Computational Complexity

Lecture 14: Recap and bonus

Ronald de Haan
me@ronalddehaan.eu

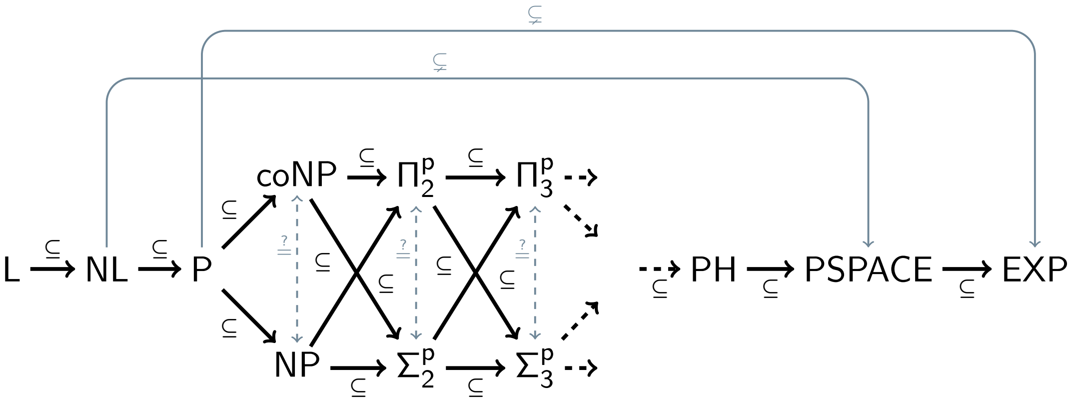
University of Amsterdam

March 19, 2026

Today

- A bird's eye overview of what we covered
- (Possible bonus: quick intro into *parameterized complexity theory*)

An overview of complexity classes



The Cook-Levin Theorem

Theorem (Cook 1971, Levin 1969)

3SAT *is* NP-*complete.*

The Time Hierarchy Theorems

Theorem

If $f, g : \mathbb{N} \to \mathbb{N}$ *are time-constructible functions such that* $f(n) \log f(n)$ *is* $o(g(n))$, *then* $\mathrm{DTIME}(f(n)) \subsetneq \mathrm{DTIME}(g(n))$.

Theorem

If $f, g : \mathbb{N} \to \mathbb{N}$ *are time-constructible functions such that* $f(n+1)$ *is* $o(g(n))$, *then* $\mathrm{NTIME}(f(n)) \subsetneq \mathrm{NTIME}(g(n))$.

Space and time

Theorem

If $S : \mathbb{N} \to \mathbb{N}$ is a space-constructible function, then:

$$\mathsf{DTIME}(S(n)) \subseteq \mathsf{SPACE}(S(n)) \subseteq \mathsf{NSPACE}(S(n)) \subseteq \mathsf{DTIME}(2^{O(S(n))}).$$

Theorem

If $f, g : \mathbb{N} \to \mathbb{N}$ are space-constructible functions such that $f(n)$ is $o(g(n))$, then:

$$\mathsf{SPACE}(f(n)) \subsetneq \mathsf{SPACE}(g(n)) \quad \textit{and} \quad \mathsf{NSPACE}(f(n)) \subsetneq \mathsf{NSPACE}(g(n)).$$

Oracles and relativizing proofs

Theorem (Baker, Gill, Solovay 1975)

There exist $A, B \subseteq \{0,1\}^*$ *such that* $\mathsf{P}^A = \mathsf{NP}^A$ *and* $\mathsf{P}^B \neq \mathsf{NP}^B$.

Quantified Boolean formulas (QBFs)

Theorem

TQBF *is* PSPACE*-complete.*

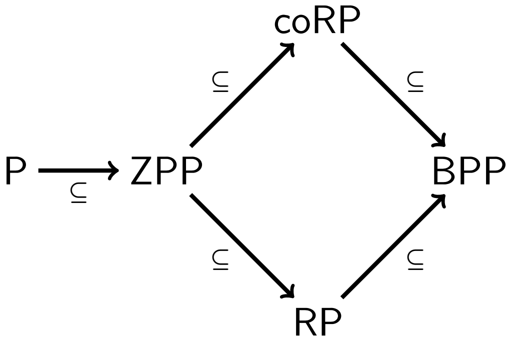
Theorem

Let $i \geq 1$*. Then* $\Sigma_i\mathsf{SAT}$ *is* Σ_i^p*-complete and* $\Pi_i\mathsf{SAT}$ *is* Π_i^p*-complete.*

Theorem (Karp, Lipton 1980)

If $\mathsf{NP} \subseteq \mathsf{P}/\mathrm{poly}$, *then* $\Sigma_2^p = \Pi_2^p$.

