Aalto University
School of Science

CS-E4530 Computational Complexity Theory

Lecture 10: The Polynomial Time Hierarchy and Alternating Turing Machines

Aalto University
School of Science
Department of Computer Science

Spring 2017

Agenda

- The polynomial time hierarchy
- Alternating Turing machines

(C. Papadimitriou: *Computational Complexity*, Chapters 16 & 17)

1. The Polynomial Time Hierarchy

Definition

The *polynomial time hierarchy* comprises the sequence of classes:

- $\Delta_0^p = \Sigma_0^p = \Pi_0^p = \mathbf{P}$
- $i \geq 0$: $\Delta_{i+1}^p = \mathbf{P}^{\Sigma_i^p}$
  $\Sigma_{i+1}^p = \mathbf{NP}^{\Sigma_i^p}$
  $\Pi_{i+1}^p = \mathbf{coNP}^{\Sigma_i^p}$
- *Cumulative polynomial time hierarchy*: $\mathbf{PH} = \bigcup_{i \geq 0} \Sigma_i^p$

The polynomial time hierarchy—cont'd

Properties:

- $\Delta_1^p = \mathbf{P}^{\Sigma_0^p} = \mathbf{P}^{\mathbf{P}} = \mathbf{P}$
  $\Sigma_1^p = \mathbf{NP}^{\Sigma_0^p} = \mathbf{NP}^{\mathbf{P}} = \mathbf{NP}$
  $\Pi_1^p = \mathbf{coNP}^{\Sigma_0^p} = \mathbf{coNP}$
- $\Delta_2^p = \mathbf{P}^{\Sigma_1^p} = \mathbf{P}^{\mathbf{NP}}$
  $\Sigma_2^p = \mathbf{NP}^{\Sigma_1^p} = \mathbf{NP}^{\mathbf{NP}}$
  $\Pi_2^p = \mathbf{coNP}^{\Sigma_1^p} = \mathbf{coNP}^{\mathbf{NP}}$
- $\Delta_i^p \subseteq \begin{matrix}\Sigma_i^p \\ \Pi_i^p\end{matrix} \subseteq \Delta_{i+1}^p \subseteq \begin{matrix}\Sigma_{i+1}^p \\ \Pi_{i+1}^p\end{matrix} \subseteq \Delta_{i+2}^p$

Certificates

Theorem

Let L be a language and $i \geq 1$. Then $L \in \Sigma_i^p$ iff there is a polynomially balanced relation R such that the language $\{x;y \mid (x,y) \in R\}$ is in Π_{i-1}^p and

$$L = \{x \mid \textit{there is a } y \textit{ such that } (x,y) \in R\}.$$

Corollary

Let L be a language and $i \geq 1$. Then $L \in \Pi_i^p$ iff there is a polynomially balanced relation R such that the language $\{x;y \mid (x,y) \in R\}$ is in Σ_{i-1}^p and

$$L = \{x \mid \textit{for all } y \textit{ with } |y| \leq |x|^k, (x,y) \in R\}.$$

Certificates—cont'd

A relation $R \subseteq (\Sigma^*)^{i+1}$ is said to be *polynomially balanced* if whenever $(x, y_1, \ldots, y_i) \in R$, it holds that $|y_1|, \ldots, |y_i| \leq |x|^k$ for some k.

Corollary

Let L be a language and $i \geq 1$. Then $L \in \Sigma_i^p$ iff there is a polynomially balanced, polynomial-time decidable $(i+1)$-ary relation R such that

$$L = \{x \mid \exists y_1 \forall y_2 \exists y_3 \cdots Q y_i \text{ such that } (x, y_1, \ldots, y_i) \in R\}$$

where Q is $\forall$ if i is even and $\exists$ if i is odd.

Is PH a proper hierarchy?

Proposition

If $\Sigma_i^p = \Pi_i^p$ *for some* $i \geq 1$, *then* $\Delta_j^p = \Sigma_j^p = \Pi_j^p = \Sigma_i^p$ *for all* $j > i$.

In this case the polynomial time hierarchy is said to *collapse* to the ith level.

- If $\mathbf{P} = \mathbf{NP}$, or if $\mathbf{NP} = \mathbf{coNP}$, then the polynomial time hierarchy collapses to the first level. (If $\mathbf{P} = \mathbf{NP}$, then actually already to the zeroth level.)
- $\mathbf{P} = \mathbf{NP}$ iff $\mathbf{P} = \mathbf{PH}$.
- Notice that it can be the case that $\mathbf{P} \neq \mathbf{NP}$ and $\mathbf{NP} \neq \mathbf{coNP}$ but the polynomial time hierarchy collapses to the second level. (This is not expected to happen, though.)

Complete problems

- QSAT_i (quantified satisfiability with i alternations of quantifiers):
  Given a Boolean expression ϕ with the Boolean variables partitioned into i sets $X_1, \ldots, X_i$, is it true that
  there is a partial truth assignment for the variables X_1 such that
  for all partial truth assignments for X_2
  there is a partial truth assignment for X_3
  $\ldots$ ϕ is satisfied by the overall truth assignment?
- QSAT_i: Is the following *quantified Boolean expression* true

$$\exists X_1 \forall X_2 \exists X_3 \cdots Q X_i \ \ \phi$$

  where Q is $\forall$ if i is even and $\exists$ if i is odd?

