University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
Z. Parsaeian

# Theoretical Computer Science - Bridging Course
# Sample Solution Exercise Sheet 9

**Due:** Tuesday, 7th of January 2025, 12:00 pm

## Exercise 1: Class $\mathcal{NPC}$ part 1

Let $L_1, L_2$ be languages (problems) over alphabets $\Sigma_1, \Sigma_2$. 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

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

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

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

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.

(a) Show $\textsc{Clique}{:=}\{\langle G, k\rangle \mid G \text{ has a clique of size at least } k\ \} {\in} \mathcal{NPC}$.

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

For both parts, use the fact that $\textsc{VertexCover} := \{\langle G, k\rangle \mid$ Graph $G$ has a *vertex cover* of size at most $k\} \in \mathcal{NPC}$.

*Hint: 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* 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 (b), 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(!).*

## Sample Solution

We have already shown that Clique and HittingSet belongs in $\mathcal{NP}$, by engineering a deterministic polynomial time verifier for it in exercise 1. Now in order to show that they are also in $\mathcal{NPC}$, we are only left to prove that they are $\mathcal{NP}$-hard problems; and we will do so by reducing a known $\mathcal{NP}$-hard problem (e.g. vertex cover as mentioned in the hint) to both Clique and HittingSet in polynomial time. We demonstrate how in the following..

(a) **Polynomial Reduction of** VertexCover **to** Clique**:** We will create a polynomial time reduction from vertex cover to clique, proving that since vertex cover is $\mathcal{NP}$-hard, clique must also be

$\mathcal{NP}$-hard. For this purpose we define a function $f$ which maps instances $\langle G, k\rangle$ of VERTEXCOVER to instances $\langle G', k'\rangle$ of CLIQUE (as usual we neglect strings that do not represent well-formed instances), and is computable in polynomial time; thus, we define $f(\langle G, k\rangle) = \langle G', k'\rangle := \langle \overline{G}, n-k\rangle$, where $\overline{G}$ is the complement graph of $G$. This means that the reduction takes as an input an undirected graph $G = (V, E)$, where $V$ is a set of nodes and $E$ a set of edges defined over those nodes, as well as a positive integer $k$ and outputs the complement graph $\overline{G} = (V, \overline{E})$ where $V$ is the same set $V$ of $G$, the set of all edges that don't exist in $G$ defined by $\overline{E} = \{(u, v) : u, v \in V, u \neq v, (u, v) \notin E\}$ as well as the positive integer $k' := n - k$.
Moreover, this reduction can be done in polynomial time by generating the complement graph as follows: copy the vertex set $V$ of the input $G$ as is and go through each pair of nodes in $G$ : generate an edge for $\overline{G}$ only if there is no edge between the pair in $G$; as well as outputting the positive integer $n-k$ . All these operations can be done in polynomial time.

It remains to prove the equivalency

$$\langle G, k\rangle \in \textsc{VertexCover} \iff f(\langle G, k\rangle) = \langle \overline{G}, n-k\rangle \in \textsc{Clique}.$$

And the proof is the following:

$\Longrightarrow$**:** Suppose that $G$ has a vertex cover $S \subseteq V$ of size at most $k$ (a yes instance of VERTEXCOVER). Then for all $u, v \in V$, if $(u, v) \in E$, then $u \in S$ or $v \in S$, or both, by definition of the vertex cover that it needs to cover all edges. Now consider $S' := V \backslash S$ . Clearly $|S'|$ is at least $n-k$. Also, notice that $S'$ is an independent set of $G$ i.e. there are no edges connecting any two nodes in $S'$, since there cannot exist an edge $\{u, v\}$ in $G$ where $u \in S'$ and $v \in S'$, else we reach a contradiction to that $S$ is a vertex cover. Moreover, if we consider the graph $\overline{G} = (V, \overline{E})$, we deduce that for all $u, v \in S'$, $\{u, v\} \in \overline{E}$. Therefore, $S'$ is a clique in $\overline{G}$ of size at least $n-k$ (a yes instance of CLIQUE).

