Technische Universität München (I7)
Prof. J. Křetínský / Dr. C.H. Broadbent

Summer term 2016
18. Juli 2016

Solution

Computational Complexity – Homework 9

Discussed on 13.6.2016.

Exercise 9.1

Given an undirected graph $G = (V, E)$ we call $C \subseteq V$ a *(vertex) cover* of G if

$$\forall (u, v) \in E : u \in C \vee v \in C.$$

In the following, we want to study the relation between several decision and function problems related to vertex cover. These are:

- $\mathrm{VC}_D := \{\langle G, k\rangle \mid G$ has a vertex cover of size at most $k\}$.
- $\textsc{MinVC}_D := \{\langle G, k\rangle \mid G$ has a minimal vertex cover of size exactly $k\}$.
- $\textsc{MinSolVC}_D := \{\langle G, C\rangle \mid C$ is a minimal vertex cover of $G\}$.
- Calculate the minimal size $\mathrm{minVC}(G)$ of a vertex cover of G.
- Calculate a minimal vertex cover $\mathrm{MinVC}(G)$ of G.

Show:

(a) $\textsc{MinSolVC}_D \leq_p \textsc{MinVC}_D$.

(b) $\textsc{MinSolVC}_D \leq_p \overline{\mathrm{VC}}_D$

(c) MinVC_D is **DP**-complete

(d) $\mathrm{VC}_D \leq_p \mathrm{MinVC}_D$ and $\overline{\mathrm{VC}}_D \leq_p \mathrm{MinVC}_D$.

Remark: You only have to show that such reductions exist.

(e) Assume that $\mathrm{minVC}(G)$ can be calculated in time $T(|G|)$.

Give bounds on the time needed to decide $\textsc{MinVC}_D$ and $\textsc{MinSolVC}_D$, resp. calculate $\mathrm{MinVC}(G)$.

In particular, show that, if $T(n)$ is polynomial, then so are the other bounds.

(f) Analogously to (e), give time bounds on the considered problems assuming that $\langle G, k\rangle \in \mathrm{MinVC}_D$ (resp. $\langle G, k\rangle \in \mathrm{VC}_D$) can be decided in time $T(|G|)$.

(g) If $\mathrm{MinVC}_D \leq_p \mathrm{MinSolVC}_D$, then $\mathbf{PH} \subseteq \mathbf{\Sigma}_1^p$.

Solution:

(a) Consider the following reduction $f(\langle G, k\rangle)$ assuming input $\langle G, C\rangle$.

- If C is not a vertex cover, then output $f(\langle G, k\rangle) := \langle G, -1\rangle$.
- Else, output $f(\langle G, k\rangle) := \langle G, |C|\rangle$.

(b) We can reduce $\textsc{MinSolVC}_D$ to $\overline{\mathrm{VC}}_D$ in polynomial-time as:

$$\langle G, C\rangle \in \textsc{MinSolVC}_D \text{ iff } C \text{ is a vertex cover of } G \wedge \langle G, |C|-1\rangle \in \overline{\mathrm{VC}_D}.$$

(c) We first show that MinVC_D is in **DP**:

$$\langle G, k\rangle \in \mathrm{MinVC}_D \text{ iff } \langle G, k\rangle \in \mathrm{VC}_D \wedge \langle G, k-1\rangle \in \overline{\mathrm{VD}}_D.$$

Define therefore $L = \{\langle G, i\rangle \mid \langle G, i-1\rangle \in \mathrm{VC}_D\}$, then:

$$\langle G, k\rangle \in \mathrm{MinVC}_D \text{ iff } \langle G, k\rangle \in \mathrm{VC}_D \wedge \langle G, k\rangle \in \overline{L}.$$

Obviously, $\mathrm{VC}_D, L \in \mathbf{NP}$.

In the lecture, you have seen that C is a cover of G iff $V \setminus C$ is an independent set of G. So:

$$\begin{array}{ll} & \langle G, k\rangle \in \textsc{MinVC}_D \\ \text{iff} & G \text{ has a minimal vertex cover } C \text{ of size exactly } k \\ \text{iff} & G \text{ has a maximal independent set } V \setminus C \text{ of size exactly } k \\ \text{iff} & \text{the largest independent set of } G \text{ has size exactly } k \\ \text{iff} & \langle G, |V|-k\rangle \in \textsc{ExactIndSet} \end{array}$$

So, $\textsc{MinVC}_D$ and $\textsc{ExactIndSet}$ are equivalent w.r.t. polynomial-time reductions. As the latter is **DP**-complete, so is the former.

