

Computational Complexity

Lecture 7: the Polynomial Hierarchy

Ronald de Haan

me@ronalddehaan.eu

University of Amsterdam

April 22, 2025

Recap

What we saw last time..

- Space-bounded computation
- Limits on memory space
- L, NL, PSPACE
- Logspace reductions
- NL-completeness

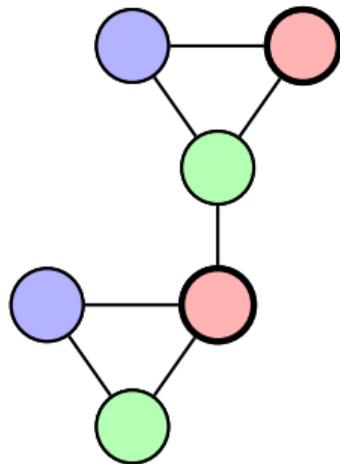
What will we do today?

- The Polynomial Hierarchy
- Bounded quantifier alternation
- Alternating Turing machines

- We saw that 3COL is NP-complete, but how about the following problem?

3COL-Extension = $\{ (G, V_0) \mid G = (V, E) \text{ is an undirected graph, } V_0 \subseteq V, \text{ and each 3-coloring of the vertices in } V_0 \text{ can be extended to a proper 3-coloring of the entire graph } G \}$

- There seems to be no single (polynomial-size) certificate for yes-inputs
- It is a “ $\forall\exists$ -type” question
- We need a different complexity class to capture the complexity of 3COL-Extension



Definition (NP)

A language $L \subseteq \{0, 1\}^*$ is in the class NP if there is a polynomial $q : \mathbb{N} \rightarrow \mathbb{N}$ and a polynomial-time Turing machine \mathbb{M} such that for every $x \in \{0, 1\}^*$:

$x \in L$ if and only if **there exists** some $u \in \{0, 1\}^{q(|x|)}$ such that $\mathbb{M}(x, u) = 1$.

Definition (coNP)

A language $L \subseteq \{0, 1\}^*$ is in the class coNP if there is a polynomial $q : \mathbb{N} \rightarrow \mathbb{N}$ and a polynomial-time Turing machine \mathbb{M} such that for every $x \in \{0, 1\}^*$:

$x \in L$ if and only if **for all** $u \in \{0, 1\}^{q(|x|)}$ it holds that $\mathbb{M}(x, u) = 1$.

Definition (Σ_2^P)

A language $L \subseteq \{0, 1\}^*$ is in the class Σ_2^P if there is a polynomial $q : \mathbb{N} \rightarrow \mathbb{N}$ and a polynomial-time Turing machine \mathbb{M} such that for every $x \in \{0, 1\}^*$:

$x \in L$ if and only if there exists $u_1 \in \{0, 1\}^{q(|x|)}$ such that
for all $u_2 \in \{0, 1\}^{q(|x|)}$ it holds that $\mathbb{M}(x, u_1, u_2) = 1$.

Definition (Π_2^P)

A language $L \subseteq \{0, 1\}^*$ is in the class Π_2^P if there is a polynomial $q : \mathbb{N} \rightarrow \mathbb{N}$ and a polynomial-time Turing machine \mathbb{M} such that for every $x \in \{0, 1\}^*$:

$x \in L$ if and only if for all $u_1 \in \{0, 1\}^{q(|x|)}$
there exists $u_2 \in \{0, 1\}^{q(|x|)}$ such that $\mathbb{M}(x, u_1, u_2) = 1$.

- It turns out that 3COL-Extension is Π_2^P -complete.

