Advanced Algorithmics

Strategies for Tackling Hard Problems

Sebastian Wild

Markus Nebel

Lecture 3

2017-04-27

3 Fixed-parameter Tractability and Efficient Exponential Algorithms

Idea: Refine complexity analysis in two dimensions (size n length of encoding, parameter k) and investigate influence of various parameters on time complexity.

Hope: We find parameter for which – if its value is smaller the problem can be solved efficiently
In practice also values of parameter is small.

A first examples: Drosophila of complexity theory CNF-SAT

- size of clause (# literals per clause)

$k = 3$ 3SAT $\mathcal{NP}$-complete

$k = 2$ 2SAT $\mathcal{P}$

- #clauses m CNF-SAT 1.24^m
- #variables n $O(2^n|x|)$ brute force 1.49^n
- #literals in φ 1.08^ℓ
- weight of formula : #1's in a satisfying assignment
- structure of formula : graph that reflects interplay of literals

Definition 3.1 (Parametrization)

Let Σ a (finite) alphabet. A *parametrization* (of $\Sigma^\star$) is a mapping $\kappa : \Sigma^\star \to \mathbb{N}$ that is poly-time computable. ◀

Definition 3.2 (Parametrized problem)

A *parameterized (decision) problem* is a pair (L, κ) of a language $L \subset \Sigma^\star$ and a parametrization κ of $\Sigma^\star$. ◀

Definition 3.3 (Canonical Parametrizations)

We can often specify a parametrized problem conveniently as a language of *pairs* $L \subset \Sigma^\star \times \mathbb{N}$ with

$$(x, k) \in L \;\wedge\; (x, k') \in L \quad \to \quad k = k'$$

using the *canonical parametrization* $\kappa(x, k) = k$. ◀

Examples

As before: Typically leave encoding implicit.
Naming convention: Add prefix p-SAT.

Definition 3.4 (p-SAT)

Given: formula boolean ϕ (same as before)
Parameter: number of variables
Question: Is there a satisfying assignment $v : [n] \to \{0, 1\}$? ◀

Definition 3.5 (p-Clique)

Given: graph $G = (V, E)$ and $k \in \mathbb{N}$
Parameter: k
Question: $\exists V' \subset V \ : \ |V'| \geq k \ \wedge \ \forall u, v \in V' : \{u, v\} \in E$? ◀

Definition 3.6 (Canonically Parametrized Optimization Problems)

Let $U = (\Sigma_I, \Sigma_O, L, L_I, M, cost, goal)$ be an optimization problem.
Then p-U denotes the *(canonically) parameterized (decision) problem* given by the threshold problem $Lang_U$. ◀

Recall: $Lang_U$ is the set of pairs (x, k) of all instances $x \in L_I$ that are weakly "better" than k.

Examples:

- p-Clique
- p-Vertex-Cover
- p-Graph-Coloring
- …

Naming convention for other parameters:
p-*clause*-CNF-SAT: CNF-SAT with parameter "number of *clauses*"

Examples of Running Times

only simple brute-force methods here

- p-SAT : n length k variables

  2^k candidates , $O(n)$

  $\to O(2^k \cdot n)$ $k = O(\log n)$ poly-time

- p-Clique : n vertices m edges , k

  $\binom{n}{k}$ candidates $O(k^2)$ checks

  $\to O(n^k k^2)$

- p-Vertex Cover $\binom{n}{k}$ $O(m)$ check $\to O(n^k m)$

$\binom{n}{k} = \frac{n^{\underline{k}}}{k!} \leq n^k$

$k = O(1)$ poly-time

- p-Graph-Coloring n, m k colors

  $O(k^n \cdot m)$ $k=3$ NP-complete

3.1 Fixed-Parameter Tractability

Definition 3.7 (fpt-algorithm)
Let κ be a parametrization for $\Sigma^\star$.
A (deterministic) algorithm A (with input alphabet Σ) is a *fixed-parameter tractable algorithm (fpt-algorithm)* w.r.t. κ if its running time on $x \in \Sigma^\star$ with $\kappa(x) = k$ is at most

$$f(k) \cdot p(|x|) \quad = \quad \mathcal{O}\big(f(k) \cdot |x|^c\big) \qquad |x|^{O(1)} \cdot f(k)$$

where p is a polynomial of degree c and f is an arbitrary computable function. ◀

Definition 3.8 (FPT)
A parametrized problem (L, κ) is *fixed-parameter tractable* if there is an fpt-algorithm that decides it.
The complexity class of all such problems is denoted by $\mathcal{FPT}$. ◀