(d) As $\textsc{MinVC}_D$ is **DP**-complete and $\mathbf{NP} \cup \mathbf{coNP} \subseteq \mathbf{DP}$, these reductions have to exist.

(e)
- Given $\langle G, k\rangle$, we calculate $\mathrm{minVC}(G)$ in time $T(|G|)$ and check that $k = \mathrm{minVC}(G)$. This amounts to run time of $T(|G|) + |G|$.
- Given $\langle G, C\rangle$, we fist check in time $|E|$ that C is a vertex cover, then we decide $\langle G, |C|\rangle \in \mathrm{MinVC}_D$ in time at most $T(|G|) + |G|$. In total $T(|G|) + 2|G|$.
- For a (undirected) graph $G = (V, E)$ and a node $v \in V$, define $G - v$ as the graph we obtain from G be removing v and all edges connected to v. Note that vertex covers of G resp. $G - v$ can be transformed to a cover of the other graph by simply removing resp. adding v to the cover. In particular, if G has a minimal cover C, then for every $v \in C$ it has to hold that $C \setminus \{v\}$ is a minimal cover of $G - v$. This leads to the following algorithm:

  Given $\langle G\rangle$, we calculate $\mathrm{minVC}(G)$ in time $T(|G|)$ and set $C := \emptyset$. Assume that $V = \{1, 2, \ldots, n\}$, i.e., fix some total order on V. Then for $v = 1, 2, \ldots, n$ do:

  - Calculate $k' := \mathrm{minVC}(G - v)$.
  - If $k' < k$, i.e., $k' = k - 1$, set $k := k'$, $G := G - v$, and $C := C \cup \{v\}$.

We consider every node at most once, i.e., remove every edge at most twice, leading to an upper bound of $|V|T(|G|) + |E|$.

So, if $T(n)$ is polynomial, all the problems can be decided, resp. calculated in polynomial time.

Remark: Obviously, we can calculate $\mathrm{minVC}(G)$ in time linear in $|G|$ from $\mathrm{MinVC}(G)$. So, the same holds when we start from $\mathrm{MinVC}(G)$.

(f) We first show how the time $T'(n)$ for deciding VC_D can be bounded by the running time $T(n)$ of MinVC_D: We have already seen in (d) that there exists a polynomial-time reduction r from VC_D to MinVC_D. Let T_r be the time needed to compute the reduction. So, $T'(|G|) \leq T(T_r(|G|))$. In particular, if T is polynomial, then also T' is polynomial.

Recall that we can also compute $\mathrm{minVC}(G)$ by using VC_D as an oracle for the binary search on the interval $[0, |V|]$. Hence, $\mathrm{minVC}(G)$ can be computed in time $\mathcal{O}(T'(|G|) \cdot \log |V|)$.

So, together with (e) it follows that: if either $T(n)$ or $T'(n)$ is polynomial, again all the other problems can also be computed/decided in polynomial time.

(g) If $\textsc{MinVC}_D \leq_p \textsc{MinSolVC}_D$, then also $\textsc{MinVC}_D \leq_p \overline{\mathrm{VC}}_D$ by (b) and transitivity of $\leq_p$. So, $\overline{\mathrm{VC}}_D$ is $\mathbf{DP}$-complete, in particular, this means $\mathrm{VC}_D \leq_p \overline{\mathrm{VC}}_D$ which implies $\mathbf{NP} = \mathbf{coNP}$ and, subsequently, $\mathbf{\Sigma}_1^p = \mathbf{\Sigma}_2^p = \mathbf{PH}$.

This also shows that if either $\mathrm{MinVC}_D \leq \mathrm{VC}_D$ or $\mathrm{MinVC}_D \leq \overline{\mathrm{VC}}_D$, then again the $\mathbf{NP} = \mathbf{coNP}$.

So, probably MinVC_D is indeed harder to solve than the other two decision problems.

