Complexity Theory

Georg Moser

Institute of Computer Science @ UIBK

Summer 2008

Outline

- Summary of Last Lecture: The Polynomial-Time Hierarchy
- Exercises
- More on the Polynomial-Time Hierarchy
- The Arithmetical Hierarchy

Definition of PH via ATMs

Definition (Σ_k-machine)

a Σ_k-machine is an ATM for which the computation path is dividable in separate sections on any input and

1. any section consists only of $\wedge$- or $\vee$-configurations
2. at most k sections
3. the first consist of $\vee$-configurations

a Π_k-machine is defined by swapping $\vee$ and $\wedge$

Σ_0, Π_0 are defined to be deterministic TMs

Definition (Σ_k^{p}, Π_k^{p})

$$\Sigma_k^{\mathrm{p}} := \{\mathrm{L}(M) \mid M \text{ is polytime bounded } \Sigma_k\text{-machines}\}$$
$$\Pi_k^{\mathrm{p}} := \{\mathrm{L}(M) \mid M \text{ is polytime bounded } \Pi_k\text{-machines}\}$$

Definition

- an oracle machine is a TM M^B with an extra write-only tape, the oracle tape
- M^B additionally has oracle query state and specific oracle answer states "yes" and "no"
- M^B writes y on oracle tape, oracle answers "yes" if $y \in B$ and "no" otherwise

Definition

let B be a language and $\mathcal{C}$ a complexity class

$$\mathrm{P}^B := \{\mathrm{L}(M) \mid M \text{ is a deterministic, polytime bounded oracle machine with oracle } B\}$$
$$\mathrm{NP}^B := \{\mathrm{L}(M) \mid M \text{ is a nondeterministic, polytime bounded oracle machine with oracle } B\}$$
$$\mathrm{P}^{\mathcal{C}} := \bigcup_{B \in \mathcal{C}} \mathrm{P}^B \qquad \mathrm{NP}^{\mathcal{C}} := \bigcup_{B \in \mathcal{C}} \mathrm{NP}^B$$

Theorem

consider

$$\mathrm{NP} \subseteq \mathrm{NP}^{\mathrm{NP}} \subseteq \mathrm{NP}^{\mathrm{NP}^{\mathrm{NP}}} \ldots$$

i.e., $\mathrm{NP}_1 := \mathrm{NP}$ and $\mathrm{NP}_{k+1} := \mathrm{NP}^{\mathrm{NP}_k}$, then $\forall k \geqslant 1$: $\mathrm{NP}_k = \Sigma_k^{\mathrm{p}}$

define $\exists^t x\ \varphi(x) :\Leftrightarrow \exists x |y| \leqslant t \wedge \varphi(x)$ and $\forall^t x\ \varphi(x) :\Leftrightarrow \forall x |y| \leqslant t \to \varphi(x)$

Theorem

a language L is in Σ_k^{p} iff there is a deterministic polytime computable $(k+1)$-ary predicate R and a constant c such that

$$A = \{x \mid \exists^{|x|^c} y_1 \forall^{|x|^c} y_2 \exists^{|x|^c} y_3 \ldots \mathrm{Q}^{|x|^c} y_k R(x, y_1, \ldots, y_k)\}$$

$(\mathrm{Q} \in \{\exists, \forall\})$

Homework

1. Miscellaneous Exercises 4
2. Miscellaneous Exercises 13
3. Miscellaneous Exercises 18
4. Homework 3.2
5. Homework 5.1

Theorem

$\forall k \geqslant 1$: $\mathrm{NP}_k = \Sigma_k^{\mathrm{p}}$

Proof

the proof proceeds by induction on k; the base case is easy:

$$\mathrm{NP}_1 = \mathrm{NP} = \Sigma_1^{\mathrm{p}}$$

employing the induction hypothesis, it remains to show $\mathrm{NP}^{\Sigma_k^{\mathrm{p}}} = \Sigma_{k+1}^{\mathrm{p}}$

$\mathrm{NP}^{\Sigma_k^{\mathrm{p}}} \supseteq \Sigma_{k+1}^{\mathrm{p}}$

- $\exists\ \Sigma_{k+1}$-machine M running in time n^c, $A \in \mathrm{L}(M)$
- we need to show $A \in \mathrm{NP}^{\Sigma_k^{\mathrm{p}}}$
- wlog assume all configurations of M are representable as string in Δ^{n^c}
- $$D := \{\alpha \mid \alpha \text{ is an } \wedge\text{-configuration of } M, |\alpha| = n^c, \text{ and } \alpha \text{ leads to acceptance via a } \Pi_k \text{ computation in time at most } n^c\}$$
- M accepts x iff $\exists$ computation leading via $\vee$-states into some $\alpha \in D$
- A is accepted by an NTM with oracle $\sim D \in \Sigma_k^{\mathrm{p}}$