Definition (Σ_i^P)

Let $i \geq 1$. A language $L \subseteq \{0, 1\}^*$ is in the class Σ_i^P if there is a polynomial $q : \mathbb{N} \rightarrow \mathbb{N}$ and a polynomial-time Turing machine \mathbb{M} such that for every $x \in \{0, 1\}^*$:

$x \in L$ if and only if there exists $u_1 \in \{0, 1\}^{q(|x|)}$ such that
 for all $u_2 \in \{0, 1\}^{q(|x|)}$
 \vdots
 for all $u_i \in \{0, 1\}^{q(|x|)}$
 it holds that $\mathbb{M}(x, u_1, \dots, u_i) = 1$. if i is even,

\vdots
 there exists $u_i \in \{0, 1\}^{q(|x|)}$
 such that $\mathbb{M}(x, u_1, \dots, u_i) = 1$. if i is odd.

Definition (Π_i^P)

Let $i \geq 1$. A language $L \subseteq \{0, 1\}^*$ is in the class Π_i^P if there is a polynomial $q : \mathbb{N} \rightarrow \mathbb{N}$ and a polynomial-time Turing machine \mathbb{M} such that for every $x \in \{0, 1\}^*$:

$x \in L$ if and only if

- for all $u_1 \in \{0, 1\}^{q(|x|)}$
- there exists $u_2 \in \{0, 1\}^{q(|x|)}$ such that
- \vdots
- for all $u_i \in \{0, 1\}^{q(|x|)}$
- it holds that $\mathbb{M}(x, u_1, \dots, u_i) = 1.$ if i is odd,

- \vdots
- there exists $u_i \in \{0, 1\}^{q(|x|)}$
- such that $\mathbb{M}(x, u_1, \dots, u_i) = 1.$ if i is even.

The Polynomial Hierarchy (PH)

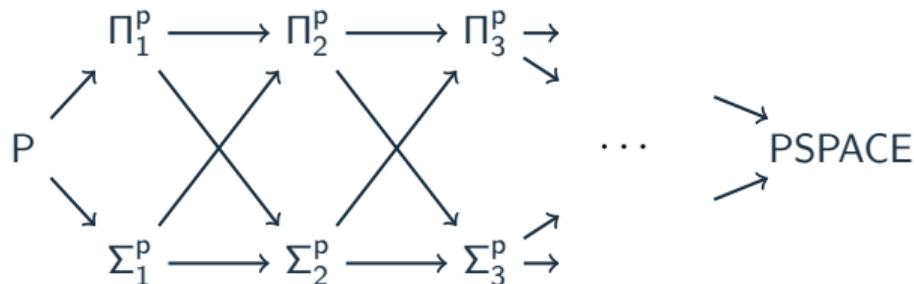
Definition (Σ_0^P , Π_0^P , PH)

$$\Sigma_0^P = \Pi_0^P = P$$

$$PH = \bigcup_{i \geq 0} \Sigma_i^P.$$

■ Some relations:

- $\Pi_i^P = \{ \bar{L} \mid L \in \Sigma_i^P \}$
- $\Sigma_1^P = NP$, $\Pi_1^P = coNP$
- $\Sigma_i^P \subseteq \Pi_{i+1}^P \subseteq \Sigma_{i+2}^P$,
 $\Pi_i^P \subseteq \Sigma_{i+1}^P \subseteq \Pi_{i+2}^P$
- $\Sigma_i^P \subseteq \Sigma_{i+1}^P$, $\Pi_i^P \subseteq \Pi_{i+1}^P$
- $\Sigma_i^P \cup \Pi_i^P \subseteq PSPACE$
- $PH \subseteq PSPACE$



“Collapse” of the hierarchy

- Statements like “ $P \neq NP$ ” and “ $NP \neq coNP$ ” are widely believed conjectures
- We can use these as assumptions to show some results
 - E.g., assuming that $P \neq NP$, NP-complete problems are not in P.
- For some results, stronger conjectures seem necessary
- Another conjecture: “the PH does not collapse”
 - “the PH collapses to P” $PH = P$
 - “the PH collapses to the i th level” $PH = \Sigma_i^P$

Theorem

Let $i \geq 1$. If $\Sigma_i^P = \Pi_i^P$, then $PH = \Sigma_i^P$.

If $P = NP$, then $PH = P$.