Example

$\exists\{p\}\forall\{q\}(p \vee q) \in \mathrm{QSAT}_2$ but $\exists\{p\}\forall\{q\}(p \wedge q) \notin \mathrm{QSAT}_2$

Complete problems

Theorem

For all $i \geq 1$, QSAT_i is Σ_i^p-complete.

Example

CIRCUIT MINIMISATION: Given a Boolean circuit C and an integer K, is there a circuit C' with at most K gates that computes the same function as C?

CIRCUIT MINIMISATION is Σ_2^p-complete (Buchfuhrer & Umans 2008).

Example

INTEGER EXPRESSION EQUIVALENCE: Do two *integer expressions* E_1 and E_2, built out of binary numbers by $+$ and $\cup$, describe the same set? (E.g. $(001 \cup 011) + (010) = (011 \cup 101)$.)

INTEGER EXPRESSION EQUIVALENCE is Π_2^p-complete (Stockmeyer 1973).

Complete problems—cont'd

Theorem

If there is a $\mathbf{PH}$*-complete problem, then the polynomial time hierarchy collapses to some finite level.*

Proof. Assume L is $\mathbf{PH}$-complete. Then $L \in \Sigma_i^p$ for some i. But then any $L' \in \Sigma_{i+1}^p$ reduces to L. This means that $\Sigma_i^p = \Sigma_{i+1}^p$ (all levels are closed under reductions). $\square$

Complete problems—cont'd

- $\mathbf{PH} \subseteq \mathbf{PSPACE}$
  $L \in \mathbf{PH}$ iff $L \in \Sigma_i^p$ for some i iff
  $L = \{x \mid \exists y_1 \forall y_2 \cdots Q y_i \text{ s.t. } (x, y_1, \ldots, y_i) \in R\}$ for some i.
- It is open whether $\mathbf{PH} = \mathbf{PSPACE}$.
  If $\mathbf{PH} = \mathbf{PSPACE}$, then the polynomial time hierarchy collapses to some finite level. (There are $\mathbf{PSPACE}$-complete problems.)
- If $\mathbf{PH}$ does not collapse, *problems are strictly harder in an upper level* when compared to the lower level: if L is a Σ_{i+1}^p-complete language and $L \in \Sigma_i^p$, then $\mathbf{PH}$ collapses to level i.

Example

Consider a Σ_2^p-complete problem. It cannot be solved with a polynomial overhead on top of a procedure for a problem in $\mathbf{NP}$ (unless $\mathbf{PH}$ collapses to level 1).

5. Alternating Turing Machines

- Alternation is an important generalisation of nondeterminism.
- In a nondeterministic computation each configuration is an implicit *OR* of its successor configurations: i.e.
  a configuration "leads to acceptance" iff at least one of its successors does.
- The idea is to allow both *OR* and *AND* configurations in a tree of configurations generated by a NTM N computing on input x.

Definition

An *alternating* Turing machine N is a nondeterministic Turing machine where the set of states K is partitioned into two sets $K = K_{\text{AND}} \cup K_{\text{OR}}$.

Given the tree of configurations of N on input x, the *eventually accepting configurations* of N are defined recursively:

1. Any leaf configuration with state "yes" is eventually accepting.
2. A configuration with state in K_{AND} is eventually accepting iff all its successors are.
3. A configuration with state in K_{OR} is eventually accepting iff at least one of its successors is.

☞ N *accepts* x iff its initial configuration is eventually accepting.

Alternation-based complexity classes

Definition

An alternating Turing machine N *decides* a language L iff N accepts all strings $x \in L$ and rejects all strings $x \notin L$.

- It is straightforward to define $\mathbf{ATIME}(f(n))$ and $\mathbf{ASPACE}(f(n))$; and using them, e.g. $\mathbf{AP} = \mathbf{ATIME}(n^k)$, $\mathbf{AL} = \mathbf{ASPACE}(\log n)$ etc.
- Roughly speaking, alternating time classes correspond to deterministic space and alternating space classes correspond to deterministic time but one exponential higher.

Theorem

$\mathbf{AL} = \mathbf{P}$, $\mathbf{AP} = \mathbf{PSPACE}$, $\mathbf{APSPACE} = \mathbf{EXP}$, ...

Alternation and the polynomial time hierarchy

Denote by $\Sigma_i\mathbf{P}$ (resp. $\Pi_i\mathbf{P}$), $i \geq 1$, the family of languages decided by polynomially time-bounded alternating Turing machines whose every computation satisfies the following conditions:

- The initial state belongs to K_{OR} (resp. K_{AND}).
- The computation *alternates* from a state in K_{OR} to a state in K_{AND} or vice versa at most $i-1$ times.

By definition, set also $\Sigma_0\mathbf{P} = \Pi_0\mathbf{P} = \mathbf{P}$.

Theorem

For every $i \geq 0$, $\Sigma_i\mathbf{P} = \Sigma_i^p$ and $\Pi_i\mathbf{P} = \Pi_i^p$.

Learning Objectives

- The concept of the polynomial time hierarchy
- Alternating Turing machines and alternation-based complexity classes