$\Longleftarrow$**:** Suppose $\overline{G} = (V, \overline{E})$ has a clique $S \subseteq V$ of size at least $n-k$. So all nodes in the clique $S$ are connected to each other by an edge in $\overline{E}$. Hence, $S$ makes up an independent set in $G = (V, E)$. Thus, $V \backslash S$ is a vertex cover in $G$, else there exists an edge $\{u, v\} \in E$ which is not covered by $V \backslash S$ i.e. both $u$ and $v$ are not in $V \backslash S$, thus $u, v \in S$. This is a contradiction since $u, v \in S$, $\{u, v\} \in E$ and $S$ is an independent set in $G$. Hence, $G$ has a vertex cover that is $V \backslash S$ of size at most $k$.
**Alternatively for the backward direction**: Suppose that $G$ has no vertex cover of size at most $k$ (a no instance of VERTEXCOVER ). Hence, $\overline{G}$ will have no clique of size at least $n-k$; otherwise, $G$ will have an independent set $S$ of size at least $n-k$ and thus using the above argument $V \backslash S$ is a vertex cover in $G$ of size at most $k$, which is a contradiction. Hence, $\overline{G}$ has no clique of size at least $n-k$ (a no instance of CLIQUE).

Therefore, we have shown that VERTEXCOVER can be reduced in polynomial time to CLIQUE, and hence CLIQUE is $\mathcal{NP}$-hard.

In summary: CLIQUE $\in \mathcal{NP}$ and $\mathcal{NP}$-hard, thus CLIQUE $\in \mathcal{NPC}$.

(b) **Polynomial Reduction of** VERTEXCOVER **to** HITTINGSET**:** We will create a polynomial time reduction from vertex cover to hitting set, proving that since vertex cover is $\mathcal{NP}$-hard, hitting set must also be $\mathcal{NP}$-hard. We define a function $f$ that can be computed in polynomial time and transforms an instance $\langle G, k\rangle$ of VERTEXCOVER into an instance $\langle \mathcal{U}, S, k'\rangle$ of HITTINGSET; thus for graph $G = (V, E)$, we define $f(\langle G, k\rangle) = \langle \mathcal{U}, S, k'\rangle := \langle V, E, k\rangle$. This means that the reduction takes as input an undirected graph $G = (V, E)$, where $V$ is a set of nodes and $E$ a set of edges defined over those nodes, as well as a positive integer $k$ and outputs the set $V$, the collection $E = \{e_1, e_2, \ldots, e_n\}$ of subsets of $V$ and the positive integer $k$. Moreover, this reduction takes time linear in the size of the input (all it does is copy the input to the output), therefore it takes polynomial time. It remains to prove the equivalency

$$\langle G, k\rangle \in \textsc{VertexCover} \iff f(\langle G, k\rangle) = \langle V, E, k\rangle \in \textsc{HittingSet},$$

where $G = (V, E)$. This means we have to prove that

"$G$ has a vertex cover of size at most $k$" $\iff$ "$(V, E)$ has a hitting set of size at most $k$"

We prove this in the following:

"$G$ has a vertex cover of size at most $k$" $\Leftrightarrow$
$\exists V' \subseteq V : |V'| \leq k$ and $\forall$ edge $e_i = \{u_i, v_i\} \in E, u_i \in V'$ or $v_i \in V'$ $\Leftrightarrow$
$\exists V' \subseteq V : |V'| \leq k$ and $\forall$ subset $e_i$ in collection $E$ $\exists c \in e_i : c \in V'$ $\Leftrightarrow$
"$(V, E)$ has a hitting set of size at most $k$"

Therefore, we have shown that VertexCover can be reduced in polynomial time to HittingSet, and hence HittingSet is $\mathcal{NP}$-hard.

In summary: HittingSet $\in \mathcal{NP}$ and $\mathcal{NP}$-hard, thus HittingSet $\in \mathcal{NPC}$.

*Note:* one might notice that this reduction was rather straightforward. This makes sense, since vertex cover is a special version of hitting set, where each subset $S_i$ in the collection $S$ has exactly two elements of $\mathcal{U}$. Obviously, no problem can be harder than its generalization and since vertex cover is $\mathcal{NP}$-hard, hitting set (as a generalization of vertex cover) must also be $\mathcal{NP}$-hard.

