An introduction to Parameterized Complexity Theory

Martín Ugarte

May 15, 2012

A motivation to parametrize

Standard Complexity Theory does not distinguish between the distinct parts of the input. It categorizes problems in terms of the length of the complete input.

- To know if a PL sentence is satisfiable is said to be a hard problem. And it is, but:

  ¿What if the sentence has very few variables?

A motivation to parametrize (contd.)

Consider the problem of knowing if a graph G is isomorphic to a subgraph of a graph H.
This problem is usually defined as:

$$\{(G, H) \mid G \simeq H' \text{ for some } H' \subseteq H\}$$

Here the input is (G, H).

What if the size of G is considerably smaller than H?

Basic notions

From now on we assume Σ to be a fixed alphabet.

Definition

A parametrization of Σ^* is a polynomial-time computable function $\kappa : \Sigma^* \to \mathbb{N}$.

Definition

A parameterized language is a pair (L, κ), where $L \subseteq \Sigma^*$ and κ is a parametrization.

Basic notions (contd.)

Definition

Let κ be a parametrization.
A Turing Machine M over Σ is said to be fixed-parameter tractable with respect to κ (κ-fpt) if there is a polynomial p and a computable function $f : \mathbb{N} \to \mathbb{N}$ such that the running time of M on input x is at most $f(\kappa(x)) \cdot p(|x|)$.

Definition

A parameterized language (L, κ) is said to be fixed-parameter tractable if there is a κ-fpt Turing Machine that decides L. The class of all fixed-parameter tractable languages is denoted by FPT.

Example

Let $\kappa : \Sigma^* \to \mathbb{N}$ be defined as:

$$\kappa(x) = \begin{cases} |var(x)| & \text{if } x \text{ represents a LP sentence} \\ 1 & \text{otherwise} \end{cases}$$

As κ is polynomial time computable it is clear that it is a parametrization.
Then (SAT, κ) is a parameterized language.

Is (SAT, κ) fixed-parameter tractable?

Are this notions interesting?

Theorem

For every decidable language L, there is a parametrization κ for which (L, κ) is fpt.

Proof: Take $\kappa(x) = |x|$ ■

Why is fixed-parameter tractability an interesting notion?

- Because the parametrization is considered as part of the problem.

Parameterized languages slices

Definition

Let (L, κ) be a parameterized language. The ℓ^{th} slice of (L, κ) is the language

$$\{x \mid x \in L \text{ and } \kappa(x) = \ell\}$$

Theorem

If a parameterized language is fpt then for every $n \in \mathbb{N}$ its n^{th} slice is in `Ptime`.

Example

Define k-COLORABILITY as the language

$$\{(G, k) \mid G \text{ is a } k\text{-colorable graph}\}$$

and the parametrization

$$\kappa(x) = \begin{cases} k \text{ if } x = (G, k) \text{ for some graph } G \text{ and } k \in \mathbb{N} \\ 1 \text{ otherwise.} \end{cases}$$

Define p-COLORABILITY as $(k\text{-COLORABILITY}, \kappa)$

Exercise

Prove that if p-COLORABILITY is in FPT *then* P=NP.

Reductions

Definition

Let (L, κ) and (L', κ') be parameterized languages. An fpt reduction from (L, κ) to (L', κ') is a mapping $R : \Sigma^* \to \Sigma^*$ such that:

1. For all $x \in \Sigma^*$ we have $(x \in L \Leftrightarrow R(x) \in L')$
2. R is computable by an κ-fpt Turing Machine. That is, there is a computable function f and a polynomial p such that $R(x)$ is computable in time $f(\kappa(x)) \cdot p(|x|)$.
3. There is a computable function $g : \mathbb{N} \to \mathbb{N}$ such that $\kappa'(R(x)) \leq g(\kappa(x))$ for all $x \in \Sigma^*$.

Reductions (contd.)

Why is the third condition important?
(There is a computable function $g : \mathbb{N} \to \mathbb{N}$ such that $\kappa'(R(x)) \leq g(\kappa(x))$ for all $x \in \Sigma^*$.)