- $\Sigma_i\text{SAT} = \{ \varphi = \exists \bar{u}_1 \forall \bar{u}_2 \dots Q_i \bar{u}_i \psi(\bar{u}_1, \dots, \bar{u}_i) : \varphi \text{ is a true QBF} \}$,
 where each $\bar{u}_j = (u_{j,1}, \dots, u_{j,\ell})$ is a sequence of propositional variables,
 $\exists \bar{u}_j$ stands for $\exists u_{j,1} \exists u_{j,2} \dots \exists u_{j,\ell}$, and $\forall \bar{u}_j$ for $\forall u_{j,1} \forall u_{j,2} \dots \forall u_{j,\ell}$
- $\Pi_i\text{SAT} = \{ \varphi = \forall \bar{u}_1 \exists \bar{u}_2 \dots Q_i \bar{u}_i \psi(\bar{u}_1, \dots, \bar{u}_i) : \varphi \text{ is a true QBF} \}$,

Theorem

Let $i \geq 1$. Then $\Sigma_i\text{SAT}$ is Σ_i^P -complete and $\Pi_i\text{SAT}$ is Π_i^P -complete (both under polynomial-time reductions).

Theorem

Let $i \geq 2$. Then $\Sigma_i^P = \text{NP}^{\Sigma_{i-1}\text{SAT}}$ and $\Pi_i^P = \text{coNP}^{\Sigma_{i-1}\text{SAT}}$.

- (Or replace $\Sigma_{i-1}\text{SAT}$ by any Σ_{i-1}^P -complete or Π_{i-1}^P -complete problem.)
- This is often written as: $\Sigma_i^P = \text{NP}^{\Sigma_{i-1}^P}$ and $\Pi_i^P = \text{coNP}^{\Sigma_{i-1}^P}$

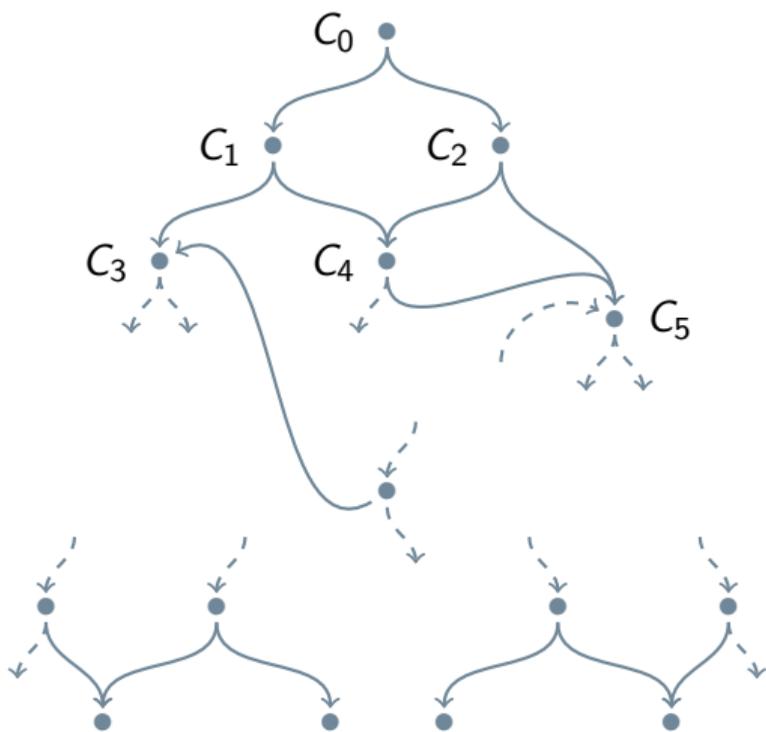
Configuration graphs

Configurations C consist of:

- (1) tape contents
- (2) tape head positions
- (3) state $q \in Q$

Configuration graph of a TM M
on some input x :

- Nodes are all the configurations that are reachable from the initial configuration C_0
- Edge from C to C' if applying one of the transition functions in C results in C'