## Exercise 2: Class $\mathcal{NPC}$ part 2

1. Given a set $U$ of $n$ elements ('universe') and a collection $S \subseteq \mathcal{P}(U)$ of subsets of $U$, a selection $C_1, \ldots, C_k \in S$ of $k$ sets is called a *set cover* of $(U, S)$ of size $k$ if $C_1 \cup \ldots \cup C_k = U$.

   Show that the problem

   $$\text{SetCover} := \{\langle U, S, k\rangle \,|\, U \text{ is a set, } S \subseteq \mathcal{P}(U) \text{ and there is a set cover of } (U, S) \text{ of size } k\}$$

   is NP-complete.

   You may use that

   $$\text{DominatingSet} = \{\langle G, k\rangle \mid G \text{ has a dominating set with } k \text{ nodes}\}.$$

   is NP-complete.

2. Show DominatingSet := $\{\langle G, k\rangle \,|\, \text{Graph } G \text{ has a dominating set of size at most } k\} \in \mathcal{NPC}$.

   Use that VertexCover := $\{\langle G, k\rangle \mid \text{Graph } G \text{ has a vertex cover of size at most } k\} \in \mathcal{NPC}$.

   *Remark: A* ***vertex cover*** *is a subset of nodes of $G$ such that every edge of $G$ is incident to a node in the subset.*

   *Hint: Transform a Graph $G$ into a Graph $G'$ such that a vertex cover of $G$ will result in a dominating set $G'$ and 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).*

## Sample Solution

1. 
   - SetCover is in NP: Guess a collection $C_1, \ldots, C_k \in S$ of $k$ sets from $S$. Go through all elements of $U$ and check if it is in one of the $C_i$. This takes polynomial time.

     SetCover is NP-hard: We reduce DominatingSet to SetCover. Let $G = (V, E)$ be a graph and $k$ an integer. We define a SetCover instance in the following way: We choose $V$ to be the universe, i.e., $U = V$ and $S := \{\Gamma_G(v) \mid v \in V\}$ where $\Gamma_G(v)$ consists of the vertex $v$ and all vertices adjacent to $v$ in $G$. This conversion takes polynomial time. Then $\Gamma_G(v_1), \ldots, \Gamma_G(v_k)$ is a set cover of $(U, S)$ iff $v_1, \cdots, v_k$ is a dominating set of $G$. Hence, $\langle U, S, k\rangle \in$ SetCover iff $\langle G, k\rangle \in$ DominatingSet.

- To prove DOMINATINGSET in $\mathcal{P}$, we would need to find a constant $c$, and an associated $O(n^c)$ algorithm, which would decide on änyïnstance $(G, k)$ , whatever $k$ is However, if we use the same brute force algorithm in exercise 1 to solve DOMINATINGSET, once we take the instance $(G, k)$ to be checked, then it will run on $O(n^c)$ for some constant $c$, but here we are choosing $c$ after we have seen $k$, for arbitrary $k$. So as $k$ increases it approaches to $n/2$, then using the same brute force algorithm will yield in an exponential of order $n/2$ time complexity, and this does not prove that DOMINATINGSET in $\mathcal{P}$.

2. **Guess and Check:** we show that DOMINATINGSET $\in \mathcal{NP}$.

   Consider the following verifier for DOMINATINGSET on input $\langle\langle G, k\rangle, D\rangle$, that verifies in polynomial time that $G$ has a dominating set of size at most $k$, where the idea of the certificate $D$ is the dominating set. Let $G = (V, E)$, where $n := |V|, m := |E|$.
   The verifier first tests if $D$ has at most $k$ different nodes from $G$ with $O(|D| + |D| \cdot n + |D|^2)$ comparisons (similar to the CLIQUE problem in a prev. sheet), then it tests if all nodes are in $D$ or adjacent to a node in $D$ in $O(n(|D| + m))$ comparisons (or you can say $O(nm)$ comparisons, since to do this second test $|D| \leq k \leq n$). If both these tests pass, accept; else reject. Since the certificate has polynomial length in the input size, therefore the total running time is polynomial in the input size. So DOMINATINGSET has a polynomial time verifier. Therefore, DOMINATINGSET is in $\mathcal{NP}$.

