

# Complexity Theory

## WS 2009/10

Prof. Dr. Erich Grädel

Mathematische Grundlagen der Informatik  
RWTH Aachen

## Contents

1	Deterministic Turing Machines and Complexity Classes	1
1.1	Turing machines	1
1.2	Time and space complexity classes	4
1.3	Speed-up and space compression	7
1.4	The Gap Theorem	9
1.5	The Hierarchy Theorems	11
2	Nondeterministic complexity classes	17
2.1	Nondeterministic Turing machines	17
2.2	Elementary properties of nondeterministic classes	19
2.3	The Theorem of Immerman and Szelepcsényi	21
3	Completeness	27
3.1	Reductions	27
3.2	NP-complete problems: SAT and variants	28
3.3	P-complete problems	34
3.4	NLOGSPACE-complete problems	38
3.5	A PSPACE-complete problem	42
4	Oracles and the polynomial hierarchy	47
4.1	Oracle Turing machines	47
4.2	The polynomial hierarchy	49
4.3	Relativisations	52
5	Alternating Complexity Classes	55
5.1	Complexity Classes	56
5.2	Alternating Versus Deterministic Complexity	57
5.3	Alternating Logarithmic Time	61



This work is licensed under:

<http://creativecommons.org/licenses/by-nc-nd/3.0/de/>

Dieses Werk ist lizenziert unter:

<http://creativecommons.org/licenses/by-nc-nd/3.0/de/>

© 2009 Mathematische Grundlagen der Informatik, RWTH Aachen.

<http://www.logic.rwth-aachen.de>

6	Complexity Theory for Probabilistic Algorithms	63
6.1	Examples of probabilistic algorithms . . . . .	63
6.2	Probabilistic complexity classes and Turing machines . . . . .	72
6.3	Probabilistic proof systems and Arthur-Merlin games . . . . .	81

## 4 Oracles and the polynomial hierarchy

### 4.1 Oracle Turing machines

**Definition 4.1.** A deterministic (respectively nondeterministic) *oracle Turing machine* is a Turing machine with a designated oracle tape and three special states ? (query), Y (yes) and N (no).

A configuration  $C$  of an Oracle Turing machine with  $k$  working tapes and a distinguished oracle tape is a tuple

$$C = (q, p_0, \dots, p_k, w_0, \dots, w_k),$$

where

- $q$  is the state of the Turing machine,
- $p_0, \dots, p_k$  are the head positions ( $p_0$  is the head position of the oracle tape), and
- $w_0, \dots, w_k$  are the head inscriptions ( $w_0$  is the inscription of the oracle tape).

The computation (respectively the computation tree) of an oracle Turing machine depends on a previously defined oracle set  $A \subseteq \Sigma^*$  (where  $\Sigma$  is the alphabet of  $M$ ). The successor configurations of a configuration  $C$  are defined as usual for  $q \neq ?$  while the successor configuration  $C'$  for  $q = ?$  is defined as :

$$C' = \begin{cases} (Y, 0, p_1, \dots, p_k, \varepsilon, w_1, \dots, w_k) & \text{if } w_0 \in A \\ (N, 0, p_1, \dots, p_k, \varepsilon, w_1, \dots, w_k) & \text{if } w_0 \notin A \end{cases}$$

where  $\varepsilon$  is the empty word. The oracle therefore determines whether or not  $w_0$  (the inscription of the oracle tape) is in  $A$ . The machine consequently enters the corresponding state (Y or N) and the inscription of the oracle tape is erased.

