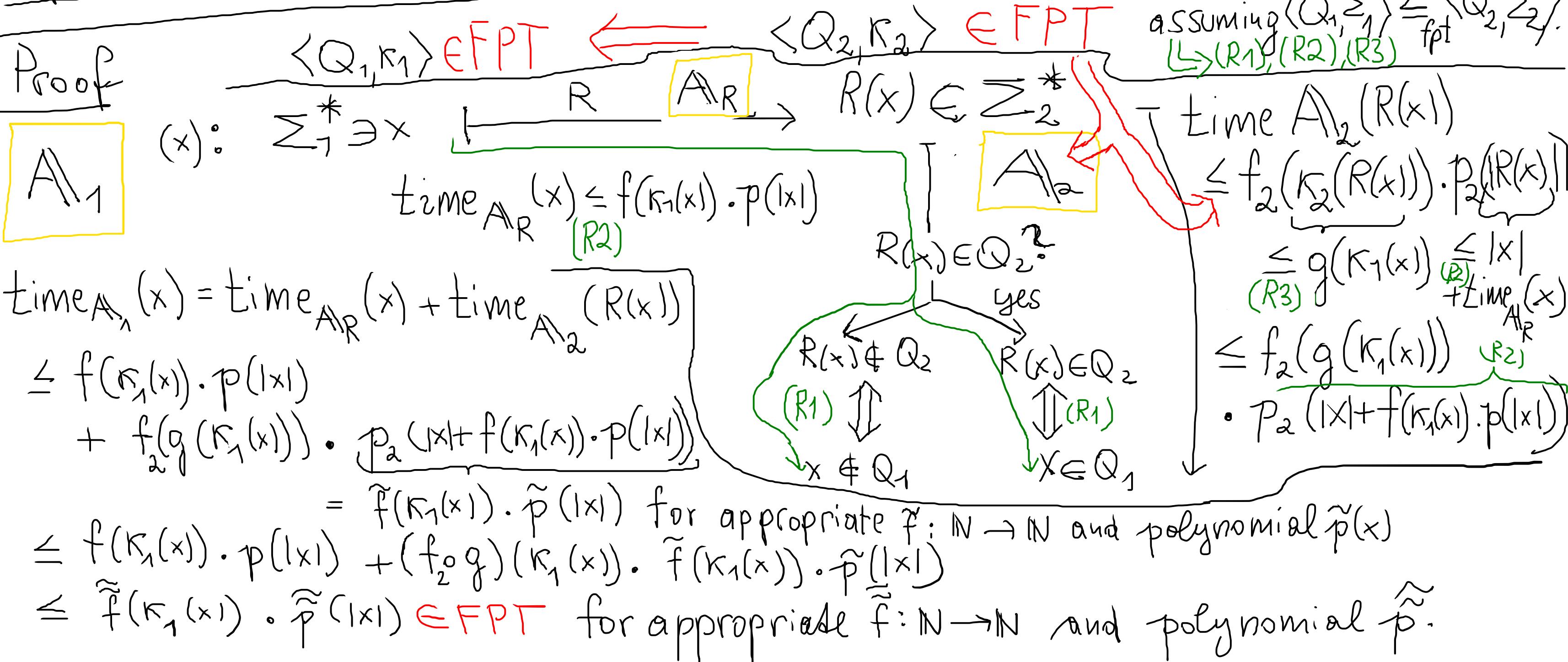
If $\langle Q_1, \kappa_1 \rangle \leq_{\text{fpt}} \langle Q_2, \kappa_2 \rangle$, then: $\langle Q_1, \kappa_1 \rangle \in \text{FPT} \iff \langle Q_2, \kappa_2 \rangle \in \text{FPT}$.
 $\langle Q_1, \kappa_1 \rangle \notin \text{FPT} \implies \langle Q_2, \kappa_2 \rangle \notin \text{FPT}$.

$\langle Q_1, \kappa_1 \rangle, \langle Q_2, \kappa_2 \rangle$ parameterized problems

Def. $\langle Q_1, \kappa_1 \rangle \leq_{\text{fpt}} \langle Q_2, \kappa_2 \rangle \iff \exists R: \sum_1^* \rightarrow \sum_2^*: (R_1) \forall x \in \sum_1^* (x \in Q_1 \Leftrightarrow R(x) \in Q_2),$
 (R_2) computable in $f(\kappa_1(x)) \cdot p(|x|)$,
 $(R_3) \kappa_2(R(x)) \leq g(\kappa_1(x)).$

Prop. $\langle Q_1, \kappa_1 \rangle \leq_{\text{fpt}} \langle Q_2, \kappa_2 \rangle \Rightarrow (\langle Q_1, \kappa_1 \rangle \in \text{FPT} \Leftarrow \langle Q_2, \kappa_2 \rangle \in \text{FPT})$

Proof



Comparing parameterizations (revisited)

Definition (computably bounded below)

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

- ▶ $\kappa_1 \succeq \kappa_2 : \iff \exists g : \mathbb{N} \rightarrow \mathbb{N} \text{ computable } \forall x \in \Sigma^* [g(\kappa_1(x)) \geq \kappa_2(x)].$
- ▶ $\kappa_1 \approx \kappa_2 : \iff \kappa_1 \succeq \kappa_2 \wedge \kappa_2 \succeq \kappa_1.$
- ▶ $\kappa_1 > \kappa_2 : \iff \kappa_1 \succeq \kappa_2 \wedge \neg(\kappa_2 \succeq \kappa_1).$

Proposition

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

$$\begin{aligned}\langle Q, \kappa_1 \rangle \in \text{FPT} &\iff \langle Q, \kappa_2 \rangle \in \text{FPT}, \\ \langle Q, \kappa_1 \rangle \notin \text{FPT} &\implies \langle Q, \kappa_2 \rangle \notin \text{FPT}.\end{aligned}$$

Comparing parameterizations (revisited)

Definition (computably bounded below)

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

- ▶ $\kappa_1 \succeq \kappa_2 : \iff \exists g : \mathbb{N} \rightarrow \mathbb{N} \text{ computable } \forall x \in \Sigma^* [g(\kappa_1(x)) \geq \kappa_2(x)].$
- ▶ $\kappa_1 \approx \kappa_2 : \iff \kappa_1 \succeq \kappa_2 \wedge \kappa_2 \succeq \kappa_1.$
- ▶ $\kappa_1 > \kappa_2 : \iff \kappa_1 \succeq \kappa_2 \wedge \neg(\kappa_2 \succeq \kappa_1).$

Proposition

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

$$\begin{aligned}\langle Q, \kappa_1 \rangle \in \text{FPT} &\iff \langle Q, \kappa_2 \rangle \in \text{FPT}, \\ \langle Q, \kappa_1 \rangle \notin \text{FPT} &\implies \langle Q, \kappa_2 \rangle \notin \text{FPT}.\end{aligned}$$

Proposition

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

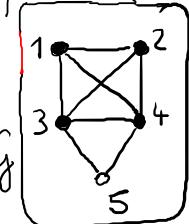
$$\kappa_1 \succeq \kappa_2 \iff \langle Q, \kappa_1 \rangle \leq_{\text{fpt}} \langle Q, \kappa_2 \rangle \text{ via } R : \Sigma^* \rightarrow \Sigma^*, x \mapsto x.$$

Fixed-parameter tractable reductions

Examples

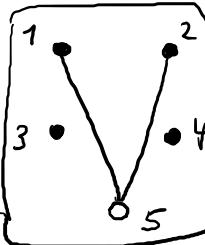
- ▶ p -CLIQUE \equiv_{fpt} p -INDEPENDENT-SET.
- ▶ p -DOMINATING-SET \equiv_{fpt} p -HITTING-SET.

p- CLIQUE



\equiv_{fpt}

p- INDEPENDENT-SET



$\langle V, E \rangle = G$

$$S = \{1, 2, 3, 4\}$$

k- clique

$$\bar{G} = \langle V, \{\{u, v\} / u \neq v, \{u, v\} \notin E\} \rangle$$

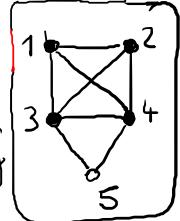
Complement graph

$$S = \{1, 2, 3, 4\}$$

k-independent set

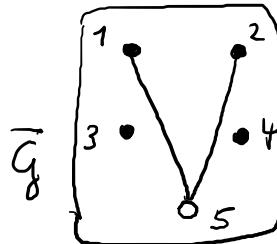
In general: $S \subseteq V$ k-clique in $G \iff S \subseteq V$ is k-element independent set in \bar{G}

p- CLIQUE



\equiv_{fpt}

p-INDEPENDENT-SET



$\langle V, E \rangle = G$

$$S = \{1, 2, 3, 4\}$$

k-clique

$\bar{G} = \langle V, \{\{u, v\} / u \neq v, \{u, v\} \notin E\} \rangle$

complement graph

$$S = \{1, 2, 3, 4\}$$

k-independent set

(R1)

In general: $S \subseteq V$ k-clique in $G \iff S \subseteq V$ is k-element independent set in \bar{G}

$R: \langle G, k \rangle \xrightarrow{R} \langle \bar{G}, k \rangle$ is \leq_{fpt} -reduction

Complement graph

(R1) ✓

(R2) computable by
poly. algor.