Definition (Alternating Turing machines; ATMs)

- Instead of a single transition function δ , there are two transition functions δ_1, δ_2 .
- The set $Q \setminus \{q_{acc}, q_{rej}\}$ is partitioned into Q_{\exists} and Q_{\forall} .
- Executions of alternating TMs are defined using a labeling procedure on the *configuration graph*. Repeatedly apply, until a fixpoint is reached:
 - Label each configuration with q_{acc} with “accept.”
 - If a configuration c with $q \in Q_{\exists}$ has an edge to a configuration c' that is labeled with “accept,” then label c with “accept.”
 - If a configuration c has a state $q \in Q_{\forall}$ and both configurations c', c'' that are reachable from it in the graph are labeled with “accept,” then label c with “accept.”

The TM accepts the input if the starting configuration is labeled with “accept.”

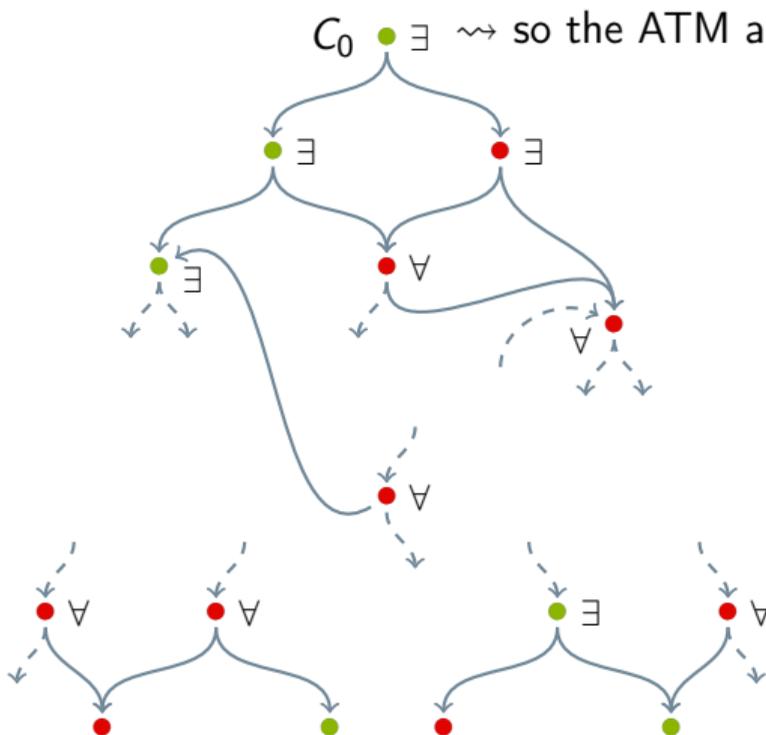
- The TM runs in time $T(n)$ if for every input x and for every possible sequence of transition function choices, the machine halts after at most $T(|x|)$ steps.

Alternating Turing machines (ct'd)

$C_0 \text{ } \bullet \exists \rightsquigarrow$ so the ATM accepts the input

\bullet = reject

\bullet = accept



Definition (ATIME)

Let $T : \mathbb{N} \rightarrow \mathbb{N}$ be a function. A decision problem $L \subseteq \{0, 1\}^*$ is in $\text{ATIME}(T(n))$ if there exists an ATM that decides L and that runs in time $O(T(n))$.

Definition (Σ_i TIME)

Let $T : \mathbb{N} \rightarrow \mathbb{N}$ be a function. A decision problem $L \subseteq \{0, 1\}^*$ is in $\Sigma_i\text{TIME}(T(n))$ if there exists an ATM that decides L , that runs in time $O(T(n))$, whose initial state is in Q_{\exists} , and that on every input and on every path in the configuration graph alternates at most $i - 1$ times between Q_{\exists} and Q_{\forall} .