**Definition 4.2.** Let  $M$  be an Oracle Turing machine and  $A \subseteq \Sigma^*$  be some oracle set. Then the accepted language is

$$L(M^A) := \{x : M \text{ accepts the input } x \text{ with oracle } A\}.$$

Based on the oracle set  $A$ , we define the following complexity classes:

- (i)  $P^A := \{L : \text{there is a deterministic Oracle TM } M \text{ that decides } L \text{ using oracle } A \text{ in polynomial time}\}.$
- (ii)  $NP^A := \{L : \text{there is a nondeterministic Oracle TM } M \text{ that decides } L \text{ using oracle } A \text{ in polynomial time}\}.$

Let  $\mathcal{C}$  be some class of languages, e.g., a complexity class. Then

$$P^{\mathcal{C}} = \bigcup_{A \in \mathcal{C}} P^A \quad \text{and} \quad NP^{\mathcal{C}} = \bigcup_{A \in \mathcal{C}} NP^A.$$

*Example 4.3.*

- (a) Let  $B \in NP$ . Then  $B \in P^{\text{SAT}}$ . Since SAT is NP-hard, there is a polynomially computable function  $f$  with  $x \in B \iff f(x) \in \text{SAT}$ . The following oracle algorithm then decides  $B$ :

---

**Input:**  $x$   
 Compute  $f(x)$   
 Query the oracle whether  $f(x) \in \text{SAT}$   
**if**  $Y$  **then** accept  
**if**  $N$  **then** reject

---

- (b) It is likely that  $NP \subsetneq P^{\text{SAT}}$  as every  $B \in \text{coNP}$  is in  $P^{\text{SAT}}$ . One can use the preceding algorithm and interchange the behaviour for the answers Yes and No.
- (c) Let  $B := \{(G, k) : G \text{ a graph, } \omega(G) = k\}$  where  $\omega(G)$  is the maximal number of nodes of cliques in  $G$ . Reminder: The problem  $\text{CLIQUE} := \{(G, k) : k \leq \omega(G)\}$  is NP-complete. It is straightforward to see that  $B \in P^{\text{CLIQUE}}$ :

---

**Input:**  $G, k$

Query the oracle whether  $(G, k) \in \text{CLIQUE}$

**if**  $N$  **then** reject **else**

Query the oracle whether  $(G, k+1) \in \text{CLIQUE}$

**if**  $N$  **then** accept

**if**  $Y$  **then** reject

**endif**

---

## 4.2 The polynomial hierarchy

**Definition 4.4.** We define the complexity classes  $\Sigma_k^p$ ,  $\Pi_k^p$ , and  $\Delta_k^p$  for all  $k \in \mathbb{N}$ :

- $\Sigma_0^p := \Pi_0^p := \Delta_0^p := P$
- $\Sigma_{k+1}^p := NP^{\Sigma_k^p}$
- $\Pi_k^p := \text{co}\Sigma_k^p = \{\bar{A} : A \in \Sigma_k^p\}$
- $\Delta_{k+1}^p := P^{\Sigma_k^p}$

**Theorem 4.5.** The classes  $\Sigma_k^p$ ,  $\Pi_k^p$ , and  $\Delta_k^p$  have the following elementary properties:

- (i)  $\Delta_1^p = P$ .
- (ii)  $\Sigma_1^p = NP$ ,  $\Pi_1^p = \text{coNP}$ .
- (iii)  $\Sigma_{k+1}^p = NP^{\Pi_k^p} = NP^{\Delta_{k+1}^p}$ .
- (iv)  $P^{\Delta_k^p} = \Delta_k^p$ .

*Proof.* (i)  $\Delta_1^p = P^P = P$ .

(ii)  $\Sigma_1^p = NP^P = NP$ ,  $\Pi_1^p = \text{co}\Sigma_1^p = \text{coNP}$ .

(iii) Let  $B \in \Sigma_{k+1}^p$ ,  $B = L(M^A)$  and  $A \in \Sigma_k^p$ . Further, let  $M'$  be the machine obtained from  $M$  by interchanging the states  $Y$  and  $N$ . Obviously,  $B = L(M'^{\bar{A}})$  and therefore  $B \in NP^{\Pi_k^p}$ . In addition,  $NP^{\Delta_{k+1}^p} = NP^{P^{\Sigma_k^p}} = NP^{\Sigma_k^p} = \Sigma_{k+1}^p$  holds.