What remains is to decide if one should assume that $\mathrm{MinSolVC}_D$ is indeed easier to solve than $\overline{\mathrm{VC}}_D$, or if the two problems are equivalent w.r.t. polynomial-time reductions, i.e., if $\overline{VC}_D \leq_p \mathrm{MinSolVC}_D$ also holds.

<u>Exercise 9.2</u>

Let $\Phi = \{\phi_1, \ldots, \phi_m\}$ be a set of m Boolean expressions in the variables $x_1, \ldots, x_n$ with the restriction that every expression involves at most 3 of these n variables.

Assume we choose a truth assignment u uniformly at random from $\{0,1\}^n$. Denote then by $\Pr[\phi_i]$ the probability that u satisfies ϕ_i.

(a) Show that $\Pr[\phi_i]$ can be calculated in time polynomial in the length of ϕ_i.

(b) Let N be the random variable which counts the number of expressions ϕ_i satisfied by the random assignment u. Show that

$$\mathbb{E}[N] = \sum_{i=1}^{m} \Pr[\phi_i]\,.$$

(c) We write $\mathbb{E}[N \mid u_1 = 0]$ for the expected number of expressions satisfied by a random assignment u which assigns 0 to x_1. Similarly, define $\mathbb{E}[N \mid u_1 = 1]$. Show:

$$\mathbb{E}[N] = \frac{1}{2} \cdot (\mathbb{E}[N \mid u_1 = 0] + \mathbb{E}[N \mid u_1 = 1]).$$

(d) Show that there is always a value b s.t. $\mathbb{E}[N \mid u_1 = b] \geq \mathbb{E}[N]$.

(e) Give now a polynomial-time algorithms which computes an assignment which satisfies at least $\mathbb{E}[N]$ expressions of Φ.

Solution:

(a) By definition, we have

$$\Pr[\phi_i] = \frac{\#\{u \in \{0,1\}^n \mid \phi_i(u) = 1\}}{2^n}.$$

Let $\mathrm{Var}(\phi_i)$ be the set of variables appearing in ϕ. By assumption, we have $\mathrm{Var}(\phi_i) \leq k := 3$ for all $i = 1, 2, \ldots, n$. As the truth value of ϕ_i is completely determined by any truth assignment for $\mathrm{Var}(\phi_i)$, we also have

$$\Pr[\phi_i] = \frac{\#\{u \in \{0,1\}^{\mathrm{Var}(\phi_i)} \mid \phi_i(u) = 1\}}{2^{|\mathrm{Var}(\phi_i)}}.$$

We there only need to evaluate every constraint ϕ_i for at most $2^k = 8$ assignments where evaluating a constraint ϕ_i can obviously be done in time polynomial in $|\phi_i|$.

(b) We may consider the constraints ϕ_i also as random variables, i.e., $\phi_i(u) = 1$ for an event $u \in \{0,1\}^n$ if u is a satisfying assignment for ϕ_i. Hence,

$$N = \sum_{i=1}^{n} \phi_i \text{ and by linearity of } \mathbb{E}\ \mathbb{E}[N] = \sum_{i=1}^{n} \mathbb{E}[\phi_i].$$

Note that

$$\mathbb{E}[\phi_i] = 0 \cdot \Pr[\phi_i = 0] + 1 \cdot \Pr[\phi_i = 1] = \Pr[\phi_i]\,.$$

Hence, we can calculate $\mathbb{E}[N]$ in time polynomial in $|\Phi|$.

Remark: Recall that for any Bernoulli random-variable $X \sim \mathrm{Bin}(1,p)$ we have $\mathbb{E}[X] = \Pr[X = 1] = p$.

(c) Intuitively, as we can simply separately consider the two cases $u_1 = 0$ and $u_1 = 1$ and first take the average for each of these cases:

$$\begin{aligned}
\mathbb{E}[N] &= \textstyle\sum_{u \in \{0,1\}^n} N(u)\Pr[u] \\
&= \textstyle\sum_{u \in 0\{0,1\}^{n-1}} N(u)\Pr[u \mid u_1 = 0]\Pr[u_1 = 0] + \sum_{u \in 1\{0,1\}^{n-1}} N(u)\Pr[u|u_1 = 1]\Pr[u_1 = 1] \\
&= \mathbb{E}[N|u_1 = 0] \cdot 1/2 + \mathbb{E}[N|u_1 = 1] \cdot 1/2.
\end{aligned}$$

(d) Otherwise, the average would be smaller than $\mathbb{E}[N]$.