(R3) $K_2(R(\langle G, k \rangle)) = K_2(\langle \bar{G}, k \rangle) = k$
 $= K_1(\langle G, k \rangle) = \text{id}(K_1(\langle G, k \rangle))$

Fixed-parameter tractable reductions

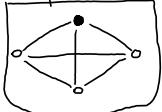
Examples

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

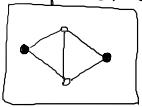
Non-Example

- ▶ For graphs $\mathcal{G} = \langle V, E \rangle$, and sets $X \subseteq V$:
 X is independent set of $\mathcal{G} \iff V \setminus X$ is a vertex cover of \mathcal{G}
yields a polynomial reduction between $p\text{-INDEPENDENT-SET}$ and $p\text{-VERTEX-COVER}$, but does not yield an fpt-reduction.

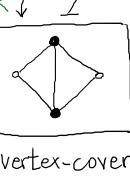
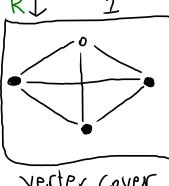
independent set



independent set



Fixed-parameter tractable reductions



vertex cover

Examples

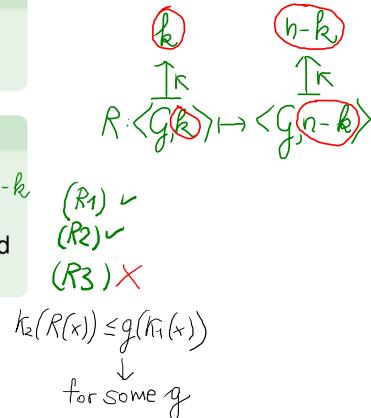
- ▶ $p\text{-CLIQUE} \equiv_{\text{fpt}} p\text{-INDEPENDENT-SET}$.
- ▶ $p\text{-DOMINATING-SET} \equiv_{\text{fpt}} p\text{-HITTING-SET}$.

Non-Example

- ▶ For graphs $\mathcal{G} = \langle V, E \rangle$, and sets $X \subseteq V$:
 X is independent set of \mathcal{G} of size k $\iff V \setminus X$ is a vertex cover of \mathcal{G} of size $n-k$
yields a polynomial reduction between $p\text{-INDEPENDENT-SET}$ and $p\text{-VERTEX-COVER}$, but does not yield an fpt-reduction.

$$\forall x \in \Sigma^* (K_2(R(x)) \leq g(K_1(x)))$$

$$\forall n \forall k (n-k \leq g(k)) \times$$

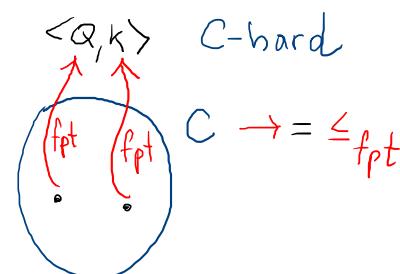
 $p\text{-INDEPENDENT-SET}$ Instance: graph G , $k \in \mathbb{N}$ Parameter: $k \in \mathbb{N}$ Question: Does G have an independent set of size $\geq k$? $p\text{-VERTEX-COVER}$ Instance: graph G , $k \in \mathbb{N}$ Parameter: $k \in \mathbb{N}$ Question: Does G have a vertex cover of size $\leq k$?

Fpt-reduction closure / hardness / reducibility

Let \mathbf{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 \mathbf{C} is fpt-reducible to $\langle Q, \kappa \rangle$

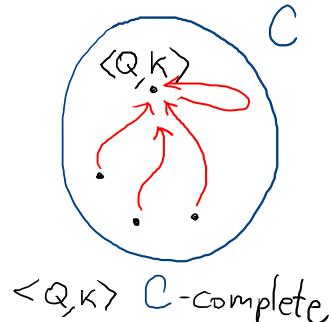


Fpt-reduction closure / hardness / reducibility

Let \mathbf{C} be a class of parameterized problems.

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

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



Fpt-reduction closure / hardness / reducibility

Let \mathbf{C} be a class of parameterized problems.

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

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

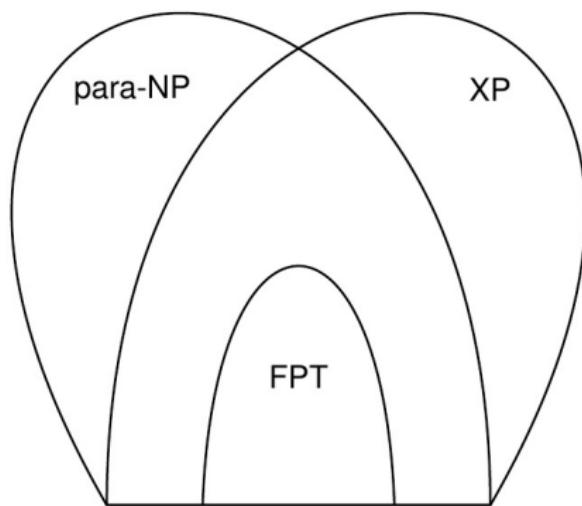
Fpt-reduction closure / hardness / reducibility

Let \mathbf{C} be a class of parameterized problems.

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

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

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 $f : \mathbb{N} \rightarrow \mathbb{N}$, 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 $f : \mathbb{N} \rightarrow \mathbb{N}$, 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 $f : \mathbb{N} \rightarrow \mathbb{N}$, 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.
- ▶ $\text{NP} \subseteq \text{para-NP}$.

para-NP

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in **para-NP** if there is a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$, 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.
- ▶ $\text{NP} \subseteq \text{para-NP}$.

Example

- ▶ $p\text{-CLIQUE}$, $p\text{-INDEPENDENT-SET}$, $p\text{-DOMINATING-SET}$,
 $p\text{-HITTING-SET}$, $p\text{-COLORABILITY} \in \text{para-NP}$.

para-NP

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in **para-NP** if there is a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$, 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.
- ▶ $\text{NP} \subseteq \text{para-NP}$.

Example

- ▶ $p\text{-CLIQUE}$, $p\text{-INDEPENDENT-SET}$, $p\text{-DOMINATING-SET}$,
 $p\text{-HITTING-SET}$, $p\text{-COLORABILITY} \in \text{para-NP}$.
- ▶ 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 $f : \mathbb{N} \rightarrow \mathbb{N}$, 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.
- ▶ $\text{NP} \subseteq \text{para-NP}$.

Example

- ▶ $p\text{-CLIQUE}$, $p\text{-INDEPENDENT-SET}$, $p\text{-DOMINATING-SET}$, $p\text{-HITTING-SET}$, $p\text{-COLORABILITY} \in \text{para-NP}$.
- ▶ $\text{FPT} = \text{para-NP}$ if and only if $\text{PTIME} = \text{NP}$.
- ▶ 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 $f : \mathbb{N} \rightarrow \mathbb{N}$, 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.
- ▶ $\text{NP} \subseteq \text{para-NP}$.

Example

- ▶ $p\text{-CLIQUE}$, $p\text{-INDEPENDENT-SET}$, $p\text{-DOMINATING-SET}$, $p\text{-HITTING-SET}$, $p\text{-COLORABILITY} \in \text{para-NP}$.
- ▶ $\text{FPT} = \text{para-NP}$ if and only if $\text{PTIME} = \text{NP}$.
- ▶ 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 $f : \mathbb{N} \rightarrow \mathbb{N}$, and a polynomial $p \in \mathbb{N}[X]$ such that there is a **non-deterministic algorithm** \mathbb{A} such that:

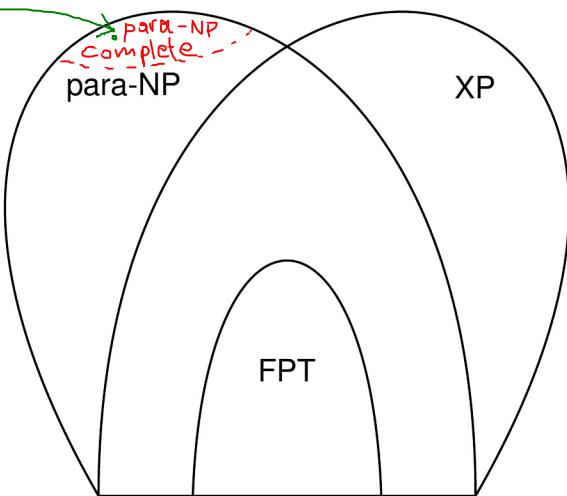
- ▶ \mathbb{A} decides, for all $x \in \Sigma^*$, whether $x \in Q$ in $\leq f(\kappa(x)) \cdot p(|x|)$ steps.
- ▶ para-NP is closed under fpt-reductions.
- ▶ $\text{NP} \subseteq \text{para-NP}$.