Probabilistic computation



PCP and approximation

Theorem (PCP)

$\mathsf{NP} = \mathsf{PCP}(\log n, 1)$.

There exists some $\rho < 1$ such that for all $L \in \mathsf{NP}$ there is a polynomial-time reduction R from L to 3SAT where for all $x \in \{0,1\}^*$:

- *if $x \in L$ then $val(R(x)) = 1$;*
- *if $x \notin L$ then $val(R(x)) < \rho$.*

ETH

Definition

Let δ_3 be the infimum of the set of constants c for which there exists an algorithm solving 3SAT in time $O(2^{cn}) \cdot m^{O(1)}$, where n is the number of variables in the q-SAT input and m the number of clauses.

The *Exponential-Time Hypothesis (ETH)* states that $\delta_3 > 0$.

Theorem

The ETH implies that there is no $2^{o(n)}$-time algorithm for 3SAT *and that there is no $2^{o(n+m)}$-time algorithm for* 3SAT*.*

Definition (distP)

$\langle L, \mathcal{D} \rangle$ is in the class distP (also called: avgP) if there exists a deterministic TM $\mathbb{M}$ that decides L and a constant $\epsilon > 0$ such that for all $n \in \mathbb{N}$:

$$\mathop{\mathbb{E}}_{x \in_{\mathsf{R}} \mathcal{D}_n} [\ \mathsf{time}_{\mathbb{M}}(x)^{\epsilon}\ ] \text{ is } O(n).$$

Parameterized complexity: with VC as example

- VC: given a graph G and $u \in \mathbb{N}$, does G have a vertex cover of size u?
- This problem is NP-complete, and the best algorithms that we have take exponential time in the worst case.
- This worst-case analysis takes into account **every** possible input.
- Can we take into account additional knowledge about the input that we might have to get more positive worst-case guarantees?

Parameterized complexity: with VC as example (ct'd)

- Suppose that we are dealing with an application where the value of u is always much smaller than the size of the graph G.
- Can we restrict the exponential factor in the running time to just u?
- **Answer:** yes!

Fixed-parameter tractability

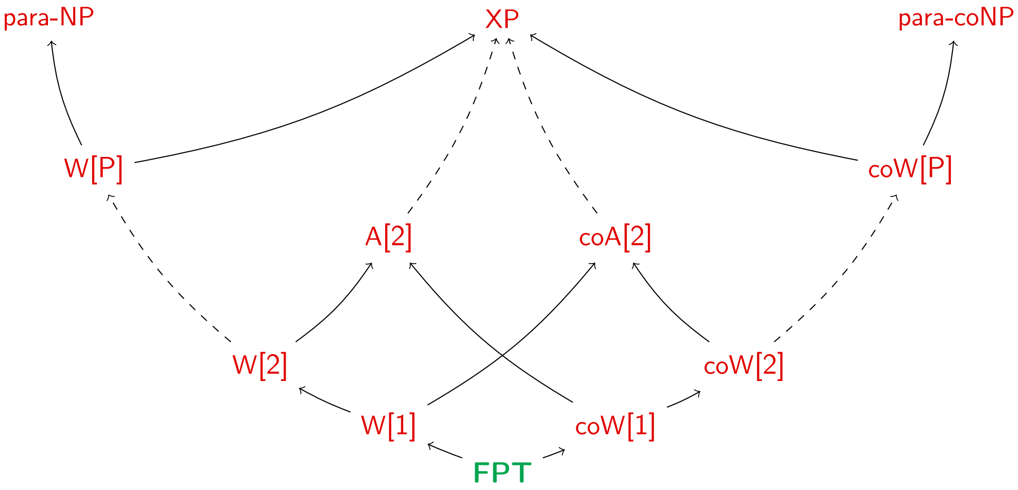
Definition

A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$ of pairs (x, k), where x is called the *main input* and k is called the *parameter*.

Definition (FPT)

A parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is *fixed-parameter tractable* if there exist a polynomial p, a computable function f, and a deterministic TM $\mathbb{M}$ that, when given input (x, k), decides if $(x, k) \in L$ and runs in time $f(k) \cdot p(|x|)$.

Parameterized complexity landscape



Parameterized complexity: ‘dialogues’ with your problems

- VC: NP-complete, and no $2^{o(v)}$-time algorithm (assuming ETH)

- With u as parameter? Fixed-parameter tractable
- With $v - u$ as parameter? W[1]-complete
- With the degree d of the graph as parameter? para-NP-complete
- With the treewidth t of the graph as parameter? Fixed-parameter tractable
- Etc.