(e) We now know that in time polynomial in $|\Phi|$ we can always determine a value b for x_1 s.t.

$$\mathbb{E}[N_\Phi] \leq \mathbb{E}[N_\Phi \mid u_1 = b] = \mathbb{E}[N_{\Phi[x_1:=b]}],$$

i.e., if we substitute b for x_1 in Φ and simplify, the expected number of simultaneously satisfied constraints in the resulting constraint system $\Phi[x_1 := b]$ does not decrease. In particular, after n steps we have determined an assignment which satisfies at least $\mathbb{E}[N]$ constraints.

Exercise 9.3

The decision version of the *traveling salesman problem* (short TSP) is defined as follows:

> Given distances $d_{ij} \geq 0$ between n cities and a bound $B \geq 0$, decide if there is a tour of the cities of length at most B.

We denote the corresponding decision problem by TSP_D, i.e.,

$$\langle d_{1,1}, \ldots, d_{n,n}, B \rangle \in \text{TSP}_D \text{ iff there is TSP-tour w.r.t. } (d_{ij}) \text{ of length at most } B.$$

A *Hamilton path* in an undirected graph $G = (V, E)$ is a path in G which visits every node exactly once. The corresponding decision problem $\textsc{HamiltonPath}_D$ is known to be **NP**-complete.

(a) Show that the following is polynomial-time reduction from HP_D to TSP_D:

> Given $G = (V, E)$ with $n = |V|$ and assume that $V = \{1, 2, \ldots, n\}$. Set $d_{i,j} := 1$ if the nodes i and j are connected by some edge, otherwise $d_{i,j} := n + 1$. Further, set $B := 2n$.

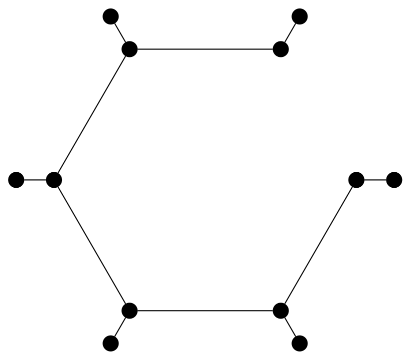
(b) We call a TSP-instance $\langle d_{1,1}, \ldots, d_{n,n}, B \rangle$ *metric* if $d_{ij} \leq d_{ik} + d_{kj}$ holds for all i, j, k.

- Give an example of a graph G where the TSP-instance produced by the reduction above is not metric.
- Modify the reduction such that it always yields a metric TSP-instance.

(c) The following algorithm for approximating the optimal solution of a metric TSP is by Christofides:

> First, compute a minimal spanning tree $T = (V_T, E_T)$ of the complete graph K_n with distance matrix (d_{ij}). Let O be the nodes of T which have odd degree (w.r.t. T!). Consider now the complete graph K_O consisting only of these nodes O, and calculate a minimal matching M for it, i.e., find a subset M of the edges of K_O s.t. every node is connected to exactly one other node (no loops) and the total weight of M, i.e., $\sum_{(i,j) \in M} d_{ij}$, is minimal. Add now the edges M to T yielding a multigraph G, i.e., assume that the original edges of T are colored black, while those of M are colored red. Still, the weight of an edge (i, j) in G is d_{ij} independent of its color. Calculate a Eulerian walk of G, i.e., a path of G which uses every edge, both black and red, of G exactly once. The approximation is then the tour embedded in the Eulerian walk.

- Convince yourself that every step of the algorithm by Christofides can be implemented in polynomial time (look it up on the Internet).
- Apply the algorithm by Christofides to the following example:



The coordinates of the inner nodes are $R \cdot (\cos\frac{k\cdot 2\pi}{n}, \sin\frac{k\cdot 2\pi}{n})$, for the outer nodes $(R+c)\cdot(\cos\frac{k\cdot 2\pi}{n}, \sin\frac{k\cdot 2\pi}{n})$ where $n = 6$, $k = 1, 2, \ldots, 6$, $R = 2cm$, $c = 0.5cm$. Distances are given by the Euclidean norm if there is an edge, otherwise ∞.

- Try to show that Christofides' algorithm always yields a tour which is at most 50% longer than the optimal tour for a metric TSP-instance.