Example

- ▶ $p\text{-CLIQUE}$, $p\text{-INDEPENDENT-SET}$, $p\text{-DOMINATING-SET}$, $p\text{-HITTING-SET}$, $p\text{-COLORABILITY} \in \text{para-NP}$.
- ▶ 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**.
 - ▶ $p\text{-COLORABILITY} \in \text{para-NP-complete}$.

para-NP and XP

p-COLORABILITY



XP (slicewise polynomial problems)

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

XP_{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_{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:** XP_{nu} contains undecidable problems:

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

(if slices of instances
are finite)

XP (slicewise polynomial problems)

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

XP_{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:** XP_{nu} contains undecidable problems:

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

Definition

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

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

XP (slicewise polynomial problems)

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

XP_{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:** XP_{nu} contains undecidable problems:

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

Definition

A parameterized problem $\langle Q, \Sigma, \kappa \rangle$ is in **XP** if there is a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$ 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 $f : \mathbb{N} \rightarrow \mathbb{N}$ 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 $\not\subseteq$ 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 $\not\subseteq$ 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 $\not\subseteq$ 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 \subsetneq 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 $\mathcal{A} \models \varphi$ (that is, $\varphi(\mathcal{A}) \neq \emptyset$).

Theorem

MC(FO) can be solved in time $O(|\varphi| \cdot |\mathcal{A}|^w \cdot w)$, 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:

$\text{MC}(\Phi)$

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

$\text{MC}(\text{FO})$ can be solved in time $O(|\varphi| \cdot |\mathcal{A}|^w \cdot w)$, 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:

$p\text{-MC}(\Phi)$.

Instance: A structure \mathcal{A} and a formula $\varphi \in \Phi$.

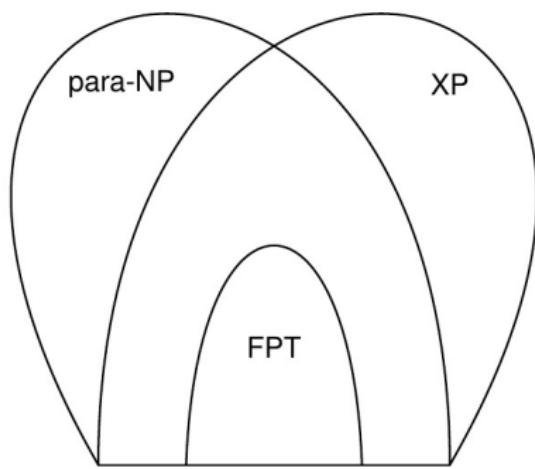
Parameter: $|\varphi|$.

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

Theorem

$p\text{-MC}(\Phi) \in \text{XP}$.

FPT versus para-NP and XP



Proposition

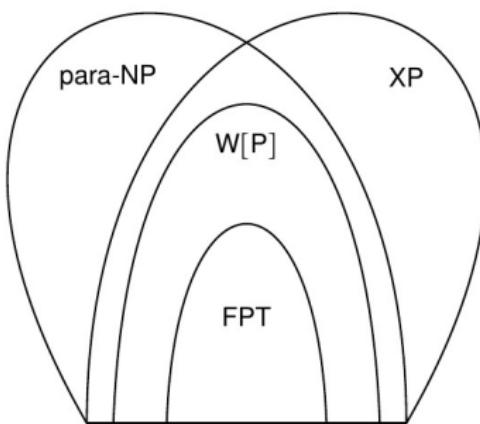
- ▶ $\text{FPT} \subseteq \text{para-NP}$, and:
 $\text{FPT} = \text{para-NP}$ if and only if $\text{PTIME} = \text{NP}$.
- ▶ $\text{para-NP} \not\subseteq \text{XP}$ if $\text{PTIME} \neq \text{NP}$.
- ▶ $\text{FPT} \subsetneq \text{XP}$.

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

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

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

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

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

W[P] and limited non-determinism

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

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

$$\left(\forall x \in \Sigma^* \left((x \in Q \iff \mathbb{M} \text{ accepts } x) \wedge \begin{array}{l} \text{on input } x, \mathbb{M} \text{ halts in } \leq p(|x|) \text{ steps, of which} \\ \text{at most } \leq f(|x|) \text{ are non-deterministic} \end{array} \right) \right)$$

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

W[P] and limited non-determinism

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

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

$$\left(\forall x \in \Sigma^* \left((x \in Q \iff \mathbb{M} \text{ accepts } x) \wedge \text{on input } x, \mathbb{M} \text{ halts in } \leq p(|x|) \text{ steps, of which} \atop \text{at most } \leq f(|x|) \text{ are non-deterministic} \right) \right)$$

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

Fact

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

W[P]

Definition

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

W[P]

Definition

- ▶ Let Σ be an alphabet, and $\kappa : \Sigma^* \rightarrow \mathbb{N}$ a parameterization.
A nondeterministic Turing machine $\textcolor{teal}{M}$ with input alphabet Σ is **κ -restricted** if there are computable functions $f, h : \mathbb{N} \rightarrow \mathbb{N}$ and a polynomial $p \in \mathbb{N}_0[x]$ such that on every run with input $x \in \Sigma^*$ the machine $\textcolor{teal}{M}$ performs
 - ▷ at most $f(\kappa(x)) \cdot p(|x|)$ steps,

W[P]

Definition

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

W[P]

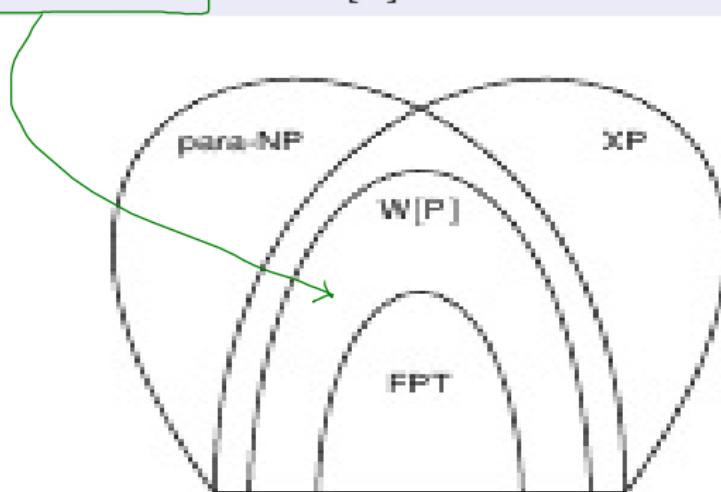
Definition

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

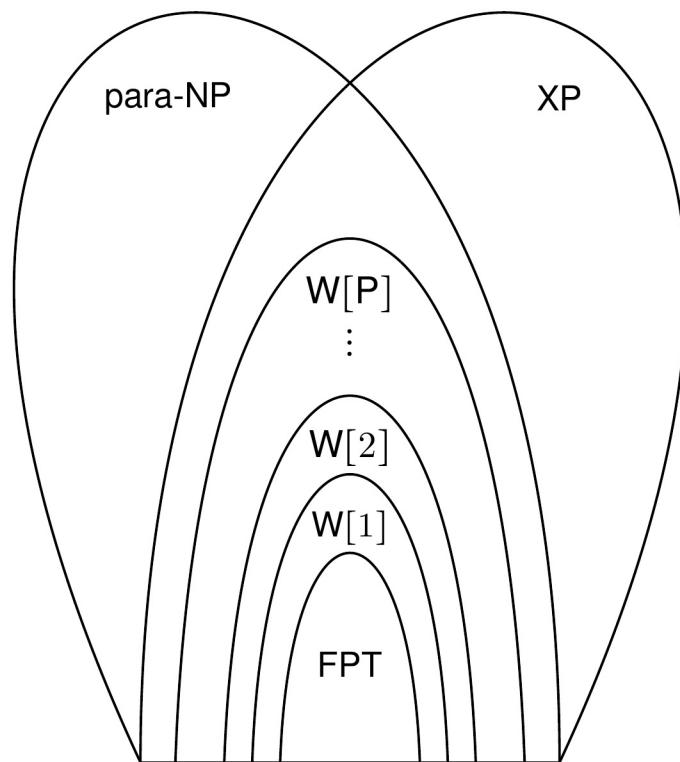
W[P] (properties)

Theorems

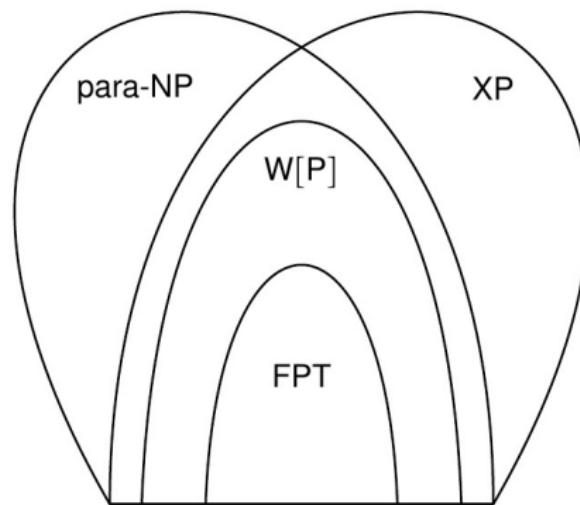
- T1. $\text{FPT} \subseteq W[P] \subseteq XP \cap \text{para-NP}$
- T2. $W[P]$ is closed under fpt-reductions.
- T3. $p\text{-CLIQUE}$, $p\text{-INDEPENDENT-SET}$, $p\text{-DOMINATING-SET}$, and $p\text{-HITTING-SET}$ are in $W[P]$.