(iv)  $k = 0$ :  $P^{\Delta_0^p} = P^P = P = \Delta_0^p$ .

$k > 0$ :  $P^{\Delta_k^p} = P^{P^{\Sigma_{k-1}^p}} = P^{\Sigma_{k-1}^p} = \Delta_k^p$ .

Q.E.D.

**Theorem 4.6.** For all  $k$ ,  $\Sigma_k^p \cup \Pi_k^p \subseteq \Delta_{k+1}^p \subseteq \Sigma_{k+1}^p \cap \Pi_{k+1}^p$ .

*Proof.* For  $k = 0$ , the theorem states  $P \subseteq P \subseteq NP \cap \text{coNP}$ . This is obviously true.

For  $k > 0$ :

- $\Sigma_k^p \subseteq P^{\Sigma_k^p}$  and therefore also  $\Pi_k^p \subseteq P^{\Sigma_k^p}$  because  $P^{\Sigma_k^p}$  is closed under complement. Hence,  $\Sigma_k^p \cup \Pi_k^p \subseteq \Delta_{k+1}^p$ .
- $\Delta_{k+1}^p = P^{\Sigma_k^p} = \text{co}P^{\Sigma_k^p} \subseteq \text{coNP}^{\Sigma_k^p} = \Pi_{k+1}^p$ .  
 $\Delta_{k+1}^p = P^{\Sigma_k^p} \subseteq NP^{\Sigma_k^p} = \Sigma_{k+1}^p$ . Therefore,  $\Delta_{k+1}^p \subseteq \Sigma_{k+1}^p \cap \Pi_{k+1}^p$ .  
 Q.E.D.

**Theorem 4.7.** If there is a  $k$  such that  $\Sigma_{k+1}^p = \Sigma_k^p$ , then  $\Sigma_{k+i}^p = \Pi_{k+i}^p = \Sigma_k^p$  for all  $i > 0$ .

*Proof.* For  $i = 1$ ,  $\Sigma_{k+1}^p = \Sigma_{k+1}^p = \Sigma_k^p$  by assumption. By induction hypothesis, assume  $\Sigma_{k+i}^p = \Sigma_k^p$ . Then,

$$\Sigma_{k+i+1}^p = NP^{\Sigma_{k+i}^p} = NP^{\Sigma_k^p} = \Sigma_{k+1}^p = \Sigma_k^p,$$

and therefore also

$$\Pi_{k+i}^p \subseteq \Sigma_{k+i+1}^p = \Sigma_k^p \quad \text{for all } i.$$

In particular,  $\Pi_k^p \subseteq \Sigma_k^p$  holds. It remains to show that  $\Sigma_k^p \subseteq \Pi_k^p$ . If  $B \in \Sigma_k^p$ , then  $\bar{B} \in \Pi_k^p \subseteq \Sigma_k^p$  and, hence,  $B \in \Pi_k^p$ .  
 Q.E.D.

**Corollary 4.8.** If there is a  $k > 0$  with  $\Sigma_k^p \neq P$ , then  $P \neq NP$ .

**Definition 4.9.**  $\text{PH} := \bigcup_{k \in \mathbb{N}} \Sigma_k^p$  is called the *polynomial hierarchy*.

In case  $\Sigma_{k+i}^p = \Sigma_k^p$ , we say that the polynomial hierarchy collapses at level  $k$ .

**Theorem 4.10.**  $\text{PH} \subseteq \text{PSPACE}$ .

*Proof.* By induction over  $k$  we show that  $\Sigma_k^p \subseteq \text{PSPACE}$  for all  $k \in \mathbb{N}$ :

$$\Sigma_0^p = P \subseteq \text{PSPACE}$$

$$\Sigma_{k+1}^p = NP^{\Sigma_k^p} \subseteq NP^{\text{PSPACE}} \subseteq \text{PSPACE}^{\text{PSPACE}} = \text{PSPACE}.$$

Here,  $\text{PSPACE}^{\text{PSPACE}}$  is the class of languages that can be decided by a deterministic polynomially-space bounded Oracle Turing machine with some oracle in  $\text{PSPACE}$ . When we speak about space complexity, we also count the space used on the oracle tape. Q.E.D.