Intuitively, $\mathcal{FPT}$ plays the role of $\mathcal{P}$.

Theorem 3.9 (p-variables-SAT is FPT)

p-variables-SAT $\in \mathcal{FPT}$.

Proof: brute-force has running time $O(2^k n)$ $\square$

$f(k) = 2^k$ $p(n) = n$

(L, x_{size}) $x_{size}(x) = |x|$

$L \in NP$ A $f(k) \cdot (\dots)$

…but #variables not usually small

Theorem 3.10 (k never decreases → FPT)

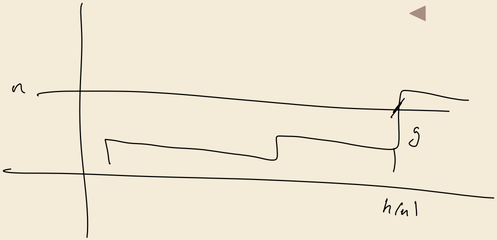
Let $g : \mathbb{N} \to \mathbb{N}$ weakly increasing, unbounded and computable, and κ a parametrization with

$$\forall x \in \Sigma^\star : \kappa(x) \geq g(|x|).$$

Then $(L, \kappa) \in \mathcal{FPT}$ for *any* decidable L.

g weakly increasing: $\quad n \leq m \;\rightarrow\; g(n) \leq g(m)$

g unbounded: $\quad \forall t\; \exists n \;:\; g(n) \geq t$



Proof: L decidable $\Rightarrow$ $\exists T$: $x \overset{?}{\in} L$ decidable in $\leq T(|x|)$ steps

can be huge

$T(|x|) \geq |x|$

wlog. T weakly increasing

$T(|x|)$ must be hidden in $f(k)$ part of FPT bound

$$h(n) = \begin{cases} \max\{m \in \mathbb{N} : g(m) \leq n\} & n \geq g(1) \\ 1 & \text{otherwise} \end{cases}$$

- g weakly incr & unbounded $\Rightarrow$ h well-defined
- – " $\Rightarrow$ h weakly increasing
- g computabl $\Rightarrow$ h computable
- $h(g(n)) \geq n$

$h(g(|x|)) \geq |x|$

time to decide $x \in L$

$$T(|x|) \underset{T \text{ incr.}}{\leq} T\Big(h(\underbrace{g(|x|)}_{\leq \varkappa(x)})\Big) \leq T(h(\underbrace{\varkappa(x)}_{k})) =: f(k) \qquad \Box$$

How to show $L \notin FPT$?

$\hookrightarrow$ reductions hardness $\rightsquigarrow$ relative statements

3.2 Parametrized Reductions and Hardness

Definition 3.11 (Parametrized Reduction)
Let (L_1, κ_1) and (L_2, κ_2) be two parametrized problems (over alphabets Σ_1 resp. Σ_2).
An *fpt-reduction* *(fpt many-one reduction)* from (L_1, κ_1) to (L_2, κ_2) is a mapping $A : \Sigma_1^\star \to \Sigma_2^\star$ so that for all $x \in \Sigma_1^\star$

1. (equivalence) $x \in L_1 \iff A(x) \in L_2$, $k = \kappa_1(x)$
2. (fpt) A is computable by an fpt-algorithm (w.r.t. to κ_1), and $O(f(k) \cdot |x|^c)$
3. (parameter-preserving) $\kappa_2\big(A(x)\big) \le g\big(\kappa_1(x)\big)$ for a computable function $g : \mathbb{N} \to \mathbb{N}$.