W-Hierarchy



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



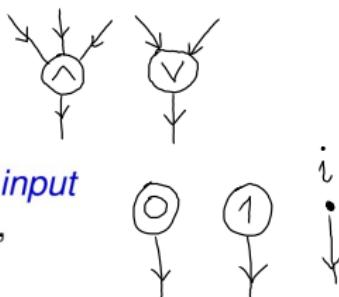
Proposition

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

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 $1, \dots, n$),



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 $1, \dots, n$),
- ▶ one node of out-degree 0 is labeled as *output node*.

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

Definition

We say that C is *k-satisfiable* if C is satisfied by a tuple of weight k .

The W-hierarchy – Boolean circuits

$\neg x_1 \wedge x_2$

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 $1, \dots, n$),
- one node of out-degree 0 is labeled as *output node*.

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

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

Definition

We say that C is *k-satisfiable* if C is satisfied by a tuple of weight k .

$\neg x_1 \wedge x_1$

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 $1, \dots, n$),
- ▶ one node of out-degree 0 is labeled as *output node*.

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

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

Definition

We say that C is *k-satisfiable* if 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.

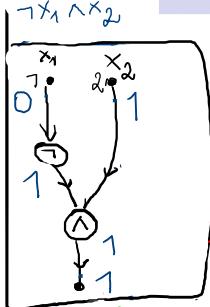
Definition

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 $\leq d$.

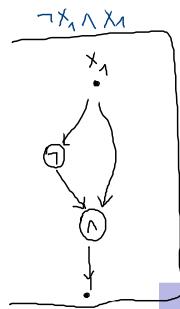
Application

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

The W-hierarchy – Boolean circuits



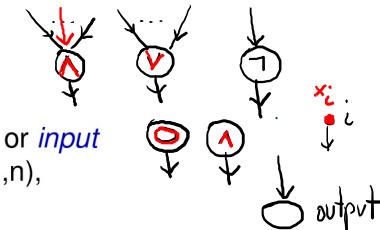
depth 3
Weft 0



depth 3
Weft 0

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 $\mathcal{C}(\cdot) : \{0, 1\}^n \rightarrow \{0, 1\}$ (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 $x = \langle x_1, \dots, x_n \rangle \in \{0, 1\}^*$ 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 .

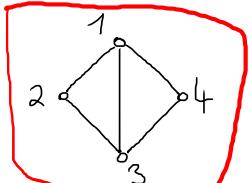
not satisfiable

p DOMINATING SET



$\forall x \in V$
 $(x \in S \vee \exists y \in V$

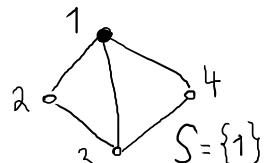
$(E(x,y) \wedge S(y))$)



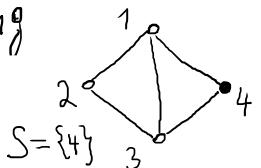
G

G has dom. set S
of k elements

dominating
set

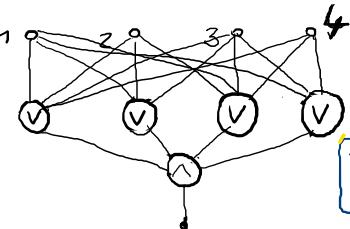


no dominating
set



$e(G)$

\cap
 $CIRC_{3,2}$

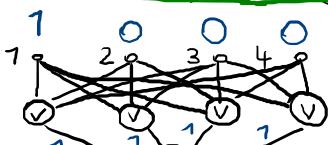


$\text{weft}(e(G)) = 2$
 $\text{depth}(e(G)) = 3$

every vertex should get
an input from itself or
from a neighbour

$e(G)$ is satisfied
by tuple x_S of weight k

(R_1)

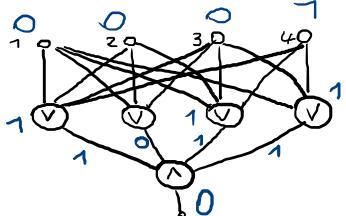


p-DOMINATING-SET

\leq_{fpt}

$p\text{-WSAT}(CIRC_{2,3})$

$\Rightarrow p\text{-DOM-SET} \in W[2]$



W-hierarchy

Definition (with Boolean circuits)

(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 $p\text{-WSAT}(\text{CIRC}_{t,d})$ (with parameter t) for some $d \geq 1$.

$$\text{FPT} \subseteq W[1] \subseteq W[2] \dots$$

- ▶ $p\text{-CLIQUE}$, $p\text{-INDEPENDENT-SET}$ are $W[1]$ -Complete.
- ▶ $p\text{-DOMINATING-SET}$, $p\text{-HITTING-SET}$ are $W[2]$ -Complete.

Hypothesis: $W[1] \neq \text{FPT}$

Proposition

Later

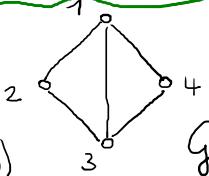
This definition of the W-hierarchy is equivalent to the one here ~~before~~.
~~before~~ It holds, for all $t \geq 1$:

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

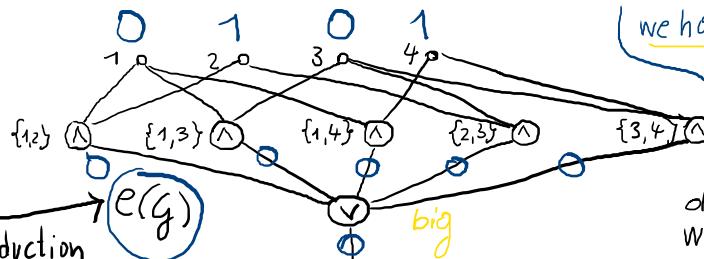
P-INDEPENDENT-SET

\Downarrow

$$\forall x \forall y \\ (E(x,y) \rightarrow S(x) \wedge S(y))$$



G_1
reduction



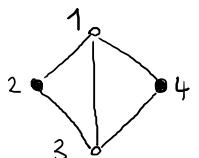
we have to make sure
that no edge gets
input from both
of its vertices

$$\text{depth} = 4 \\ \text{width} = 1$$

$$S = \{2, 4\}$$

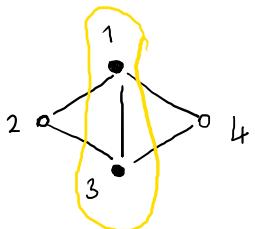
independent set

\Downarrow



$$S = \{1, 3\}$$

not an independent set



Hence:

P-INDEPENDENT-SET

$\leq_{fpt} p\text{-WSAT(CIRC)}$

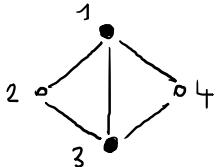
$\Rightarrow p\text{-IND.SET} \in W[0]$

G has a k -element in } \Leftrightarrow $e(G)$ is k -satisfiable

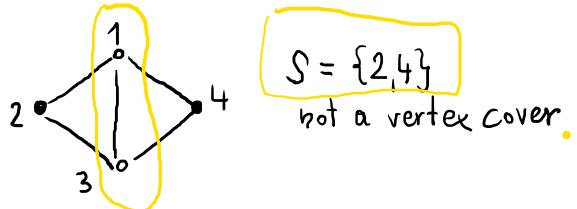
independent set

p-VERTEX-COVER

$$\forall x \forall y (\in(x,y) \rightarrow S(x) \vee S(y))$$



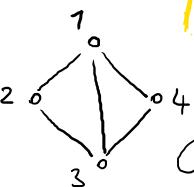
$S = \{1, 3\}$
vertex-cover



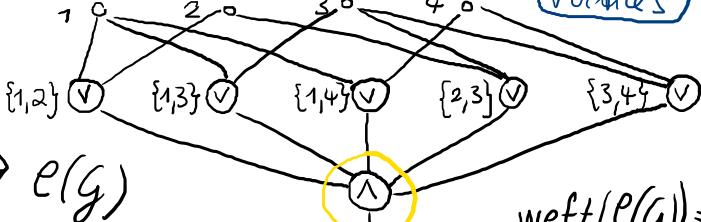
$S = \{2, 4\}$
not a vertex cover.