   **Polynomial reduction of** VERTEXCOVER **to** DOMINATINGSET**:** we show that DOMINATINGSET is $\mathcal{NP}$-hard.
   We define a function $f$ that can be computed in polynomial time and transforms an instance $\langle G, k\rangle$ of DOMINATINGSET into an instance $f(\langle G, k\rangle) = \langle G', k\rangle$, such that $G$ has a vertex cover of size at most $k$, iff $G'$ has a vertex cover of size at most $k$, i.e.,

   $$\langle G, k\rangle \in \text{VERTEXCOVER} \iff f(\langle G, k\rangle) = \langle G', k\rangle \in \text{DOMINATINGSET}.$$

   For $G = (V, E)$ we construct $G' = (V', E')$ as follows. Initially, we set $V' := V, E' := E$. For each edge $\{u, v\} \in E$ we add an additional node $w$ to $V'$ and add the edges $\{u, w\}, \{v, w\}$ to $E'$ (i.e., $G'$ has a triangle with nodes $u, v, w$). Furthermore we remove all isolated nodes from $V'$. The construction of $G'$ can be accomplished, by generating $V'$ and $E'$, in $O((m + n) + nm + m)$ (i.e. $\mathcal{O}(nm)$) comparisons. Indeed, it takes $O((m + n) + nm)$ comparisons to generate $V'$, since we add $\mathcal{O}(m)$ new nodes alongside the old ones and remove at most $\mathcal{O}(n)$ nodes ( for each node in $V$ we can check if it is isolated in $O(m)$ comparisons); moreover, we can generate $E'$ in $O(m)$ comparisons as we add at most $\mathcal{O}(m)$ new edges ( for each corresponding edge in $E'$, we add 2 new edges). It remains to prove the equivalency stated above.

   $\Longrightarrow$**:** Let $(G, k)$ be such, that $G$ has a vertex cover $C$ of size at most $k$. Let $D := C \cap V'$ which corresponds to $C$ but without isolated nodes. We have $|D| \leq |C| \leq k$ since $D$ is a subset of $C$.

   It remains to show that $D$ is a dominating set. We know that for every edge $\{u, v\} \in E$ either $u \in C$ or $v \in C$ (or both). Therefore, every node $w$ that was added to $V'$ during the construction due to an edge $\{u, v\} \in E$ is adjacent to either $u \in D$ or $v \in D$. All other nodes in $v \in V'$ have an incident edge $\{u, v\} \in E \subset E'$ since we removed isolated nodes from $V'$. Therefore $v \in C$ (and thus $v \in D$), or $v$ is adjacent to a node $u$ in $C$ (hence dominated by one in $D$).

   $\Longleftarrow$**:** Let the transformed instance $f(\langle G, k\rangle) = \langle G', k'\rangle$ be such that $G'$ has a dominating set $D$ with $|D| \leq k$. We show that we can construct a vertex cover $C$ of size at most $k$ in the original graph $G$ from $D$. Let $\{u, v\}$ be an arbitrary edge of $G$. Due to the way $G'$ was constructed, it has a triangle formed by the nodes $u, v, w$ where $w$ is only connected to $u$ and $v$.

   This means that at least one of the three cases holds: $w$ is dominated by $u \in D$ or by $v \in D$ or it holds that $w \in D$. In the first two cases we add $u$ or $v$ respectively to $C$ (whichever was in $D$). In the third case we simply add one of the two nodes $u$ or $v$ instead. In all cases $\{u, v\}$ is covered. Since we add at most $|D| \leq k$ nodes to $C$ it holds that $|C| \leq k$.

In summary: DominatingSet $\in \mathcal{NP}$ and $\mathcal{NP}$-hard, thus DominatingSet $\in \mathcal{NPC}$.