If  $\text{PH} = \text{PSPACE}$ , the polynomial hierarchy collapses:

**Theorem 4.11.** If  $\text{PH} = \text{PSPACE}$ , there is some  $k$  with  $\text{PH} = \Sigma_k^p$ .

*Proof.* If  $\text{PH} = \text{PSPACE}$ , then  $\text{QBF} \in \text{PH}$  holds. Consequently, there is some  $k$  such that  $\text{QBF} \in \Sigma_k^p$ . For each  $A \in \text{PSPACE}$ , we have  $A \leq_m^p \text{QBF}$ , i.e.,  $A \in \Sigma_k^p$ . Thus, we obtain  $\text{PH} = \Sigma_k^p$ .  
 Q.E.D.

It is assumed that  $\Sigma_k^p \subsetneq \Sigma_{k+1}^p$  for all  $k$ , the polynomial hierarchy is strict and therefore,  $\text{PH} \subsetneq \text{PSPACE}$ .

#### 4.2.1 Additions

There are two natural complete problems for  $\Sigma_k^p$  and  $\Pi_k^p$ :

$$\Sigma_k\text{-QBF} = \{\psi = (\exists \bar{X}_1)(\forall \bar{X}_2) \dots (Q_k \bar{X}_k) \varphi : \varphi \text{ quantifier free, } \psi \text{ true}\}$$

Here,  $Q_k$  is the universal quantifier if  $k$  is even and the existential quantifier otherwise.  $\Pi_k\text{-QBF}$  is defined analogously but the formulae begin with universal quantifiers. The problem  $\Sigma_k\text{-QBF}$  is  $\Sigma_k^p$ -complete and, analogously,  $\Pi_k\text{-QBF}$  is  $\Pi_k^p$ -complete. This generalises the NP-completeness of  $\text{SAT}$ .

Recall the definition of NP. A problem  $A \in \text{NP}$  if, and only if, there is some  $B \in P$  and some polynomial  $p(n)$  such that  $A = \{x : \exists^p y(x \# y \in B)\}$  where  $\exists^p y$  is an abbreviation for  $\exists y : |y| \leq p(|x|)$ . We can generalise this definition to obtain a characterisation of  $\Sigma_k^p$  and  $\Pi_k^p$  as follows:

- $A \in \Sigma_k^p$  if, and only if, there is some  $B \in P$  and some polynomial  $p(n)$  such that

$$A = \{x : (\exists^p y_1)(\forall^p y_2)(\exists^p y_3) \dots (Q_k^p y_k) x \# y_1 \# y_2 \# \dots \# y_k \in B\}.$$

Here,  $Q_k$  is the existential quantifier if  $k$  is odd and the universal quantifier otherwise.

- $\Pi_k^p$  can be characterised analogously, using formulae that begin with universal quantifiers.

### 4.3 Relativisations

To approach the  $P = NP$  question, it is interesting to see whether there are oracles  $A, B$  such that

- $P^A = NP^A$
- $P^B \neq NP^B$ .

The first question is easy to answer since  $P^{QBF} = NP^{QBF} = PSPACE$ . The answer to the second question is not as straightforward.

**Theorem 4.12.** There is an oracle  $B$  such that  $P^B \neq NP^B$ .

*Proof.* Just as Turing machines, polynomially time-bounded oracle Turing machines can also be enumerated recursively (exercise). We choose one such recursive enumeration  $\{M_i : i \in \mathbb{N}\}$  of deterministic polynomial oracle Turing machines such that the following holds for all oracles  $A$ :

- (1)  $P^A = \{L(M_i^A) : i \in \mathbb{N}\}$ .
- (2) There is a sequence  $\{p_i(n) : i \in \mathbb{N}\}$  of polynomials with:
  - (i)  $M_i$  is  $p_i$ -time bounded,
  - (ii)  $p_i(n) \leq p_{i+1}(n)$  for all  $i, n$ .