G is a k -elements
vertex cover

every edge should receive an input from one of its vertices



G



$e(G)$

$$\text{width}(e(G)) = 1$$

$$\text{depth}(e(G)) = 3$$

Hence:

p-VERTEX-COVER

\leq_{fpt}

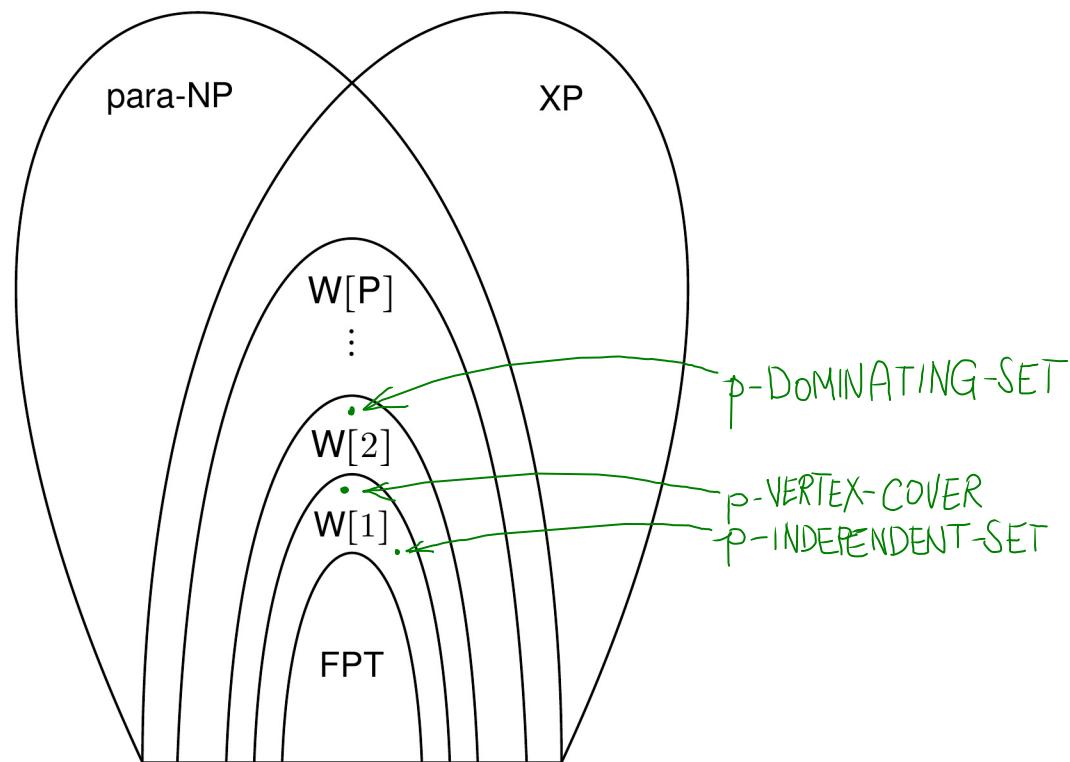
p-WSAT(CIRC_{Q3})



$e(G)$ is k -satisfiable

\Rightarrow p-VERTEX-COVER \leq_{fpt} WSAT

W-Hierarchy



Limited non-determinism (classically)

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

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

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

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

Limited non-determinism (classically)

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

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

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

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

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

Limited non-determinism (classically)

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

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

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

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

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

Fact

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

Theorem (Cai, Chen, 1997)

The following are equivalent:

- (i) $\text{FPT} = \text{W}[\text{P}]$.
- (ii) There is a computable, nondecreasing, unbounded function $\iota : \mathbb{N} \rightarrow \mathbb{N}$ such that $\text{P} = \text{NP}[\iota(n) \cdot \log n]$.

Limited non-determinism (classically)

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

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

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

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

Fact

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

Theorem (Cai, Chen, 1997)

The following are equivalent:

- (i) $\text{FPT} \not\subseteq \text{W[P]}$.
- (ii) There is a computable, nondecreasing, unbounded function $\iota : \mathbb{N} \rightarrow \mathbb{N}$ such that $\text{P} \not\subseteq \text{NP}[\iota(n) \cdot \log n]$.

Showing

$\text{FPT} \not\subseteq \text{W[P]}$

is probably HARDER than

Showing

$\text{P} \not\subseteq \text{NP}$

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.

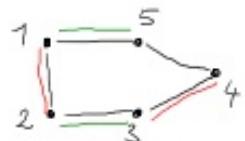
Maximum 2-edge colorable subgraphs

p^* -tw-max-2-edge-colorable-subgraph

Instance: A graph $G = \langle V, E \rangle$.

Parameter: $tw(G)$.

Compute: Maximum number of edges
in a 2-edge colored subgraph of G .



graph G



"line graph"
 $L(G)$

$S \subseteq E$ is 2-colorable



Example [AA & Vahan Mkrtchyan]

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

We find an MSO_2 -formula $\varphi_{\text{2ec}}(S)$ such that $(A_{\text{JHG}} \models \varphi(S))$

$$\varphi_{\text{2ec}}(X) := \forall e (X(e) \rightarrow \text{EDGE}(e))$$

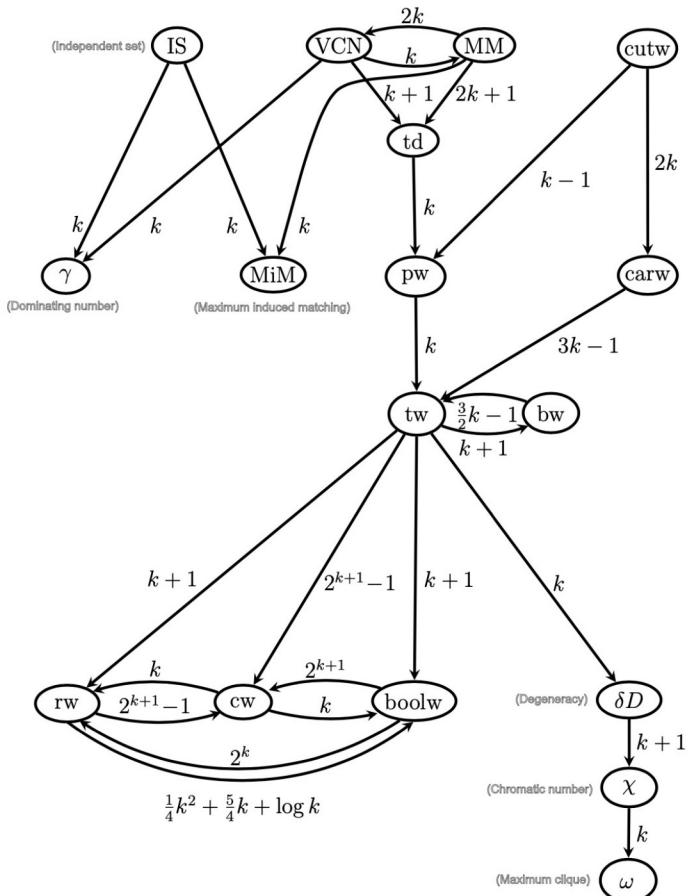
$$\begin{aligned} \text{edge-set-partition } (C_1, C_2, X) := & \exists C_1 \exists C_2 [(\text{edge-set-partition } (C_1, C_2, X) \\ & \wedge \forall e_1 \forall e_2 (\text{EDGE}(e_1) \wedge \text{EDGE}(e_2) \wedge \neg e_1 = e_2 \\ & \wedge \exists v (\text{INC}(v, e_1), \text{INC}(v, e_2)) \\ & \rightarrow \neg(C_1(e_1) \wedge C_1(e_2)) \\ & \wedge \neg(C_2(e_1) \wedge C_2(e_2)))] \\ & \wedge \forall x (\text{EDGE}(x) \rightarrow (X(x) \leftrightarrow C_1(x) \vee C_2(x))) \\ & \wedge \neg(C_1(x) \wedge C_2(x)) \end{aligned}$$