Let $\kappa_1 : \Sigma^* \to \mathbb{N}$ defined as $\kappa_1(x) = 1$ for every $x \in \Sigma^*$.
Let $\kappa_\ell : \Sigma^* \to \mathbb{N}$ defined as $\kappa_\ell(x) = |x|$ for every $x \in \Sigma^*$.

Should (SAT, κ_1) be fpt-reducible to $(\text{SAT}, \kappa_\ell)$?

We would have

$$x = \kappa_\ell(R(x)) \leq g(\kappa_1(x)) = g(1),$$

a contradiction.

Reductions (contd.)

Lemma

FPT *is closed under fpt reductions.*

Proof

Let R be an fpt reduction from (L, κ) to (L', κ') computable in time $h(\kappa(x)) \cdot q(|x|)$, with $\kappa'(R(x)) \leq g(\kappa(x))$.
Let M' be an κ'-fpt-Turing Machine deciding L' in time $f'(\kappa'(x)) \cdot p'(|x|)$. We can assume f' to be non-decreasing.
Then L can be decided by computing $R(x) = x'$ and then deciding if $x' \in L'$ by running M'. This takes at most time

$$h(\kappa(x)) \cdot q(|x|) + f'(\kappa'(x')) \cdot p'(|x'|)$$
$$\leq h(\kappa(x)) \cdot q(|x|) + f'(g(\kappa(x))) \cdot p'(h(\kappa(x)) \cdot q(|x|))$$

Reductions (contd.)

We write $(L, \kappa) \leq^{\text{fpt}} (L', \kappa')$ if there is an fpt reduction from (L, κ) to (L, κ').

Lemma

For every parameterized language $(L, \kappa) \in FPT$, every nontrivial language L' and every parametrization κ', it is the case that $(L, \kappa) \leq^{fpt} (L', \kappa')$.

Example

p-Clique $\leq^{\text{fpt}}$ p-Independent-set. For both problems, an instance is a pair (G, k) and the parameter is k.

$$R(x) = \begin{cases} (\overline{G}, k) \text{ if } x = (G, k) \\ (K_1, 2) \text{ otherwise} \end{cases}$$

Reductions (contd.)

Not every polynomial-time reduction is an fpt reduction. Moreover, polynomial-time (or logspace) reductions and fpt reductions are non comparable.

Exercise

Show that there are parameterized languages (L, κ) and (L', κ') such that

$$(L, \kappa) <^{fpt} (L', \kappa') \qquad \text{and} \qquad L' <^{\text{Ptime}} L$$

para-NP

FPT can be seen as the analogue to Ptime for parameterized languages. Now we introduce non-determinism.

Definition

Given a parametrization κ, a non-deterministic Turing Machine M is said to be κ-fpt if there is a computable function f and a polynomial p such that for every $x \in L(M)$, there is an accepting run of M with input x that takes time at most

$$f(\kappa(x)) \cdot p(|x|)$$

Definition (para-NP)

para-NP is the class of parameterized languages (L, κ) for which there is a non-deterministic κ-fpt Turing Machine deciding L.

para-NP(contd.)

Recall the parameterized language p-Colorability. This language is clearly in para-NP as $k -$ Colorability is in NP. But we already proved p-Colorability $\in FPT \Rightarrow$ P = NP. It can be shown that

$$\text{FPT} = \text{para-NP} \Leftrightarrow \text{Ptime} = \text{NP}$$

XP

Definition

XP_{nu} (non-uniform XP) is the class of all parameterized problems with every slice in Ptime.

Lemma

If Ptime$\neq$NP *then* para-NP$\neq \mathrm{XP}_{nu}$.

Proof

If para-NP$= \mathrm{XP}_{nu}$ *then* $p-$Colorability $\in \mathrm{XP}_{nu}$. *Hence* $3-$Colorability $\in$ *Ptime*.

XP (contd.)