Any given sequence  $\{q_i(n) : i \in \mathbb{N}\}$  of time bounds for  $\{M_i : i \in \mathbb{N}\}$  can be modified to such a sequence  $p_{i+1}(n)$  by setting:

- $p_0(n) := q_0(n)$ ,
- $p_{i+1}(n) := \max(p_i(n), q_{i+1}(n))$ .

For every  $C \subseteq \{0, 1\}^*$ , let  $S(C) = \{0^n : \text{there is an } x \in C \text{ with } |x| = n\}$ .

We obviously have:

**Lemma 4.13.**  $S(B) \in NP^B$  for all  $B$ .

The goal now is to find some  $B$  such that  $S(B) \notin P^B$ . This  $B$  is constructed as follows: At the beginning, initialise  $B_0 := \emptyset$  and  $k_0 := 0$ . For  $n > 0$ , construct  $B_n, k_n$  as follows:

- Set  $k_n$  so it is the smallest integer with  $2^{k_n} > p_n(k_n)$  and  $k_n > p_{n-1}(k_{n-1})$ .
- If  $0^{k_n} \in L(M_n^{B_{n-1}})$ , then set  $B_n := B_{n-1}$ . Otherwise, let  $w(n)$  be the lexicographically first word in  $\{0, 1\}^{k(n)}$  for which the oracle is not queried during the computation of  $M_n^{B_{n-1}}$  on input  $0^{k_n}$ . Such a word exists since  $p_n(k_n) < 2^{k_n}$ . Set  $B_n = B_{n-1} \cup \{w(n)\}$ .

Set  $B := \bigcup_{n \in \mathbb{N}} B_n$ .

**Lemma 4.14.**  $0^{k_n} \in L(M_n^B) \iff 0^{k_n} \in L(M_n^{B_{n-1}})$  for all  $n$ .

The oracle is never queried for  $w \in B \setminus B_{n-1}$  during the computation of  $M_n^B$  on  $0^{k_n}$ :

- for  $w = w(n)$  by construction and
- for  $w = w(m)$  with  $m > n$  because  $|w(m)| = k(m) > p_n(k_n)$ .

**Lemma 4.15.**  $S(B) \notin P^B$ .

Otherwise, there would be some  $n \in \mathbb{N}$  with  $S(B) = L(M_n^B)$ . However, this cannot be the case since, by definition,  $0^{k_n} \in S(B)$  if, and only if, there is some  $w$  such that  $|w| = k_n$  and  $w \in B$ . By construction, this is the case if, and only if,  $0^{k_n} \notin L(M_n^{B_{n-1}})$ . This follows from the fact that a word of length  $k_n$  is added to  $B$  if, and only if,  $0^{k_n} \notin L(M_n^{B_{n-1}})$ . By Lemma 4.14,  $0^{k_n} \notin L(M_n^{B_{n-1}})$  is equivalent to  $0^{k_n} \notin L(M_n^B)$ , and therefore,  $S(B) \neq L(M_n^B)$ . Q.E.D.

The problem whether  $\mathcal{C}_1 = \mathcal{C}_2$  remains open for many pairs of complexity classes  $\mathcal{C}_1$  and  $\mathcal{C}_2$ . In most cases, there are oracles  $A$  and  $B$  such that:

$$\mathcal{C}_1^A = \mathcal{C}_2^A \quad \text{and} \quad \mathcal{C}_1^B \neq \mathcal{C}_2^B.$$

It has further been shown that for almost all oracles  $D$ :  $\mathcal{C}_1^D \neq \mathcal{C}_2^D$ . Contradictory relativisations of this kind show that, with respect to the

### 4.3 Relativisations

problem  $\mathcal{C}_1 \neq \mathcal{C}_2$ , proof techniques that 'relativise' (i.e., techniques that are independent of oracles) fail.