Computably boundedness between notions of width

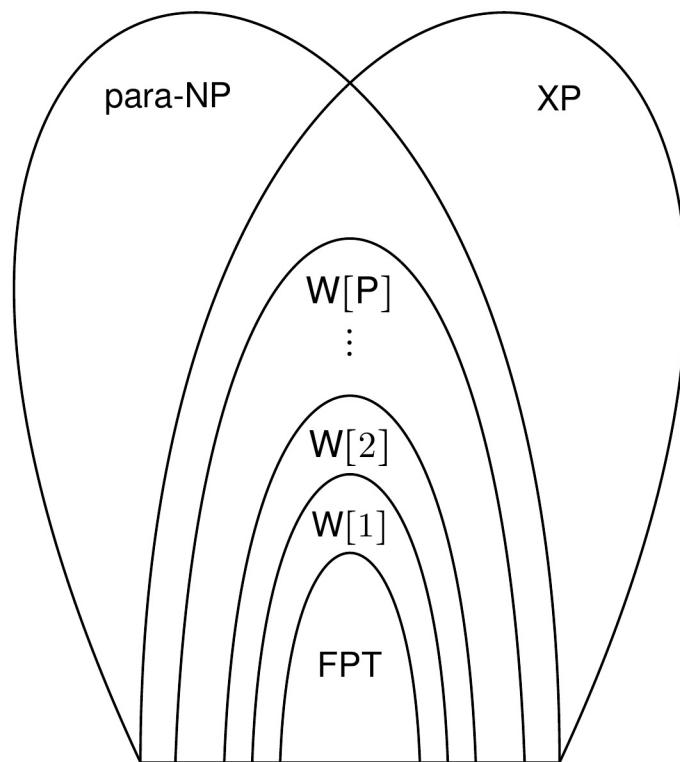
(from Sasák, [6])

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

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



W-Hierarchy



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 $Q_1, \dots, Q_k \in \{\exists, \forall\}$

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 $\mathcal{A} \models \varphi(\bar{S})$.

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

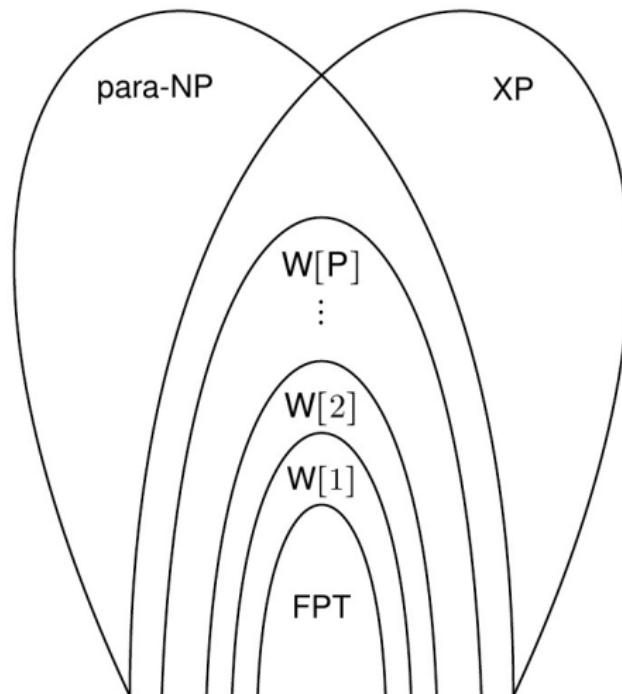
WD_{φ}

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

W-Hierarchy



W-Hierarchy

$p\text{-WD}_\varphi$ (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 $\mathcal{A} \models \varphi(S)$.

$p\text{-WD-}\Phi$: the class of all problems $p\text{-WD-}\varphi$ 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**.

W-Hierarchy

$p\text{-WD}_\varphi$ (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 $\mathcal{A} \models \varphi(S)$.

$p\text{-WD-}\Phi$: the class of all problems $p\text{-WD-}\varphi$ 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

- ▶ $p\text{-CLIQUE} \in W[1]$.
- ▶ $p\text{-DOMINATING-SET} \in W[2]$.
- ▶ $p\text{-HITTING-SET} \in W[2]$.

W-Hierarchy

$p\text{-WD}_\varphi$ (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 $\mathcal{A} \models \varphi(S)$.

$p\text{-WD-}\Phi$: the class of all problems $p\text{-WD-}\varphi$ 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

- $p\text{-CLIQUE} \in W[1]$.

W-Hierarchy

$p\text{-WD}_\varphi$ (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 $\mathcal{A} \models \varphi(S)$.

$p\text{-WD-}\Phi$: the class of all problems $p\text{-WD-}\varphi$ 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

- ▶ $p\text{-DOMINATING-SET} \in W[2]$.

W-Hierarchy

$p\text{-WD}_\varphi$ (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 $\mathcal{A} \models \varphi(S)$.

$p\text{-WD-}\Phi$: the class of all problems $p\text{-WD-}\varphi$ 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

- ▶ $p\text{-HITTING-SET} \in W[2]$.

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 $p\text{-WSAT}(\text{CIRC}_{t,d})$ (with parameter t) for some $d \geq 1$.

$\text{FPT} \subseteq W[1] \subseteq W[2] \dots$

- ▶ $p\text{-CLIQUE}$, $p\text{-INDEPENDENT-SET}$ are $W[1]$ -Complete.
- ▶ $p\text{-DOMINATING-SET}$, $p\text{-HITTING-SET}$ are $W[2]$ -Complete.

Hypothesis: $W[1] \neq \text{FPT}$

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 $p\text{-WSAT}(\text{CIRC}_{t,d})$ (with parameter t) for some $d \geq 1$.

$\text{FPT} \subseteq W[1] \subseteq W[2] \dots$

- ▶ $p\text{-CLIQUE}$, $p\text{-INDEPENDENT-SET}$ are $W[1]$ -Complete.
- ▶ $p\text{-DOMINATING-SET}$, $p\text{-HITTING-SET}$ are $W[2]$ -Complete.

Hypothesis: $W[1] \neq \text{FPT}$

Proposition

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

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

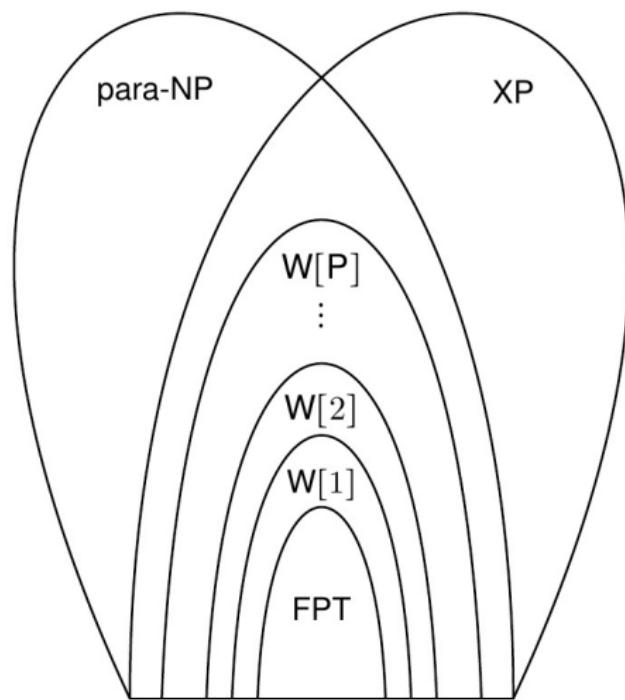
W-Hierarchy (properties)

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

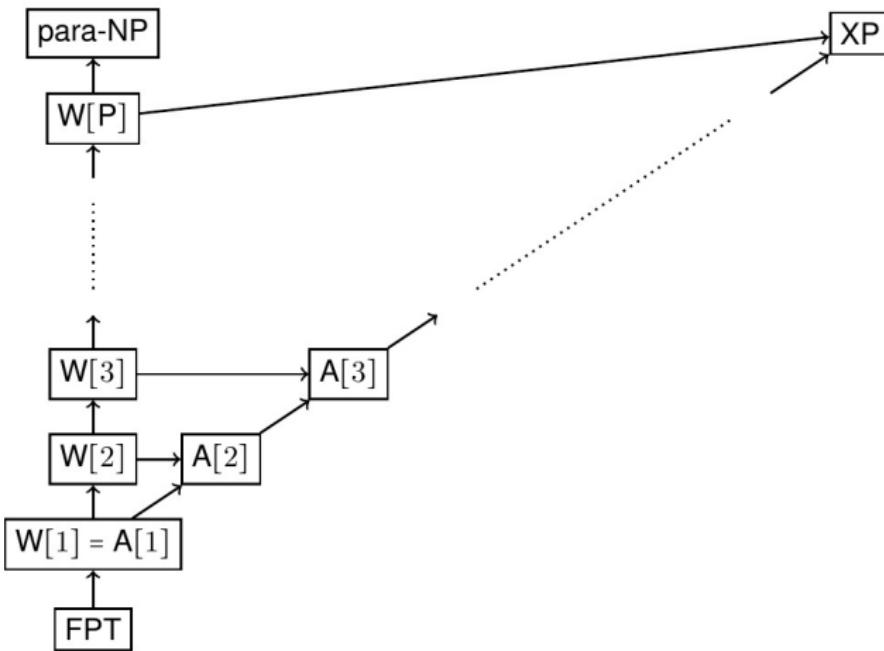
Theorems

- T1. $p\text{-WD-FO} \subseteq W[P]$, and hence $W[t] \subseteq W[P]$ 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. $W[t] = [p\text{-WD-}\Sigma_{t+1}]^{\text{fpt}}$ 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:

$p\text{-MC}(\Phi)$

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)

$A[t] := [p\text{-MC}(\Sigma_t)]^{\text{fpt}}$, 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:

$p\text{-MC}(\Phi)$

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)

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

Examples

- ▶ $p\text{-CLIQUE} \in A[1]$.
- ▶ $p\text{-DOMINATING-SET} \in A[2]$.

