ETH zürich
Departement Informatik

Theoretische Informatik

Prof. Dr. Juraj Hromkovič
Dr. Hans-Joachim Böckenhauer
`https://courses.ite.inf.ethz.ch/theo_inf_21`

Exemplary Solutions – Sheet 9

Zürich, December 3, 2021

Solution to Exercise 25

(a) The idea behind our reduction is to interpret SCP as a generalization of VC.

The input for VC is a graph $G = (V, E)$ and a natural number k. We transform every such input to an input for SCP. Let $E_v := \{e \in E \mid v \text{ is incident to } e\}$ be the set of edges incident to the vertex v and let $\mathcal{S}_G = \{E_v \mid v \in V\}$. Then

$$(E, \mathcal{S}_G, k)$$

is our input for SCP. Hence, for every vertex v, the set family $\mathcal{S}_G$ contains the set of all edges incident to v. Note that two vertices v and w can have the same set $E_v = E_w$ of incident edges; in this case E_v is contained in $\mathcal{S}$ only once.

It is clear that the transformation can be computed in polynomial time. It remains to show that (G, k) is a yes-instance for VC if and only if $(E, \mathcal{S}_G, k)$ is a yes-instance for SCP.

Suppose that there exists a vertex cover of size k in G. Then there exists a set of k vertices $\{v_1, v_2, \dots, v_k\}$ that cover all edges in G. The covered edges can also be expressed as $\bigcup_{i=1}^{k} E_{v_i}$. Every set E_{v_i} is in $\mathcal{S}_G$. Hence, there exists a set cover of size at most k for $(E, \mathcal{S}_G, k)$.

Suppose that there exists a set cover of size k for $(E, \mathcal{S}_G, k)$. Then there exists some $\mathcal{C} \subseteq \mathcal{S}_G$ such that $|\mathcal{C}| = k$ and $\bigcup_{S \in \mathcal{C}} S = E$. Then the set of k vertices that are mapped to the sets in $\mathcal{C}$ by our reduction is incident to all edges in E. Hence, these vertices form a vertex cover of size k in G. If multiple vertices have the same set of incident edges and thus get mapped to the same set in $\mathcal{C}$, then it clearly suffices to choose one such vertex for every such set in $\mathcal{C}$.

(b) Let $(X, \mathcal{S}, k)$ be an input for SCP such that $X = \{x_1, x_2, \dots, x_n\}$, $\mathcal{S} = \{S_1, S_2, \dots, S_m\}$, and k is a natural number. We transform every such input to an input (G, k) for DS, $G = (V, E)$. The vertices of G are $V = V_X \cup V_{\mathcal{S}}$, where $V_X = \{x_1, x_2, \dots, x_n\}$ and $V_{\mathcal{S}} = \{s_1, s_2, \dots, s_m\}$, i.e., V_X corresponds to the elements in X and $V_{\mathcal{S}}$ corresponds to the sets in $\mathcal{S}$. The edges in G between the vertices from $V_{\mathcal{S}}$ form a clique. Moreover, the following equivalence holds:

$$\{x_i, s_j\} \in E \iff x_i \in S_j \ ,$$

i. e., there exists an edge between a vertex in V_X and a vertex in $V_{\mathcal{S}}$ if and only if the element $x_i \in X$ is contained in the set $S_j \in \mathcal{S}$, i. e., the element x_i is covered by S_j. There are no edges between the vertices in V_X. It is clear that the transformation can be computed in polynomial time.

Now let $(X, \mathcal{S}, k)$ be an input for SCP such that X has a set cover from $\mathcal{S}$ of size k; let $\mathcal{C}$ be one such set cover. The corresponding vertices in $V_{\mathcal{S}}$ are a dominating set D of the same size. This can be justified as follows. All vertices in $V_{\mathcal{S}}$ are trivially dominated because $V_{\mathcal{S}}$ is a clique. Moreover, every vertex in V_X is adjacent to a vertex in D because every element in X is contained in at least one set in $\mathcal{C}$.

On the other hand, let D be a dominating set of size k in G. If $D \subseteq V_{\mathcal{S}}$, then the corresponding sets in $\mathcal{S}$ are a set cover of X of the same size. This can be justified as before. If there is a vertex $x \in D \setminus V_{\mathcal{S}}$, then we can modify D by replacing x by an adjacent vertex in $V_{\mathcal{S}}$. The two vertices are still dominated and no other vertex in V_X looses x as its dominating vertex because the vertices V_X form an independent set (there is no edge within V_X) while the vertices in $V_{\mathcal{S}}$ are certainly dominated (because $V_{\mathcal{S}}$ is a clique).