Why is this class called non-uniform?
Consider a non-decidable language $L \subseteq \{1\}^*$. Then the parameterized language (L, κ_ℓ) (where $\kappa_\ell(x) = |x|$) is in XP_{nu}.

Definition

A parameterized problem (L, κ) belongs to XP if there is a computable function $f : \mathbb{N} \to \mathbb{N}$ and a Turing Machine M that on input x it decides if $x \in L$ in at most

$$|x|^{f(\kappa(x))} + f(\kappa(x))$$

setps.

Why is this a "uniform" notion?

Complete problems for XP and PARA-NP

Theorem

$(CNF\text{-}SAT, \kappa)$ where $\kappa(\varphi)$ is the maximum amount of literals in a clause of φ, is PARA-NP*-complete.*

Definition

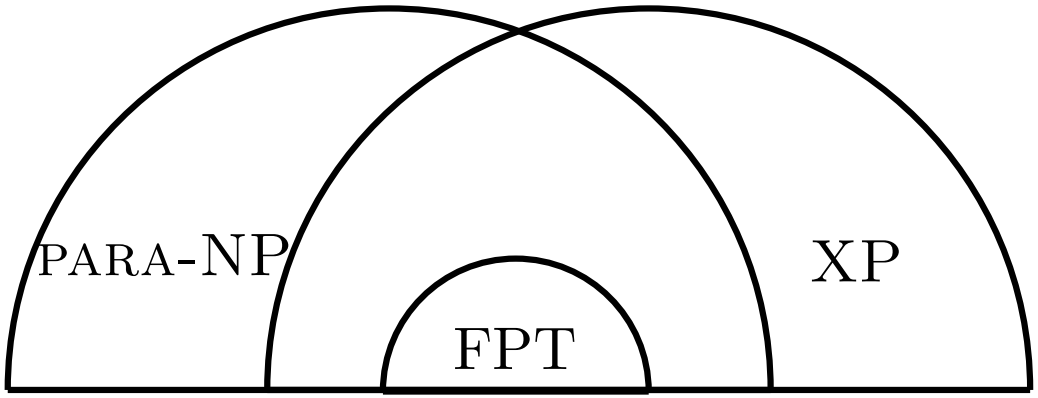
p-EXP-DTM-Halt is the parametrized language containing all tuples $((M, n, k), \kappa)$ where $\kappa((M, n, k)) = k$, M is a turing machine, n is a natural number in unary notation, k is a natural number, and M accepts the empty string in at most n^k steps.

Theorem

p-EXP-DTM-Halt is XP*-complete.*

What we have so far

Relations between the complexity classes we have so far are illustrated by the next figure:



New complexity classes

Some natural problems are not captured by the complexity classes defined so far. Basically FPT is to strict and para-NP and XP are to permissive.
Nobody has been able to show fpt-completeness for any of these three classes for the problem p-Clique.
Moreover, people believe that the above problem is not fixed-parameter tractable. A new family of complexity classes is born.

The W hierarchy

Next we define the W hierarchy. To this end, we introduce some notions in logics.

Definition

Π_t (Σ_t respectively) is the set of all formulae φ such that

$$\varphi = Q_1 \overline{x_1} Q_2 \overline{x_2} \cdots Q_t \overline{x_t}\ \psi(\overline{x_1}, \overline{x_2}, \ldots, \overline{x_t})$$

where $Q_i = \forall$ if i is odd (respectively even) and $Q_i = \exists$ if i is even (respectively odd) and ψ is a quantifier-free sentence.

The W hierarchy (contd.)

Let $\varphi(X)$ be a first-order formula with a free relation variable X of arity s. Define $L(\varphi(X))$ as the language of all pairs $(\mathfrak{A}, k)$ where $\mathfrak{A}$ is a first-order structure for which there is a relation $S \subseteq dom(\mathfrak{A})^s$ of cardinality k such that $\mathfrak{A} \models \varphi(S)$.