A-Hierarchy (definition and examples 3,4)

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

$p\text{-MC}(\Phi)$

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)

$A[t] := [p\text{-MC}(\Sigma_t)]^{\text{fpt}}$, 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:

$p\text{-MC}(\Phi)$

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)

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

Examples

- ▶ $p\text{-HITTING-SET} \in A[2]$.
- ▶ $p\text{-SUBGRAPH-ISOMORPHISM} \in A[1]$.

A-Hierarchy (example 5)

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

$p\text{-MC}(\Phi)$

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)

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

Examples

- ▶ $p\text{-SUBGRAPH-ISOMORPHISM} \in A[1]$.

$p\text{-SUBGRAPH-ISOMORPHISM}$

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:

$p\text{-MC}(\Phi)$

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)

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

Examples

- $p\text{-VERTEX-DELETION} \in A[2]$.

$p\text{-VERTEX-DELETION}$

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:

$p\text{-MC}(\Phi)$

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)

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

Examples

- ▶ $p\text{-CLIQUE-DOMINATING-SET} \in A[2]$.

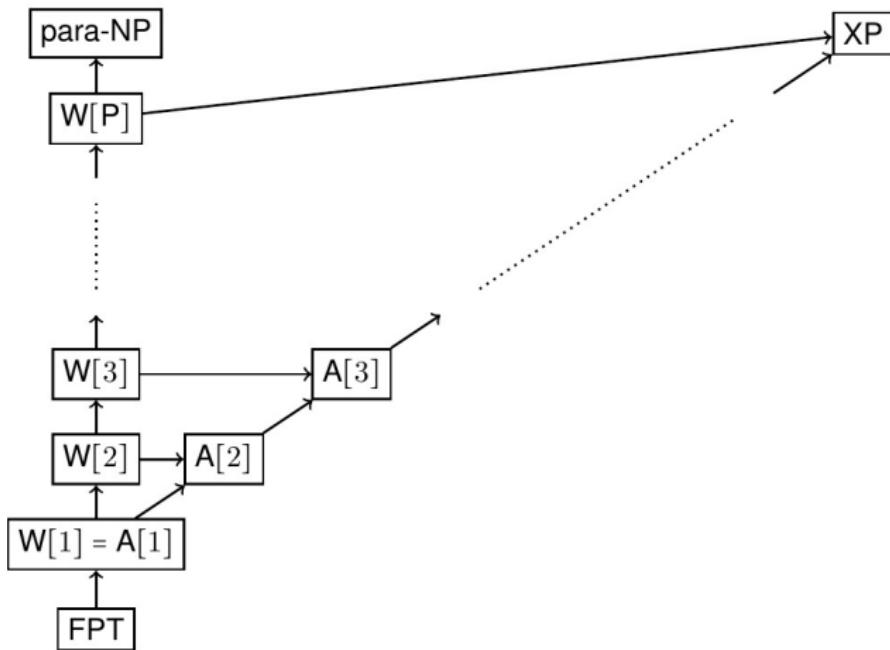
$p\text{-CLIQUE-DOMINATING-SET}$

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.

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



A-Hierarchy (properties)

Theorems

- T1. $A[1] \subseteq W[P]$.
- T2. $W[t] \subseteq A[t]$, for all $t \in \mathbb{N}$.

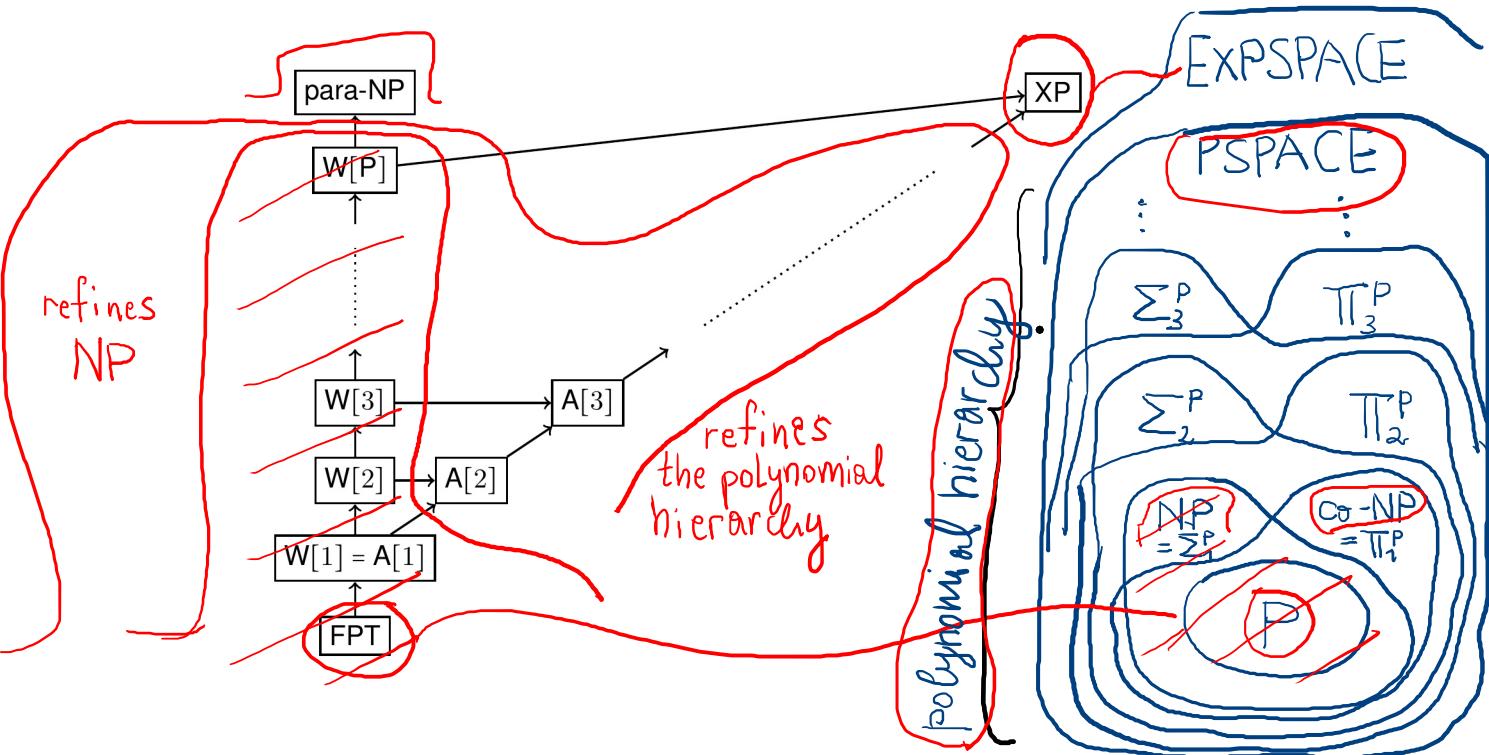
- ▶ **Unlikely:** $A[t] \subseteq W[t]$, for $t > 1$.

