Theory of Computer Science

G. Röger
Spring Term 2019

University of Basel
Computer Science

Exercise Sheet 10 — Solutions

Exercise 10.1 (1+1+1.5+1.5 marks)

Prove or refute the following statements. In all cases, specify a short proof (2–3 sentences are sufficient).

(a) Let X be an NP-hard problem and Y a problem with $X \leq_{\mathrm{p}} Y$. Then Y is NP-hard.

Solution:

Correct. According to the definition, a problem Y is NP-hard if $A \leq_{\mathrm{p}} Y$ is true for all problems $A \in \mathrm{NP}$. Consider an arbitrary problem $A \in \mathrm{NP}$. Since X is NP-hard, we know $A \leq_{\mathrm{p}} X$, and we conclude with $X \leq_{\mathrm{p}} Y$ (based on the transitivity of reductions) that $A \leq_{\mathrm{p}} Y$.

(b) Let X be an NP-hard problem. If there is a deterministic polynomial algorithm for X, then there also is a deterministic polynomial algorithm for DirHamiltonCycle.

Solution:

Correct. We know: DirHamiltonCycle $\in$ NP (chapter E1). Since X is NP-hard, this implies DirHamiltonCycle $\leq_{\mathrm{p}} X$. With $X \in \mathrm{P}$ we can conclude DirHamiltonCycle $\in$ P with part 1 of the theorem about properties of polynomial reductions (chapter E2).

(c) There are NP-complete problems X and Y where there is a deterministic polynomial algorithm for X but not for Y.

Solution:

Incorrect. NP-complete problems are in NP by definition, and as the previous exercise shows: if one of the problems is in P, then the other one is in P as well.

(d) Let $Y \subseteq \Sigma^*$ be any problem with $Y \neq \emptyset$ and $Y \neq \Sigma^*$. Then $X \leq_{\mathrm{p}} Y$ holds for all $X \in \mathrm{P}$.

Solution:

Correct. Let $z \in Y$ be an arbitrarily chosen "Yes-answer" for Y (possible since $Y \neq \emptyset$) and $z' \in \Sigma^* \setminus Y$ an arbitrarily chosen "No-answer" for Y (possible since $Y \neq \Sigma^*$). Now consider the following mapping f for input words of X:

$$f(w) = \begin{cases} z & \text{if } w \in X \\ z' & \text{otherwise} \end{cases}$$

Based on the choice of z and z' we obviously have $w \in X$ iff. $f(w) \in Y$, and since $X \in \mathrm{P}$ we can compute f in polynomial time. The reduction f thus shows that $X \leq_{\mathrm{p}} Y$.

Exercise 10.2 (Polynomial Reduction, 4 + 1 marks)

A *Hamilton path* is defined analogously to a Hamilton cycle (see chapter E1) with the only difference that we look for a simple path instead of a cycle. More formally: a Hamilton path in a directed graph $\langle V, E\rangle$ is a sequence of vertices $\pi = \langle v_1, \dots, v_n\rangle$ that defines a path ($\langle v_i, v_{i+1}\rangle \in E$ for all $1 \leq i < n$) and contains every vertex in the graph exactly once.

Consider the decision problem DirHamiltonPath:

- *Given:* directed graph $G = \langle V, E \rangle$
- *Question:* Does G contain a Hamilton path?

(a) Prove that DirHamiltonPath is NP-hard. You can use without proof that DirHamiltonCycle is NP-complete.

Solution:

We have to show that

$$\textsc{DirHamiltonCycle} \leq_{\text{p}} \textsc{DirHamiltonPath}.$$

So for any given graph G, we have to construct a directed graph $f(G)$ in polynomial time, so the following holds: G contains a Hamilton cycle iff $f(G)$ contains a Hamilton path.

Let $G = \langle V, E \rangle$ be the given graph. Without loss of generality let $|V| \geq 2$. (The special case $|V| \leq 1$ can easily be treated separately.)

Idea: we chose an arbitrary node $\bar{v} \in V$. We can rewrite every Hamilton cycle of G in a way that $\bar{v}$ is at its start and end, i.e., the cycle has the form $\langle \bar{v}, v_1, v_2, \ldots, v_{n-1}, \bar{v} \rangle$ where $n = |V|$. We now make sure that every Hamilton cycle in G uniquely corresponds to a Hamilton path in $f(G)$, by replacing the node $\bar{v}$ in $f(G)$ with two new nodes: a node $\bar{v}_{\text{start}}$ that only has the outgoing edges of $\bar{v}$ (and no incoming edges) and a node $\bar{v}_{\text{end}}$ that only has the incoming edges of $\bar{v}$ (and no outgoing edges).

Formally: $f(\langle V, E \rangle) = \langle V', E' \rangle$ with

$$\begin{aligned} V' &= (V \setminus \{\bar{v}\}) \cup \{\bar{v}_{\text{start}}, \bar{v}_{\text{end}}\} \\ E' &= \{\langle u, v \rangle \mid \langle u, v \rangle \in E, u \neq \bar{v}, v \neq \bar{v}\} \cup \\ &\quad\ \{\langle \bar{v}_{\text{start}}, v \rangle \mid \langle \bar{v}, v \rangle \in E, v \neq \bar{v}\} \cup \\ &\quad\ \{\langle u, \bar{v}_{\text{end}} \rangle \mid \langle u, \bar{v} \rangle \in E, u \neq \bar{v}\}. \end{aligned}$$

where $\bar{v}$ is an arbitrary element of V and $\bar{v}_{\text{start}}, \bar{v}_{\text{end}}$ are two new vertices.

This function can easily be calculated in polynomial time.

Since the two new nodes only have outgoing and incoming edges respectively, every Hamilton path in $f(G)$ has to start with $\bar{v}_{\text{start}}$ and end with $\bar{v}_{\text{end}}$. In general $\langle \bar{v}, v_1, v_2, \ldots, v_{n-1}, \bar{v} \rangle$ is a Hamilton cycle in G iff $\langle \bar{v}_{\text{start}}, v_1, v_2, \ldots, v_{n-1}, \bar{v}_{\text{end}} \rangle$ is a Hamilton path in $f(G)$:

$\Rightarrow$: the edges $\langle v_i, v_{i+1} \rangle$ exist unchanged in $f(G)$ for all $1 \leq i < n-1$. The edges $\langle \bar{v}_{\text{start}}, v_1 \rangle$ and $\langle v_{n-1}, \bar{v}_{\text{end}} \rangle$ exist in $f(G)$, since there are edges $\langle \bar{v}, v_1 \rangle$ and $\langle v_{n-1}, \bar{v} \rangle$ in G. Thus $\langle \bar{v}_{\text{start}}, v_1, v_2, \ldots, v_{n-1}, \bar{v}_{\text{end}} \rangle$ is a path. The nodes $\bar{v}, v_1, \ldots, v_{n-1}$ are pairwise different. Thus the nodes $\bar{v}_{\text{start}}, v_1, v_2, \ldots, v_{n-1}, \bar{v}_{\text{end}}$ are also pairwise different. The path thus visits every node in $f(G)$ exactly once.

$\Leftarrow$: Analogously.

This proves that f is a reduction.

(b) Is DirHamiltonPath NP-complete? Justify your answer.

Solution:

Yes. For a problem to be NP-complete, it has to be (1) NP-hard, and (2) in NP. We have already shown (1) in exercise (a). (2) is satisfied if there is a non-deterministic polynomial algorithm for the problem.

We can guess a path in polynomial time (as shown in the lecture) and then test whether this path is a Hamilton path (also just like in the lecture) in polynomial time as well.