Computational Complexity

Handout – Lecture 2

Definition 1 (DTIME)**.** Let $T : \mathbb{N} \to \mathbb{N}$ be a function. A language $L \subseteq \Sigma^*$ is in $\mathrm{DTIME}(T(n))$ if there exists a Turing machine that decides L and that runs in time $O(T(n))$.

Definition 2 (The complexity class P)**.**

$$\mathrm{P} = \bigcup_{c \geq 1} \mathrm{DTIME}(n^c)$$

Definition 3 (The complexity class EXP)**.**

$$\mathrm{EXP} = \bigcup_{c \geq 1} \mathrm{DTIME}(2^{n^c})$$

Definition 4 (The complexity class NP)**.** A language $L \subseteq \Sigma^*$ is in the complexity class NP if there is a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time Turing machine $\mathbb{M}$ (the *verifier*) such that for every $x \in \Sigma^*$:

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

The string $u \in \{0,1\}^{p(|x|)}$ is called a *certificate* for x if $\mathbb{M}(x,u) = 1$.

Definition 5. *Nondeterministic Turing machines* are variants of (deterministic) Turing machines, where a few elements are modified.

- Instead of one halting state q_{halt}, there are two halting states q_{acc} (the *accept state*) and q_{rej} (the *reject state*).
- Instead of a single transition function δ, there are two transition functions δ_1, δ_2.
- At each step, one of δ_1, δ_2 is chosen nondeterministically to determine the next configuration.
- We write $\mathbb{M}(x) = 1$ if there is some sequence of nondeterministic choices such that $\mathbb{M}$ reaches the state q_{acc} on input x.
- The machine $\mathbb{M}$ runs in time $T(n)$ if for every input x and every sequence of nondeterministic choices, $\mathbb{M}$ halts within $T(|x|)$ steps.

Definition 6 (NTIME)**.** Let $T : \mathbb{N} \to \mathbb{N}$ be a function. A language $L \subseteq \Sigma^*$ is in $\mathrm{NTIME}(T(n))$ if there exists a nondeterministic Turing machine that decides L and that runs in time $O(T(n))$.

Definition 7 (The complexity class coNP)**.** A language $L \subseteq \Sigma^*$ is in coNP if $\overline{L} \in \mathrm{NP}$, where $\overline{L} = \{ x \in \Sigma^* \mid x \notin L \}$.

Definition 8 (Polynomial-time reductions)**.** A language $L_1 \subseteq \Sigma^*$ is *polynomial-time reducible* to a language $L_2 \subseteq \Sigma^*$ if there is a polynomial-time computable function $f : \Sigma^* \to \Sigma^*$ (the *reduction*) such that for every $x \in \Sigma^*$ it holds that:

$$x \in L_1 \quad \text{if and only if} \quad f(x) \in L_2.$$

Definition 9 (NP-hardness and NP-completeness)**.** A language $L \subseteq \Sigma^*$ is *NP-hard* if every language $L' \in$ NP is polynomial-time reducible to L.

A language $L \subseteq \Sigma^*$ is *NP-complete* if $L \in$ NP and L is NP-hard.