# Homework 7

**Released on November 25, 2016**

**Note:** This set of problems is for practice. You do not need to hand in your solutions.

---

**1.** Consider the following matching problems:

**Maximum Weighted Matching (MWM):** Given an undirected graph $G = (V, E)$ and weight function $w : E \to \mathbb{R}$, find a matching $M$ of maximum total weight.

**Minimum Weight Perfect Matching (MWPM):** Given an undirected graph $G = (V, E)$ and weight function $w : E \to \mathbb{R}$, find a *perfect matching* $M$ of minimum total weight.

1. For MWPM, explain that we can assume w.l.o.g. that a perfect matching of $G$ always exist and $w_e \geq 0$ for all $e \in E$.
2. Consider MWPM on $G$ and a subgraph $H \subseteq G$. Let $M_G, M_H$ be the optimal solutions of $G$ and $H$ respectively. Let $w(M_G)$, $w(M_H)$ be the total weight of $M_G$ and $M_H$ respectively. As $H$ is a subgraph of $G$, do we have $w(M_H) \leq w(M_G)$? (Prove the claim if "yes", else give a counter example.)
3. Assume we have an algorithm $\mathbb{A}_{\min}$ for MWPM. Show that we can solve MWM by applying $\mathbb{A}_{\min}$ (*i.e.*, show that MWM can be transformed into MWPM).

**2.** A sorting instance $I$ is given by an array $A[1..n] \in \mathbb{N}$. Present a sorting algorithm which runs in $O(|I|)$, where $|I|$ is the encoding length of $I$ (using binary encoding).

**3.** Consider the decision version $\mathrm{VC}_k$ of **vertex cover**: Given an undirected graph $G = (V, E)$ and an integer $k$, does exist a vertex cover $C \subseteq V$ of cardinality $|C| \leq k$? (A vertex cover $C$ is a set of vertices such that every edge $e \in E$ is incident with an vertex of $C$.) Show that if there exists a polynomial time algorithm $\mathbb{A}_{\mathrm{K}}$ that tells whether $\mathrm{VC}_k$ is yes or no, then there exists a polynomial time algorithm for solving vertex cover (*i.e.*, present an algorithm which returns a minimum "vertex cover", not only the minimum cardinality!).

**4.** Consider the **linear programming problem**: Given $A \in \mathbb{R}^{m \times n}$, $c \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$, find a solution $x \in \mathbb{R}^n$ with $Ax \leq b$ such that $c^T x$ is minimized (assuming that the optimal objective value is not $-\infty$). The problem can also be described as follows:

$$\begin{aligned} \min\ & c^T x \\ \text{subject to } & Ax \leq b, \\ & x \geq 0. \end{aligned} \tag{LP}$$

For any $x \in \mathbb{R}^n$ and $x \geq 0$, if $Ax \leq b$, we call $x$ is a feasible solution of (LP). The **dual problem** of (LP) is defined as below:

$$\begin{aligned} \max\ & b^T y \\ \text{subject to } & A^T y \geq c, \\ & y \geq 0. \end{aligned} \tag{DP}$$

Note that $y \in \mathbb{R}^m$.

**Theorem 1** (Duality Theorem)**.** *Consider the above linear program (LP) and its dual program (DP). It holds that*

1. *$b^T y \leq c^T x$ for all feasible solutions $x$ of (LP), $y$ of (DP).*
2. *The optimal objective values of (LP) and (DP) are equal.*

1. Write down the corresponding decision problem of (LP) and the associated language $L$.
2. Write down the complement $\bar{L}$.
3. Prove that $L \in \mathcal{NP} \cap$ co-$\mathcal{NP}$, *i.e.*, show that $L, \bar{L} \in \mathcal{NP}$. Proving $\bar{L} \in \mathcal{NP}$ is non-trivial. You may need to use the duality theorem.

**5.** Prove the following claims:

(1) $\mathcal{P} =$ co-$\mathcal{P}$;

(2) $\mathcal{P} \subseteq \mathcal{NP} \cap \text{co} - \mathcal{NP}$;

(3) If $\mathcal{P} = \mathcal{NP}$, then $\mathcal{NP} =$ co-$\mathcal{NP}$. (Thus, to prove $\mathcal{P} \neq \mathcal{NP}$, it suffices to show $\mathcal{NP} \neq \text{co} - \mathcal{NP}$.)

**6.** Consider the *longest path problem*: Given an undirected graph $G = (V, E)$ and a distance function $l : E \to \mathbb{R}^+$. We are also given a source node $s \in V$ and a sink node $t \in V$. The question is to find a simple $s$-$t$ path $P$ such that the total length of $P$ is maximized.

(1) write down the decision version of the longest path problem.

(2) prove that the problem is NP-complete.

**7.** Consider the KNAPSACK problem: Given a knapsack of capacity $W$, $n$ items of values $v_i$ and weights $w_i$ for $i = 1, \ldots, n$. The goal is to pack a set $S$ of items into the knapsack such that the total value of $S$ is maximized. We require that the total weight of $S$ cannot exceed $W$. Show that KNAPSACK is NP-hard by reduction from PARTITION (*i.e.*, show the decision version is NP-complete).

**8.** Prove that SUBSET-SUM is NP-complete by reduction from PARTITION.

**9.** You are given a directed graph $G = (V, E)$ with edge wights $w_e$ on its edges $e \in E$. The weights can be negative or positive. The *zero-weight-cycle problem* is to decide if there is a simple cycle in $G$ so that the sum of the edge weights on this cycle is exactly 0. Prove that this problem is NP-complete.
*Hint:* You may prove by reduction from SUBSET-SUM.

**10.** Consider the greedy algorithm for KNAPSACK: Order items w.r.t. non-increasing value densities, *i.e.*,

$$\frac{v_1}{w_1} \geq \frac{v_2}{w_2} \geq \ldots \frac{v_n}{w_n}.$$

Pack items to the knapsack in this order until no items can be added into the knapsack.

1. Show that the greedy algorithm can be arbitrarily bad.

2. Now consider another algorithm ALG: First run the greedy algorithm and let $g$ be the total value of the greedy solution; Second, return the corresponding solution of $\max\{v_{\max}, g\}$, where $v_{\max} = \max_{i:w_i \leq W} v_i$. Show that ALG is a 2-approximation algorithm.