Definition

Given a first-order formula $\varphi(X)$ with a free relation variable X, the parameterized problem p-WD$_\varphi$ is defined as $(L(\varphi(X)), \kappa)$ where $\kappa((\mathfrak{A}, k)) = k$.

Definition

For a class Φ of first-order formulas with a free relation variable, p-WD-Φ denotes the set $\bigcup_{\varphi \in \Phi}\{\text{p-WD}_\varphi\}$

The W hierarchy (contd.)

Definition

$\Pi_t(X)$ is the set of formulas of the form

$$\varphi = Q_1\overline{x_1}Q_2\overline{x_2}\cdots Q_t\overline{x_t}\ \psi(\overline{x_1},\overline{x_2},\ldots,\overline{x_t})$$

where Q_i is defined as in Π_t and ψ mentions the additional relation variable X.

Definition (W hierarchy)

Let $t \in \mathbb{N}$. $W[t]$ is the set of problems fpt-reducibles to some problem in P-WD-$\Pi_t(X)$. This is denoted by $[\text{P-WD-}\Pi_t(X)]^{\text{fpt}}$

$W[1]$ and p-Clique

It is clear that the problem p-Clique is in XP. Can we know that it is not in FPT? Not certainly, but it is believed it is not:

Theorem

p-Clique *is fpt-complete for* $W[1]$.

Exercise

Show that p-Clique $\in W[1]$.

$$\varphi(X) = \forall x \forall y (X(x) \wedge X(y) \wedge x \neq y \rightarrow E(X, y))$$

$W[2]$ and p-Dominating-Set

A natural complete problem for $W[2]$ is p-Dominating-Set.

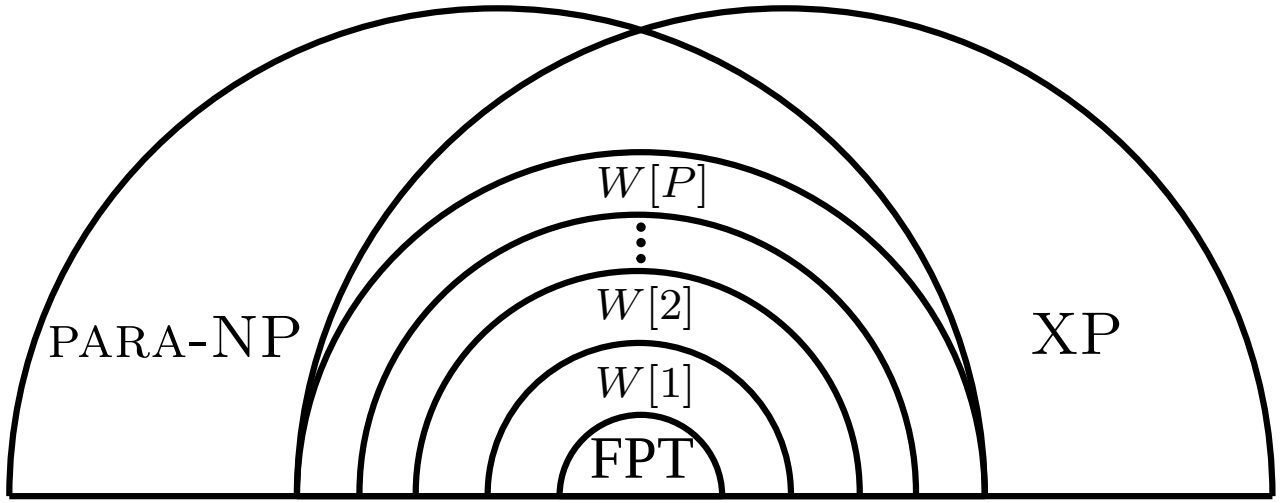
Exercise

Show that p-Dominating-Set $\in W[2]$.

$$\varphi(X) = \forall x \exists y (\neg X(x) \to (E(x,y) \wedge X(y)))$$

Known classes containments

The known containments for the already seen classes is illustrated by the next figure:



¿Questions?