We then write $(L_1, \kappa_1) \le_{fpt} (L_2, \kappa_2)$. $\le_p$

Not all reductions are fpt.

Many reductions from classical complexity theory are not parameter preserving.

Recall:

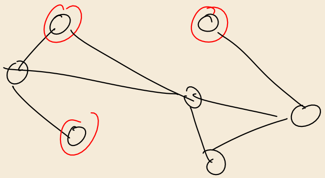
VERTEX-COVER
Given: graph $G = (V, E)$ and $k \in \mathbb{N}$
Question: $\exists V' \subset V \ : \ |V'| \leq k \ \wedge \ \forall \{u, v\} \in E : \big(u \in V' \vee v \in V'\big)$

INDEPENDENT SET
Given: graph $G = (V, E)$ and $k \in \mathbb{N}$
Question: $\exists V' \subset V \ : \ |V'| \geq k \ \wedge \ \forall u, v \in V' : \{u, v\} \notin E$

Independent Set $\leq_p$ Vertex-Cover

$G' = G$ $\qquad \boxed{k' = |V| - k}$

p-Independent-Set $\overset{?}{\leq}_{fpt}$ p-Vertex-Cover



Parametrized NP: Non-deterministic NP

$\mathcal{P}$ corresponds to $\mathcal{FPT}$ … but what is the analogue for $\mathcal{NP}$?

Definition 3.12 (para-NP)

The class *para-$\mathcal{NP}$* consists of all parametrized decision problems that are solved by a *non-deterministic* fpt-algorithm.

Some nice properties:

1. para-$\mathcal{NP}$ is closed under fpt-reductions.
2. $\mathcal{FPT} = \text{para-}\mathcal{NP} \iff \mathcal{P} = \mathcal{NP}$
3. an analogue for *kernalization* in $\mathcal{FPT}$ holds for para-$\mathcal{NP}$ (discussed later)

$\leq_{fpt}$ is transitive

$\rightsquigarrow$ Can define para-$\mathcal{NP}$-hard and para-$\mathcal{NP}$-complete similarly as for $\mathcal{NP}$:

Definition 3.13 (para-NP-hard)

(L, κ) is para-$\mathcal{NP}$-hard if $(L', \kappa') \leq_{fpt} (L, \kappa)$ for all $(L', \kappa') \in \text{para-}\mathcal{NP}$.

. . . is too strict

Theorem 3.14 (para-NP-complete → NP-complete for finite parameter)
Let (L, κ) be a nontrivial ($\emptyset \neq L \neq \Sigma^\star$) parametrized problem that is para-$\mathcal{NP}$-complete.
Then $L_{\leq d} = \{x \in L : \kappa(x) \leq d\}$ is $\mathcal{NP}$-hard. ◀

The converse is essentially also true.

Proof: (L, κ) para-$\mathcal{NP}$-comp.

L' $\mathcal{NP}$-complete $\Rightarrow$ $(L', \kappa_{one}) \in$ para-$\mathcal{NP}$

$\Rightarrow$ $(L', \kappa_{one}) \leq_{fpt} (L, \kappa)$

i.e. A fpt rd. $\leq f(k) \cdot n^c$ ($k=1$)

$\kappa(A(x)) \leq g(\kappa_{one}(x)) = g(1)$ ($\kappa_{one}(x) = 1$)

$\Rightarrow$ A runs in $f(1) \cdot n^c$ = poly-time $\leadsto$ A poly-time reduction from L' to $L_{\leq g(1)}$

$$L_{\leq g(1)} = \{ x \in L : \kappa(x) \leq g(1) \} \Rightarrow NP\text{-hard}$$

$\square$.

This means that only very few problems are para-NP-complete. (p-Graph-Coloring)

But p-Clique, p-Vertex Cover, p-Independent Set cannot be para-NP-complete, but we think there are not in FPT.