Solution to Exercise 26

Let ϕ be an input for 3SAT, i.e., a formula $\phi = C_1 \wedge \ldots \wedge C_m$ in 3CNF with the clauses $C_1, \ldots, C_m$ over the variables in $X = \{x_1, \ldots, x_n\}$. We transform ϕ to an input ψ for E3SAT. We first modify ϕ so that no variable occurs multiple times in a clause. To this end, we drop all repeated occurrences of a literal in a clause. This does not affect the satisfiability of the formula. If a clause contains a variable y and its negation $\overline{y}$ as literals, then the clause is satisfied for every assignment. Hence, we drop such clauses with no effect on the satisfiability of the formula. In the following, we assume that every clause in ϕ contains every variable at most once. Then the literals in a clause consist of pairwise distinct variables.

Now we construct ψ from ϕ as follows: all clauses in ϕ that contain three literals are left unchanged. A clause $C_i = (l_{i,1} \vee l_{i,2})$ gets replaced by the two clauses $C_{i,1} = (l_{i,1} \vee l_{i,2} \vee y_i)$ and $C_{i,2} = (l_{i,1} \vee l_{i,2} \vee \overline{y_i})$, where y_i is a fresh variable that does not occur anywhere else in ψ. A clause $C_i = (l_i)$ gets replaced by the four clauses $C_{i,1} = (l_i \vee y_{i,1} \vee y_{i,2})$, $C_{i,2} = (l_i \vee y_{i,1} \vee \overline{y_{i,2}})$, $C_{i,3} = (l_i \vee \overline{y_{i,1}} \vee y_{i,2})$, and $C_{i,4} = (l_i \vee \overline{y_{i,1}} \vee \overline{y_{i,2}})$, where $y_{i,1}$ and $y_{i,2}$ are two fresh variables that do not occur anywhere else in ψ. It is clear that this construction can be computed in polynomial time and that the additional conditions of E3SAT compared to 3SAT are met.

Next we show that ϕ is satisfiable if and only if ψ is satisfiable.

Let α be a satisfying assignment for ϕ. Then α assigns at least one literal in each of the clauses $C_1, \ldots, C_m$ to 1. This way, an arbitrary extension of α satisfies every clause in ψ because these clauses may only contain additional literals.

Let β be a satisfying assignment for ψ. Then β satisfies all clauses in ψ. We show that the restriction of β to the variables in X satisfies ϕ. Every clause of length 3 in ϕ was left unchanged in ψ and thus it is satisfied by β. Instead of the clause $C_i = (l_{i,1} \vee l_{i,2})$ of length 2 in ϕ, there are the two clauses $C_{i,1}$ and $C_{i,2}$ in ψ. The variable y_i occurs in one of these two clauses positively and in the other one negatively. Let us first suppose that $\beta(y_i) = 1$. Because β satisfies every clause in ψ by assumption, the clause $C_{i,2}$ is satisfied as well although the literal $\overline{y_i}$ is assigned to 0. Hence, $C_i = (l_{i,1} \vee l_{i,2})$ must be satisfied by β. The case of $\beta(y_i) = 0$ is analogous.

Instead of a clause C_i of length 1 in ϕ, there are four clauses in ψ so that every assignment of the two new variables $y_{i,1}$ and $y_{i,2}$ only satisfies one of these clauses if C_i is satisfied. Altogether, β satisfies all clauses in ϕ.

Solution to Exercise 27

We first show that LARGE-CLIQUE is NP-complete. It has been shown in the lecture that CLIQUE is in NP. Because an instance of LARGE-CLIQUE is also an instance of CLIQUE, the same justification shows that LARGE-CLIQUE is in NP (in addition, one only needs to check the condition $k \geq |V|/3$).

To show that LARGE-CLIQUE is NP-hard, we show that E3SAT $\leq_p$ LARGE-CLIQUE. To this end, we use the same transformation of a formula in E3CNF to an instance of CLIQUE (G, k) as in the proof of SAT $\leq_p$ CLIQUE in the textbook. For a formula ϕ with m clauses, the resulting graph $G = (V, E)$ has exactly $3m$ vertices. By construction, we have $k = m = |V|/3$. The claim follows immediately using the proof of Lemma 6.9 from the textbook.

Now we show that VERY-LARGE-CLIQUE is in P. For a graph with n vertices, we can simply enumerate all $\binom{n}{n-3} + \binom{n}{n-2} + \binom{n}{n-1} + \binom{n}{n} \in O(n^3)$ possible subsets of at least $n-3$ vertices and check if they form a clique. These cheks can be carried out in time $O(n^2)$ per subset. Hence, we obtain a polynomial total running time in $O(n^5)$.