University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
D. Atalay, S. Faour

Theoretical Computer Science - Bridging Course
Exercise Sheet 9

Due: Tuesday, 16th of December 2025, 12:00 pm

Exercise 1: Class $\mathcal{NPC}$ *(3+4+5 Points)*

Recall: Let L_1, L_2 be languages (problems) over alphabets Σ_1, Σ_2. Then $L_1 \leq_p L_2$ (L_1 is polynomially reducible to L_2), iff a function $f : \Sigma_1^* \to \Sigma_2^*$ exists, that can be calculated in polynomial time and

$$\forall s \in \Sigma_1^* : s \in L_1 \Longleftrightarrow f(s) \in L_2.$$

Language L is called $\mathcal{NP}$-hard, if *all* languages $L' \in \mathcal{NP}$ are polynomially reducible to L, i.e.

$$L \text{ is } \mathcal{NP}\text{-hard} \Longleftrightarrow \forall L' \in \mathcal{NP} : L' \leq_p L.$$

The reduction relation '$\leq_p$' is transitive ($L_1 \leq_p L_2$ and $L_2 \leq_p L_3 \Rightarrow L_1 \leq_p L_3$). Therefore, in order to show that L is $\mathcal{NP}$-hard, it suffices to reduce a known $\mathcal{NP}$-hard problem $\tilde{L}$ to L, i.e. $\tilde{L} \leq_p L$.
Finally a language is called $\mathcal{NP}$-complete ($\Leftrightarrow$: $L \in \mathcal{NPC}$), if

1. $L \in \mathcal{NP}$ and
2. L is $\mathcal{NP}$-hard.

Now, show that the following problems are in $\mathcal{NPC}$

(a) IndependentSet$:= \{\langle G, k\rangle \mid G$ has an independent set of size at least k $\}$, where an independent set is a subset of nodes of G such that no two nodes in the subset share an edge in G.

(b) Clique$:= \{\langle G, k\rangle \mid G$ has a clique of size at least k $\}$.

(c) HittingSet$:= \{\langle \mathcal{U}, S, k\rangle \mid$ universe $\mathcal{U}$ has subset of size at most k that *hits* all sets in $S \subseteq 2^{\mathcal{U}}\}$.

(Bonus) DominatingSet $:= \{\langle G, k\rangle \mid$ *Graph G has a dominating set of size at most k*$\}$*, where a dominating set is a subset of nodes of G such that every node in G is in the subset or has a neighbor in the subset.*

Hint: For all four parts, use the fact that VertexCover $:= \{\langle G, k\rangle \mid$ *Graph G has a vertex cover of size at most k*$\} \in \mathcal{NPC}$*, where a vertex cover is a subset of nodes of G such that every edge of G is incident to a node in the subset.*

For the poly. transformation ($\leq_p$) you have to describe an algorithm (with poly. run-time!) that transforms:
For part (a), an instance $\langle G, k\rangle$ of VertexCover *into an instance $\langle G', k'\rangle$ of* IndependentSet *s.t. a vertex cover of size $\leq k$ in G becomes an independent set of G' of size $\geq k'$ vice versa(!)*

For part (b), an instance $\langle G, k\rangle$ of VertexCover *into an instance $\langle G', k'\rangle$ of* Clique *s.t. a vertex cover of size $\leq k$ in G becomes a clique of G' of size $\geq k'$ vice versa(!)*

For part (c), an instance $\langle G, k\rangle$ of VertexCover *into an instance $\langle \mathcal{U}, S, k'\rangle$ of* HittingSet*, s.t. a vertex cover of size $\leq k$ in G becomes a hitting set of $\mathcal{U}$ of size $\leq k'$ for S and vice versa(!).*

For the bonus, transfrom an instance $\langle G, k\rangle$ of VertexCover *into an instance $\langle G', k'\rangle$ of* DominatingSet *s.t. a vertex cover of size $\leq k$ in G becomes a dominating set of G' of size $\leq k'$ vice versa(!)*
Note that a dominating set is not necessarily a vertex cover ($G = (\{v_1, v_2, v_3, v_4\}, \{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}\})$) has the dominating set $\{v_1, v_4\}$ which is not a vertex cover). Also a vertex cover is not necessarily a dominating set (consider isolated notes).

Exercise 2: Complexity Classes: Big Picture *(2+3+3 Points)*

(a) Why is $\mathcal{P} \subseteq \mathcal{NP}$?

(b) Show that $\mathcal{P} \cap \mathcal{NPC} = \emptyset$ if $\mathcal{P} \neq \mathcal{NP}$.
Hint: Assume that there exists a $L \in \mathcal{P} \cap \mathcal{NPC}$ and derive a contradiction to $\mathcal{P} \neq \mathcal{NP}$.

(c) Give a Venn Diagram showing the sets $\mathcal{P}, \mathcal{NP}, \mathcal{NPC}$ for both cases $\mathcal{P} \neq \mathcal{NP}$ and $\mathcal{P} = \mathcal{NP}$.
Remark: Use the results of (a) and (b) even if you did not succeed in proving those.