$\mathrm{NP}^{\Sigma_k^{\mathrm{p}}} \subseteq \Sigma_{k+1}^{\mathrm{p}}$

- $\exists$ NTM n^c-time bounded with oracle $B \in \Sigma_k^{\mathrm{p}}$, $A = \mathrm{L}(M)$
- construct Σ_{k+1}-machine N:
  1. on input x, N simulates M
  2. every time M wants to ask oracle on y, N remembers y and spawns processes to guess answer
  3. if M rejects, N rejects
  4. if M accepts, correctness of guesses need to be verified
- this part of N is a Σ_1-machine
- each leaf of N's computation tree collects
  positive guesses $y_1, \ldots, y_m$ $\in B$?
  negative guesses $z_1, \ldots, z_\ell$ $\notin B$?
- we extend N by guessing strings $w_1, \ldots, w_m$ used in the first section of the Σ_k-TM deciding $y_i \in B$
- the subsequent $\wedge$-state forks $m + \ell$ processes each process either checking $y_i \in B$ or $z_j \notin B$
- these processes are Π_{k-1} and Π_k respectively ■

Definition

- a set A is recursive enumerable in B if $A = \mathrm{L}(M^B)$ for some oracle TM M^B
- A is recursive in B if $A = \mathrm{L}(M^B)$ and M^B is a total oracle TM
- $A \leqslant_T B$, if A recursive in B — Turing reducibility

Definition — Arithmetical Hierarchy

we fix a binary alphabet $\Sigma = \{0, 1\}$

$\Sigma_1^0 := \{\text{r.e. sets}\}$ $\qquad$ $\Sigma_{n+1}^0 := \{\mathrm{L}(M^B) \mid B \in \Sigma_n^0\}$

$\Delta_1^0 := \{\text{recursive sets}\}$ $\qquad$ $\Delta_{n+1}^0 := \{\mathrm{L}(M^B) \mid B \in \Sigma_n^0, M^B \text{ total}\}$

$\Pi_n^0 := \{\sim L \mid L \in \Sigma_n^0\}$

Example

$$\mathrm{HP} = \{M\#x \mid \exists t \ M \text{ halts on } x \text{ in } t \text{ steps}\} \in \Sigma_1^0$$

$$\mathrm{MP} = \{M\#x \mid \exists t \ M \text{ accepts } x \text{ in } t \text{ steps}\} \in \Sigma_1^0$$

Theorem

- a set A is in Σ_n^0 iff $\exists$ a decidable $(n+1)$-ary predicate R such that
$$A = \{x \mid \exists y_1 \forall y_2 \dots \mathsf{Q} y_n R(x, y_1, \dots, y_n)\}$$
$(\mathsf{Q} \in \{\exists, \forall\})$
- a set A is in Π_n^0 iff $\exists$ a decidable $(n+1)$-ary predicate R such that
$$A = \{x \mid \forall y_1 \exists y_2 \dots \mathsf{Q} y_n R(x, y_1, \dots, y_n)\}$$
$(\mathsf{Q} \in \{\exists, \forall\})$

Definition

- let $A \subseteq \Sigma^*$, $B \subseteq \Gamma^*$, define $A \leqslant_m B$ if $\exists$ total recursive function $\sigma \colon \Sigma^* \to \Gamma^*$ such that $\forall\, x \in \Sigma^*$
$$x \in A \Leftrightarrow \sigma(x) \in B$$
- a set A is r.e.-hard if every r.e. set $\leqslant_m$-reduces to A
- if A is r.e. and r.e.-hard, then A is r.e.-complete
- let $\mathcal{C}$ be a class of sets, we say A is $\leqslant_m$-complete for $\mathcal{C}$ if $A \in \mathcal{C}$ and A is $\leqslant_m$-hard

Example

- HP is $\leqslant_m$-complete for Σ_1^0
- MP is $\leqslant_m$-complete for Σ_1^0
- $\mathrm{FIN} = \{M \mid \mathrm{L}(M) \text{ is finite}\}$ is $\leqslant_m$-complete for Σ_2^0

Lemma

FIN is $\leqslant_m$-complete for Σ_2^0

Proof

on blackboard