- $\Pi_i\text{TIME}$ is defined similarly to $\Sigma_i\text{TIME}$, with the difference that the initial state of the ATM is in Q_{\forall}

Theorem

$$\text{PSPACE} = \bigcup_{c \geq 0} \text{ATIME}(n^c).$$

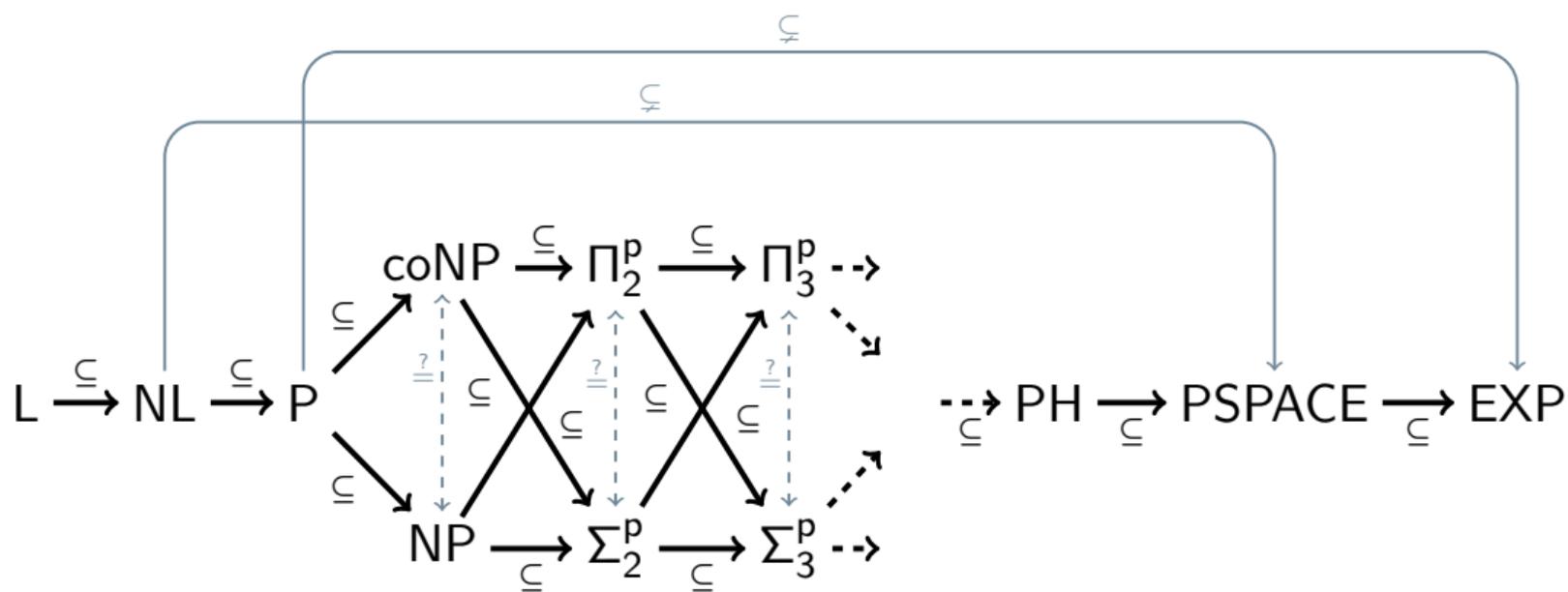
Theorem

Let $i \geq 1$. Then:

$$\Sigma_i^p = \bigcup_{c \geq 0} \Sigma_i \text{TIME}(n^c)$$

$$\Pi_i^p = \bigcup_{c \geq 0} \Pi_i \text{TIME}(n^c).$$

An overview of complexity classes



- The classes Σ_i^P and Π_i^P
- The Polynomial Hierarchy
- Σ_i^P -complete and Π_i^P -complete QBF problems
- Characterizations using oracles and ATMs

- Non-uniform complexity
- Circuit complexity
- TMs that take advice
- The Karp-Lipton Theorem