Reason:

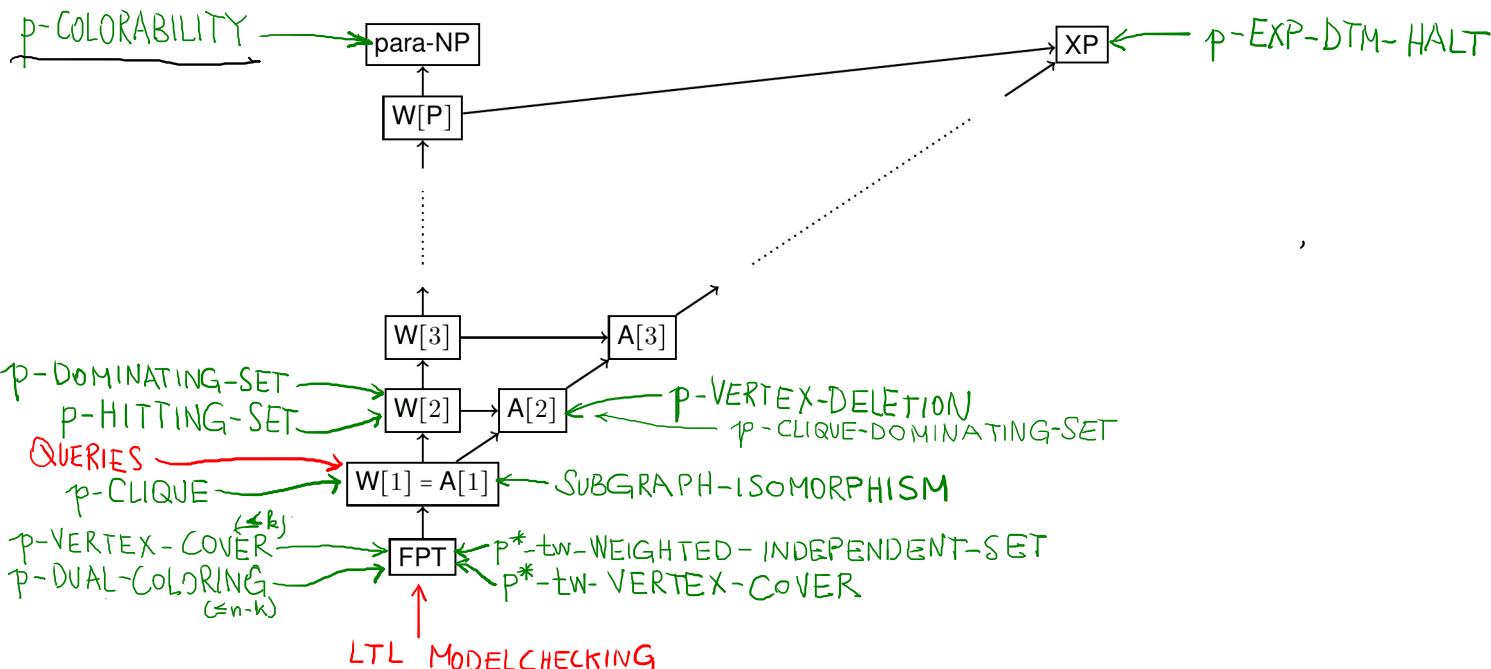
- ▶ the A-hierarchy are parameterizations of problems that are complete for the levels of the polynomial hierarchy
 - ▶ the W-hierarchy is a refinement of NP in parameterized complexity
- ▶ **Unlikely:** $[p\text{-MC(FO)}]^{\text{fpt}} = \bigcup_{i=1}^{\infty} A[i]$,
contrasting with: $[p\text{-WD-FO}]^{\text{fpt}}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

A-Hierarchy

classical hierarchy



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



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.

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.



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 $n = \|\mathcal{K}\|$.

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.

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 $n = \|\mathcal{K}\|$.

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 $\in \text{W[1]}$ (= 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 $\in O(k \cdot 2^{2k} \cdot n) \in \text{FPT}$ 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 10 10.30 – 12.30 | Tuesday, June 11 | Wednesday, June 12 10.30 – 12.30 | Thursday, June 13 | Friday, June 14 |
|--|--|--|-------------------|---|
| Introduction & basic FPT results motivation for FPT kernelization, Crown Lemma, Sunflower Lemma | | Algorithmic Meta-Theorems 1st-order logic, monadic 2nd-order logic, FPT-results by Courcelle's Theorems for tree and clique-width | | |
| | GDA | | GDA | GDA |
| <i>Algorithmic Techniques</i> | | <i>Formal-Method & Algorithmic Techniques</i> | | |
| | 14.30 – 16.30 | | | 14.30 – 16.30 |
| | Notions of bounded graph width path-, tree-, clique width, FPT-results by dynamic programming, transferring FPT results betw. widths | | | FPT-Intractability Classes & Hierarchies motivation for FP-intractability results, FPT-reductions, class XP (slicewise polynomial), W- and A-Hierarchies, placing problems on these hierarchies |
| | | GDA | GDA | |

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 $Q_2 \in \text{FPT}$ implies $Q_1 \in \text{FPT}$.
2. Find the idea for:
 $p\text{-DOMINATING-SET} \equiv_{\text{fpt}} p\text{-HITTING-SET}$.
- 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.

Fixed-parameter tractable reductions

Examples

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

Non-Example

- ▶ For graphs $\mathcal{G} = \langle V, E \rangle$, and sets $X \subseteq V$:
 X is independent set of $\mathcal{G} \iff V \setminus X$ is a vertex cover of \mathcal{G}
yields a **polynomial reduction** between $p\text{-INDEPENDENT-SET}$ and $p\text{-VERTEX-COVER}$, but **does not yield an fpt-reduction**.

p-DOMINATING-SET

Instance: Graph $G = \langle V, E \rangle$, $k \in \mathbb{N}$

Parameter: k

Question: Does G have a dominating set of size $\leq k$

p-DOMINATING-SET

\equiv_{fpt}

\leq_{fpt}

\geq_{fpt}

p-HITTING-SET

Instance: U, S , $B \subseteq P(U)$, $k \in \mathbb{N}$

Parameter: $k \in \mathbb{N}$

Question: Is there a hitting set for B of size $\leq k$

p-HITTING-SET

p-DOMINATING-SET

Instance: Graph $G = \langle V, E \rangle$, $k \in \mathbb{N}$

Parameter: k

Question: Does G have a dominating set of size $\leq k$

p-DOMINATING-SET

$$\begin{array}{c} \equiv_{\text{fpt}} \\ \Leftarrow_{\text{fpt}} \\ \Rightarrow_{\text{fpt}} \end{array}$$

p-HITTING SET

Instance: Used, $B \subseteq P(U)$, $k \in \mathbb{N}$

Parameter: $k \in \mathbb{N}$

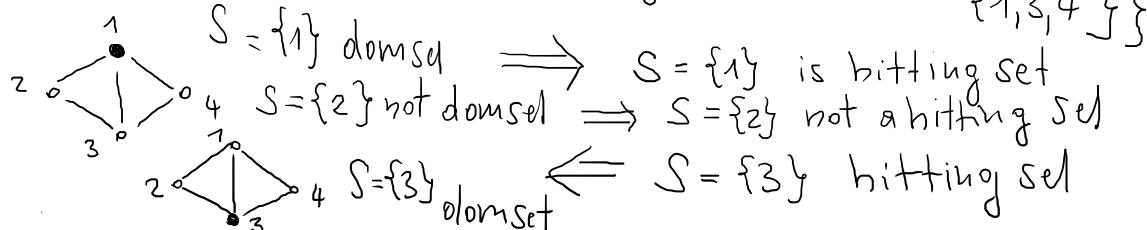
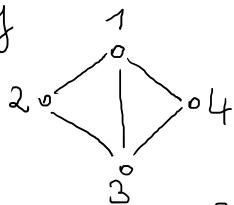
Question: Is there a hitting set for $\cup B$ of size $\leq k$

p-HITTING-SET

$$N(v) := \{w \mid \{v, w\} \in E\}$$

$$\langle G, k \rangle \longleftrightarrow \langle B_G := \{ \{v\} \cup N(v) \mid v \in V \}, k \rangle$$

G



p-DOMINATING-SET

Instance: Graph $G = \langle V, E \rangle$, $k \in \mathbb{N}$

Parameter: k

Question: Does G have a dominating set of size $\leq k$

p-DOMINATING-SET

\equiv_{fpt}
 \leq_{fpt}
 \geq_{fpt}

p-HITTING-SET

Instance: Sets, $\mathcal{B} \subseteq \mathcal{P}(U)$, $k \in \mathbb{N}$

Parameter: $k \in \mathbb{N}$

Question: Is there a hitting set for \mathcal{B} of size $\leq k$

p-HITTING-SET

$$G_{\mathcal{B}} := \langle U, E_{\mathcal{B}} \rangle \quad \xleftarrow{\times} \quad \langle U, \mathcal{B}, k \rangle$$

$$E_{\mathcal{B}} = \{ \{u, v\} \mid \exists B \in \mathcal{B} (u, v \in B \wedge u \neq v) \}$$

$$\langle G_{\mathcal{B}} := \langle U \cup \mathcal{B}, E_1 \cup E_2 \rangle \rangle^k \quad \xleftarrow{\hspace{1cm}} \quad \langle U, \mathcal{B}, k \rangle$$

$$E_1 := \{ \{u, B\} \mid u \in U, u \in B \in \mathcal{B} \}$$

$$E_2 := \{ \{v, w\} \mid v, w \in U, v \neq w \}$$